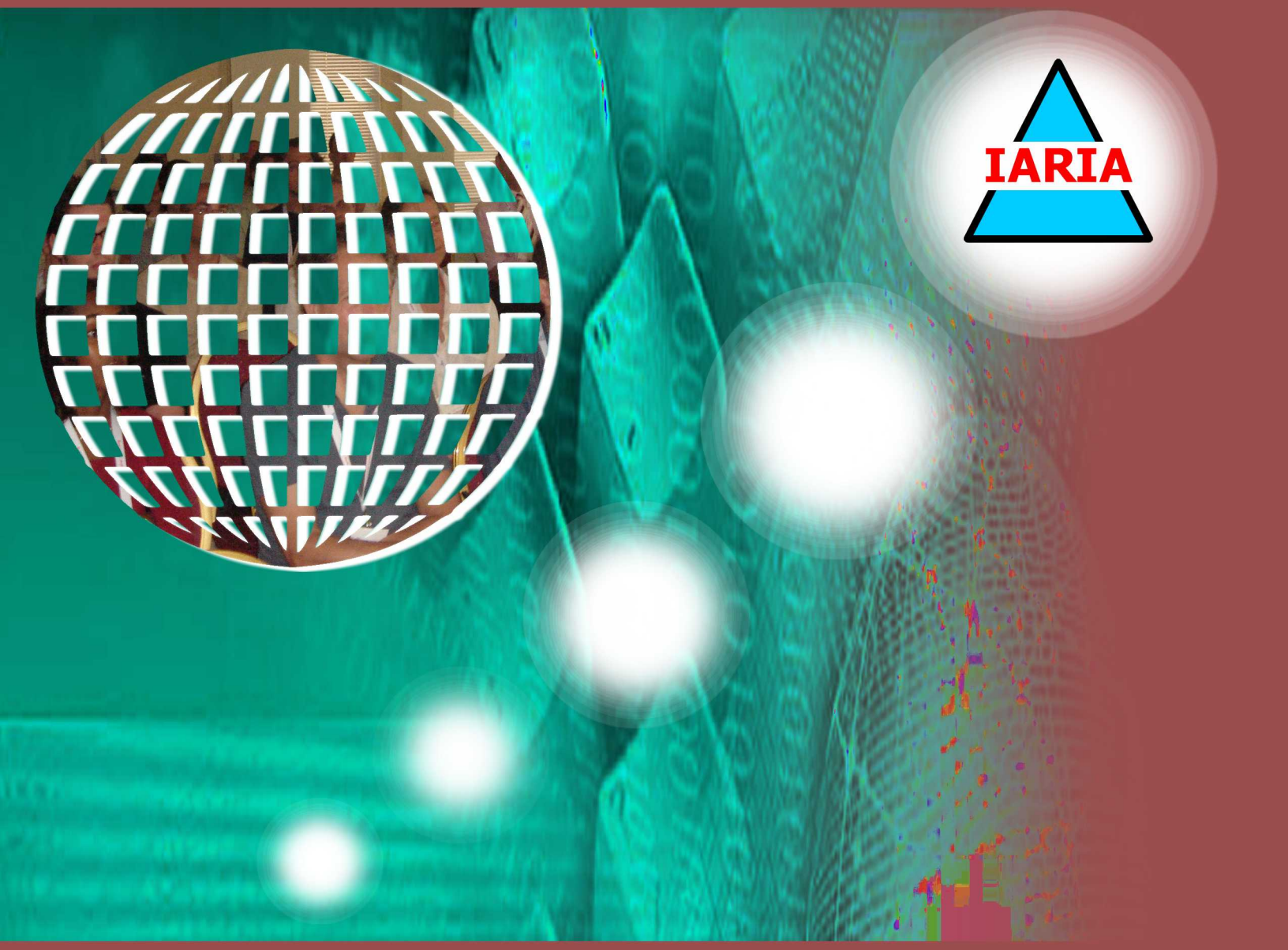
International Journal on Advances in Software



2023 vol. 16 nr. 3&4

The *International Journal on Advances in Software* is published by IARIA.
ISSN: 1942-2628
journals site: http://www.iariajournals.org
contact: petre@iaria.org

Sponsored by IARIA
www.iaria.org

International Journal on Advances in Software
Volume 16, Number 3 & 4, 2023

Thierry Badard, Université Laval, Canada
Muneera Bano, International Islamic University - Islamabad, Pakistan
Fabian Barbato, Technology University ORT, Montevideo, Uruguay
Peter Baumann, Jacobs University Bremen / Rasdaman GmbH Bremen, Germany
Gabriele Bavota, University of Salerno, Italy
Grigorios N. Beligiannis, University of Western Greece, Greece
Noureddine Belkhatir, University of Grenoble, France
Jorge Bernardino, ISEC - Institute Polytechnic of Coimbra, Portugal
Rudolf Berrendorf, Bonn-Rhein-Sieg University of Applied Sciences - Sankt Augustin, Germany
Ateet Bhalla, Independent Consultant, India
Fernando Boronat Seguí, Universidad Politecnica de Valencia, Spain
Pierre Borne, Ecole Centrale de Lille, France
Farid Bourennani, University of Ontario Institute of Technology (UOIT), Canada
Narhimene Boustia, Saad Dahlab University - Blida, Algeria
Hongyu Pei Breivold, ABB Corporate Research, Sweden
Carsten Brockmann, Universität Potsdam, Germany
Antonio Bucchiarone, Fondazione Bruno Kessler, Italy
Georg Buchgeher, Software Competence Center Hagenberg GmbH, Austria
Dumitru Burdescu, University of Craiova, Romania
Martine Cadot, University of Nancy / LORIA, France
Isabel Candal-Vicente, Universidad Ana G. Méndez, Puerto Rico
Juan-Vicente Capella-Hernández, Universitat Politècnica de València, Spain
Jose Carlos Metrolho, Polytechnic Institute of Castelo Branco, Portugal
Alain Casali, Aix-Marseille University, France
Yaser Chaaban, Leibniz University of Hanover, Germany
Savvas A. Chatzichristofis, Democritus University of Thrace, Greece
Antonin Chazalet, Orange, France
Jiann-Liang Chen, National Dong Hwa University, China
Shiping Chen, CSIRO ICT Centre, Australia
Wen-Shiung Chen, National Chi Nan University, Taiwan
Zhe Chen, College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, China PR
Yoonsik Cheon, The University of Texas at El Paso, USA
Lau Cheuk Lung, INE/UFSC, Brazil
Robert Chew, Lien Centre for Social Innovation, Singapore
Andrew Connor, Auckland University of Technology, New Zealand
Rebeca Cortázar, University of Deusto, Spain
Noël Crespi, Institut Telecom, Telecom SudParis, France
Carlos E. Cuesta, Rey Juan Carlos University, Spain
Duilio Curcio, University of Calabria, Italy
Mirela Danubianu, "Stefan cel Mare" University of Suceava, Romania
Paulo Asterio de Castro Guerra, Tapijara Programação de Sistemas Ltda. - Lambari, Brazil
Cláudio de Souza Baptista, University of Campina Grande, Brazil
Maria del Pilar Angeles, Universidad Nacional Autonónoma de México, México
Rafael del Vado Vírseda, Universidad Complutense de Madrid, Spain
Giovanni Denaro, University of Milano-Bicocca, Italy

Nirmit Desai, IBM Research, India
Vincenzo Deufemia, Università di Salerno, Italy
Leandro Dias da Silva, Universidade Federal de Alagoas, Brazil
Javier Diaz, Rutgers University, USA
Nicholas John Dingle, University of Manchester, UK
Roland Dodd, CQUniversity, Australia
Aijuan Dong, Hood College, USA
Suzana Dragicevic, Simon Fraser University- Burnaby, Canada
Cédric du Mouza, CNAM, France
Ann Dunkin, Palo Alto Unified School District, USA
Jana Dvorakova, Comenius University, Slovakia
Hans-Dieter Ehrich, Technische Universität Braunschweig, Germany
Jorge Ejarque, Barcelona Supercomputing Center, Spain
Atilla Elçi, Aksaray University, Turkey
Khaled El-Fakih, American University of Sharjah, UAE
Gledson Elias, Federal University of Paraíba, Brazil
Sameh Elnikety, Microsoft Research, USA
Fausto Fasano, University of Molise, Italy
Michael Felderer, University of Innsbruck, Austria
João M. Fernandes, Universidade de Minho, Portugal
Luis Fernandez-Sanz, University of de Alcala, Spain
Felipe Ferraz, C.E.S.A.R, Brazil
Adina Magda Florea, University "Politehnica" of Bucharest, Romania
Wolfgang Fohl, Hamburg Universiy, Germany
Simon Fong, University of Macau, Macau SAR
Gianluca Franchino, Scuola Superiore Sant'Anna, Pisa, Italy
Naoki Fukuta, Shizuoka University, Japan
Martin Gaedke, Chemnitz University of Technology, Germany
Félix J. García Clemente, University of Murcia, Spain
José García-Fanjul, University of Oviedo, Spain
Felipe Garcia-Sanchez, Universidad Politecnica de Cartagena (UPCT), Spain
Michael Gebhart, Gebhart Quality Analysis (QA) 82, Germany
Tejas R. Gandhi, Virtua Health-Marlton, USA
Andrea Giachetti, Università degli Studi di Verona, Italy
Afzal Godil, National Institute of Standards and Technology, USA
Luis Gomes, Universidade Nova Lisboa, Portugal
Pascual Gonzalez, University of Castilla-La Mancha, Spain
Björn Gottfried, University of Bremen, Germany
Victor Govindaswamy, Texas A&M University, USA
Gregor Grambow, AristaFlow GmbH, Germany
Christoph Grimm, University of Kaiserslautern, Austria
Michael Grottke, University of Erlangen-Nuernberg, Germany
Vic Grout, Glyndwr University, UK
Ensar Gul, Marmara University, Turkey
Richard Gunstone, Bournemouth University, UK
Zhensheng Guo, Siemens AG, Germany

Ismail Hababeh, German Jordanian University, Jordan
Shahliza Abd Halim, Lecturer in Universiti Teknologi Malaysia, Malaysia
Herman Hartmann, University of Groningen, The Netherlands
Jameleddine Hassine, King Fahd University of Petroleum & Mineral (KFUPM), Saudi Arabia
Tzung-Pei Hong, National University of Kaohsiung, Taiwan
Peizhao Hu, NICTA, Australia
Chih-Cheng Hung, Southern Polytechnic State University, USA
Edward Hung, Hong Kong Polytechnic University, Hong Kong
Noraini Ibrahim, Universiti Teknologi Malaysia, Malaysia
Anca Daniela Ionita, University "POLITEHNICA" of Bucharest, Romania
Chris Ireland, Open University, UK
Kyoko Iwasawa, Takushoku University - Tokyo, Japan
Mehrshid Javanbakht, Azad University - Tehran, Iran
Wassim Jaziri, ISIM Sfax, Tunisia
Dayang Norhayati Abang Jawawi, Universiti Teknologi Malaysia (UTM), Malaysia
Jinyuan Jia, Tongji University. Shanghai, China
Maria Joao Ferreira, Universidade Portucalense, Portugal
Ahmed Kamel, Concordia College, Moorhead, Minnesota, USA
Teemu Kanstrén, VTT Technical Research Centre of Finland, Finland
Nittaya Kerdprasop, Suranaree University of Technology, Thailand
Ayad ali Keshlaf, Newcastle University, UK
Nhien An Le Khac, University College Dublin, Ireland
Sadegh Kharazmi, RMIT University - Melbourne, Australia
Kyoung-Sook Kim, National Institute of Information and Communications Technology, Japan
Youngjae Kim, Oak Ridge National Laboratory, USA
Cornel Klein, Siemens AG, Germany
Alexander Knapp, University of Augsburg, Germany
Radek Koci, Brno University of Technology, Czech Republic
Christian Kop, University of Klagenfurt, Austria
Michal Krátký, VŠB - Technical University of Ostrava, Czech Republic
Narayanan Kulathuramaiyer, Universiti Malaysia Sarawak, Malaysia
Satoshi Kurihara, Osaka University, Japan
Eugenijus Kurilovas, Vilnius University, Lithuania
Alla Lake, Linfo Systems, LLC, USA
Fritz Laux, Reutlingen University, Germany
Luigi Lavazza, Università dell'Insubria, Italy
Fábio Luiz Leite Júnior, Universidade Estadual da Paraiba,Brazil
Alain Lelu, University of Franche-Comté / LORIA, France
Cynthia Y. Lester, Georgia Perimeter College, USA
Clement Leung, Hong Kong Baptist University, Hong Kong
Weidong Li, University of Connecticut, USA
Corrado Loglisci, University of Bari, Italy
Francesco Longo, University of Calabria, Italy
Sérgio F. Lopes, University of Minho, Portugal
Pericles Loucopoulos, Loughborough University, UK
Alen Lovrencic, University of Zagreb, Croatia

Qifeng Lu, MacroSys, LLC, USA
Xun Luo, Qualcomm Inc., USA
Stephane Maag, Telecom SudParis, France
Ricardo J. Machado, University of Minho, Portugal
Maryam Tayefeh Mahmoudi, Research Institute for ICT, Iran
Nicos Malevris, Athens University of Economics and Business, Greece
Herwig Mannaert, University of Antwerp, Belgium
José Manuel Molina López, Universidad Carlos III de Madrid, Spain
Francesco Marcelloni, University of Pisa, Italy
Eda Marchetti, Consiglio Nazionale delle Ricerche (CNR), Italy
Gerasimos Marketos, University of Piraeus, Greece
Abel Marrero, Bombardier Transportation, Germany
Adriana Martin, Universidad Nacional de la Patagonia Austral / Universidad Nacional del Comahue, Argentina
Goran Martinovic, J.J. Strossmayer University of Osijek, Croatia
Paulo Martins, University of Trás-os-Montes e Alto Douro (UTAD), Portugal
Stephan Mäs, Technical University of Dresden, Germany
Constandinos Mavromoustakis, University of Nicosia, Cyprus
Jose Merseguer, Universidad de Zaragoza, Spain
Seyedeh Leili Mirtaheri, Iran University of Science & Technology, Iran
Lars Moench, University of Hagen, Germany
Yasuhiko Morimoto, Hiroshima University, Japan
Antonio Navarro Martín, Universidad Complutense de Madrid, Spain
Filippo Neri, University of Naples, Italy
Muaz A. Niazi, Bahria University, Islamabad, Pakistan
Natalja Nikitina, KTH Royal Institute of Technology, Sweden
Roy Oberhauser, Aalen University, Germany
Pablo Oliveira Antonino, Fraunhofer IESE, Germany
Rocco Oliveto, University of Molise, Italy
Sascha Opletal, Universität Stuttgart, Germany
Flavio Oquendo, European University of Brittany/IRISA-UBS, France
Claus Pahl, Dublin City University, Ireland
Marcos Palacios, University of Oviedo, Spain
Constantin Paleologu, University Politehnica of Bucharest, Romania
Kai Pan, UNC Charlotte, USA
Yiannis Papadopoulos, University of Hull, UK
Andreas Papasalouros, University of the Aegean, Greece
Rodrigo Paredes, Universidad de Talca, Chile
Päivi Parviainen, VTT Technical Research Centre, Finland
João Pascoal Faria, Faculty of Engineering of University of Porto / INESC TEC, Portugal
Fabrizio Pastore, University of Milano - Bicocca, Italy
Kunal Patel, Ingenuity Systems, USA
Óscar Pereira, Instituto de Telecomunicacoes - University of Aveiro, Portugal
Willy Picard, Poznań University of Economics, Poland
Jose R. Pires Manso, University of Beira Interior, Portugal
Sören Pirk, Universität Konstanz, Germany
Meikel Poess, Oracle Corporation, USA

Thomas E. Potok, Oak Ridge National Laboratory, USA
Christian Prehofer, Fraunhofer-Einrichtung für Systeme der Kommunikationstechnik ESK, Germany
Ela Pustułka-Hunt, Bundesamt für Statistik, Neuchâtel, Switzerland
Mengyu Qiao, South Dakota School of Mines and Technology, USA
Kornelije Rabuzin, University of Zagreb, Croatia
J. Javier Rainer Granados, Universidad Politécnica de Madrid, Spain
Muthu Ramachandran, Leeds Metropolitan University, UK
Thurasamy Ramayah, Universiti Sains Malaysia, Malaysia
Prakash Ranganathan, University of North Dakota, USA
José Raúl Romero, University of Córdoba, Spain
Henrique Rebêlo, Federal University of Pernambuco, Brazil
Hassan Reza, UND Aerospace, USA
Elvinia Riccobene, Università degli Studi di Milano, Italy
Daniel Riesco, Universidad Nacional de San Luis, Argentina
Mathieu Roche, LIRMM / CNRS / Univ. Montpellier 2, France
José Rouillard, University of Lille, France
Siegfried Rouvrais, TELECOM Bretagne, France
Claus-Peter Rückemann, Universität Münster / DIMF / Leibniz Universität Hannover, Germany
Djamel Sadok, Universidade Federal de Pernambuco, Brazil
Ismael Sanz, Universitat Jaume I, Spain
M. Saravanan, Ericsson India Pvt. Ltd -Tamil Nadu, India
Idrissa Sarr, University of Cheikh Anta Diop, Dakar, Senegal / University of Quebec, Canada
Patrizia Scandurra, University of Bergamo, Italy
Daniel Schall, Vienna University of Technology, Austria
Rainer Schmidt, Munich University of Applied Sciences, Germany
Sebastian Senge, TU Dortmund, Germany
Isabel Seruca, Universidade Portucalense - Porto, Portugal
Kewei Sha, Oklahoma City University, USA
Simeon Simoff, University of Western Sydney, Australia
Jacques Simonin, Institut Telecom / Telecom Bretagne, France
Cosmin Stoica Spahiu, University of Craiova, Romania
George Spanoudakis, City University London, UK
Cristian Stanciu, University Politehnica of Bucharest, Romania
Lena Strömbäck, SMHI, Sweden
Osamu Takaki, Japan Advanced Institute of Science and Technology, Japan
Antonio J. Tallón-Ballesteros, University of Seville, Spain
Wasif Tanveer, University of Engineering & Technology - Lahore, Pakistan
Ergin Tari, Istanbul Technical University, Turkey
Steffen Thiel, Furtwangen University of Applied Sciences, Germany
Jean-Claude Thill, Univ. of North Carolina at Charlotte, USA
Pierre Tiako, Langston University, USA
Božo Tomas, HT Mostar, Bosnia and Herzegovina
Davide Tosi, Università degli Studi dell'Insubria, Italy
Dragos Truscan, Åbo Akademi University, Finland
Chrisa Tsinaraki, Technical University of Crete, Greece
Roland Ukor, FirstLinq Limited, UK

Torsten Ullrich, Fraunhofer Austria Research GmbH, Austria
José Valente de Oliveira, Universidade do Algarve, Portugal
Dieter Van Nuffel, University of Antwerp, Belgium
Shirshu Varma, Indian Institute of Information Technology, Allahabad, India
Konstantina Vassilopoulou, Harokopio University of Athens, Greece
Miroslav Velev, Aries Design Automation, USA
Tanja E. J. Vos, Universidad Politécnica de Valencia, Spain
Krzysztof Walczak, Poznan University of Economics, Poland
Yandong Wang, Wuhan University, China
Rainer Weinreich, Johannes Kepler University Linz, Austria
Stefan Wesarg, Fraunhofer IGD, Germany
Wojciech Wiza, Poznan University of Economics, Poland
Martin Wojtczyk, Technische Universität München, Germany
Hao Wu, School of Information Science and Engineering, Yunnan University, China
Mudasser F. Wyne, National University, USA
Zhengchuan Xu, Fudan University, P.R.China
Yiping Yao, National University of Defense Technology, Changsha, Hunan, China
Stoyan Yordanov Garbatov, Instituto de Engenharia de Sistemas e Computadores - Investigação e Desenvolvimento, INESC-ID, Portugal
Weihai Yu, University of Tromsø, Norway
Wenbing Zhao, Cleveland State University, USA
Hong Zhu, Oxford Brookes University, UK
Martin Zinner, Technische Universität Dresden, Germany

International Journal on Advances in Software
Volume 16, Numbers 3 & 4, 2023

CONTENTS

VR-GitCity: Immersively Visualizing Git Repository Evolution Using a City Metaphor in Virtual Reality

Roy Oberhauser[0000-0002-7606-8226]
Computer Science Dept.
Aalen University
Aalen, Germany
e-mail: roy.oberhauser@hs-aalen.de

Abstract – **The increasing demand for software functionality necessitates an increasing amount of program source code that is retained and managed in version control systems, such as Git. As the number, size, and complexity of Git repositories increases, so does the number of collaborating developers, maintainers, and other stakeholders over a repository's lifetime. In particular, visual limitations of command line or two-dimensional graphical Git tooling can hamper repository comprehension, analysis, and collaboration across one or multiple repositories when a larger stakeholder spectrum is involved. This is especially true for depicting repository evolution over time. This paper contributes VR-GitCity, a Virtual Reality (VR) solution concept for visualizing and interacting with Git repositories in VR. The evolution of the code base is depicted via a 3D treemap utilizing a city metaphor, while the commit history is visualized as vertical planes. Our prototype realization shows its feasibility, and our evaluation results based on a case study show its depiction, comprehension, analysis, and collaboration capabilities for evolution, branch, commit, and multi-repository analysis scenarios.**

Keywords – Git; virtual reality; visualization; version control systems; software configuration management; city metaphor.

I. INTRODUCTION

This paper is an extended version of our original paper on VR-Git [1] and extends our solution to VR-GitCity, which incorporates a city metaphor.

In this digitalization era, the global demand for software functionality is increasing across all areas of society, and with it there is a correlating necessity for storing and managing the large number of underlying program source code files that represent the instructions inherent in software. Program source code is typically stored and managed in repositories within version control systems, currently the most popular being Git. Since these repositories are often shared, various cloud-based service providers offer Git functionality, including GitHub, BitBucket, and GitLab. GitHub reports over 305m repositories [2] with over 91m users [3]. Even within a single company, the source code portfolio can become very large, as exemplified with the over 2bn Lines Of Code (LOC) accessed by 25k developers at Google [4]. Over 25m professional software developers worldwide [5] continue to add source code to private and public repositories.

To gain insights into these code repositories, various command-line, visual tools, and web interfaces are provided. Yet, repository analysis can be challenging due to the potentially large number of files involved, and the added complexity of branches, commits, and users involved over the history of a repository. Furthermore, the analysis can be hampered by the limited visual space available for analysis. It can be especially difficult for those stakeholders unfamiliar with a repository, or for collaborating with stakeholders who may not be developers but have a legitimate interest in insights to code development. Possible scenarios include someone transferred to the development team (ramp-up), joining an open-source code project, quality assurance activities, forensic or intellectual property analysis, maintenance activities, defect or resolution tracking, repository fork analysis, etc. Furthermore, while repositories are dynamic and retain historical information, it nevertheless can be challenging to readily convey these aspects intuitively using conventional Git tooling. Especially as the size of a repository grows in number of elements (subfolders and files), it becomes difficult to comprehend the "big picture" as to how it has been evolving, which areas were the focus for changing or adding code when, and depicting the final state.

Virtual Reality (VR) is a mediated visual environment which is created and then experienced as telepresence by the perceiver. VR provides an unlimited immersive space for visualizing and analyzing models and their interrelationships simultaneously in a 3D spatial structure viewable from different perspectives. As repository models grow in size and complexity, an immersive digital environment provides additional visualization capabilities to comprehend and analyze code repositories and include and collaborate with a larger spectrum of stakeholders.

As to our prior work with VR for software engineering, VR-UML [6] provides VR-based visualization of the Unified Modeling Language (UML) and VR-SysML [7] for Systems Modeling Language (SysML) diagrams. Our original paper described VR-Git [1], a solution concept for visualizing and interacting with Git repositories in VR. This paper contributes VR-GitCity, which extends our VR-Git visualization capabilities to incorporate a 3D treemap using a city metaphor to convey repository evolution of relative files sizes in Lines of Code (LOC), while using vertical planes for branch and commit analysis. Our prototype realization shows its feasibility, and a case-based evaluation provides insights into its capabilities for repository comprehension, analysis and collaboration.

The remainder of this paper is structured as follows: Section 2 discusses related work. In Section 3, the solution

concept is described. Section 4 provides details about the realization. The evaluation is described in Section 5 and is followed by a conclusion.

II. Related Work

With regard to VR-based Git visualization, Bjørklund [8] used a directed acyclic graph visualization in VR using the Unreal Engine, with a backend using NodeJS, Mongoose, and ExpressJS, with SQLite used to store data. GitHub Skyline [9] provides a VR Ready 3D contribution graph as an animated skyline that can be annotated.

For non-VR based Git visualization, RepoVis [10] provides a comprehensive visual overview and search facilities using a 2D JavaScript-based web application and Ruby-based backend with a CouchDB. Githru [11] utilizes graph reconstruction, clustering, and context-preserving squash merge to abstract a large-scale commit graph, providing an interactive summary view of the development history. VisGi [12] utilizes tagging to aggregate commits for a coarse group graph, and Sunburst Tree Layout diagrams to visualize group contents. It is interesting to note that the paper states "showing all groups at once overloads the available display space, making any two-dimensional visualization cluttered and uninformative. The use of an interactive model is important for clean and focused visualizations." UrbanIt [13] utilizes an iPad to support mobile Git visualization aspects, such as an evolution view. Besides the web-based visualization interfaces of Git cloud providers, various desktop Git tools, such as Sourcetree and Gitkracken, provide typical 2D branch visualizations.

The city metaphor is a well-known software visualization paradigm. An early paper to apply it was the File System Navigator (FSN) [14], and although it did not explicitly use the word 'city,' it nevertheless used a landscape paradigm with a network of roads, buildings, and towns. MediaMetro [15] applies the metaphor to media documents. CodeCity [16] is a 3D software visualization approach based on a city metaphor with the Moose reengineering framework implemented in SmallTalk. In this context, Buildings represent classes, districts represent packages, and visible properties depict selected metrics. ExplorViz [17] uses a city metaphor for live trace exploration, implemented in JavaScript as a browser-based WebVR application using Oculus Rift together with Microsoft Kinect for gesture recognition. Code2City$_{VR}$ [18], which is a VR implementation of the previously mentioned CodeCity [16], focuses on metrics and smells for Java code.

In contrast to the above work, VR-GitCity utilizes a city metaphor for Git repositories in VR, depicting their dynamic evolution with regard to LOC size, while mapping familiar 2D visual Git constructs and commit content to VR to make its usage relatively intuitive without training. In contrast to other approaches that apply clustering, aggregating, merging, metrics, or data analytics, our concept preserves the chronological sequence of commits and retains their content details in support of practical analysis for Software Engineering (SE) tasks. To reduce visual clutter, detailed informational aspects of an element of interest can be obtained via the VR-Tablet.

III. Solution Concept

Our VR-Git solution concept is shown relative to our other VR solutions in Figure 1. VR-Git is based on our generalized VR Modeling Framework (VR-MF) (detailed in [14]). VR-MF provides a VR-based domain-independent hypermodeling framework addressing four aspects requiring special attention when modeling in VR: visualization, navigation, interaction, and data retrieval. Our VR-SE area includes VR-GitCity (a superset of our VR-Git) and the aforementioned VR-UML [6] and VR-SysML [7]. Since Enterprise Architecture (EA) can encompass SE models and development and be applicable for collaboration in VR. Our other VR modeling solutions in the EA area include: VR-EA [19] for visualizing EA ArchiMate models; VR-ProcessMine [20] for process mining and analysis; and VR-BPMN [21] for Business Process Modeling Notation (BPMN) models. VR-EAT [22] integrates the EA Tool (EAT) Atlas to provide dynamically-generated EA diagrams, while VR-EA+TCK [23] integrates Knowledge Management Systems (KMS) and/or Enterprise Content Management Systems (ECMS).

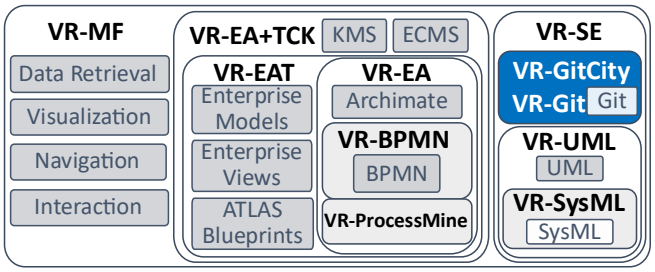


Figure 1. Conceptual map of our various VR solution concepts.

In support of our view that an immersive VR experience can be beneficial for a software analysis, Müller et al. [24] compared VR vs. 2D for a software analysis task, finding that VR does not significantly decrease comprehension and analysis time nor significantly improve correctness (although fewer errors were made). While interaction time was less efficient, VR improved the user experience, was more motivating, less demanding, more inventive/innovative, and more clearly structured.

A. Visualization in VR

A hyperplane is used to intuitively represent and group the commits related to a repository. Each commit is then represented by a vertical *commit plane*. These commit planes are then sequenced chronologically on the hyperplane as a set of planes. Since VR space is unlimited, we can thus convey the sequence of all commits in the repository. Each 2D plane then represents each file involved in that commit as a tile. These are then colored to be able to quickly determine what occurred. Green indicates a file was added, red a file removed, and blue that a file was modified. On the left side of the hyperplane, a transparent *branch plane* (branch perspective) perpendicular to the hyperplane and the commit planes depicts branches as an acyclic colored graph to indicate which branch is involved with a commit. This allows the user to travel down that side to follow a branch, see to which branch any commit relates, and to readily detect merges. Accordingly, the commit

planes are slightly offset vertically, as they dock to a branch, thus "deeper" or "higher" commits indicate how close or far they relatively are from the main branch. Via the anchor, commit planes can be manually collapsed (hidden), expanded, or moved to, for example, compare one commit with another side-by-side. In order to view the contents of a file, when a file tile is selected, a *content plane* (i.e., code view) extends above the commit plane to display the file contents.

B. Navigation in VR

A navigation challenge resulting from VR immersion is supporting intuitive spatial navigation while reducing potential VR sickness symptoms. We thus incorporate two navigation modes in our solution concept: gliding controls for fly-through VR (default), while teleporting instantly places the camera at a selected position either via the VR controls or by selection of a commit in our VR-Tablet. While teleporting is potentially disconcerting, it may reduce the likelihood of VR sickness induced by fly-through for those prone to it.

C. Interaction in VR

As VR interaction has not yet become standardized, in our concept we support user-element interaction primarily through VR controllers and a *VR-Tablet*. The VR-Tablet is used to provide detailed context-specific element information based on VR object selection, menu, scrolling, field inputs, and other inputs. It includes a *virtual keyboard* for text entry via laser pointer key selection. As another VR interaction element, we provide the aforementioned corner *anchor sphere* affordance, that supports moving, collapsing / hiding, or expanding / displaying hyperplanes or vertical commit planes.

IV. Realization

The logical architecture for our VR-GitCity prototype realization is shown in Figure 2. Basic visualization, navigation, and interaction functionality in our VR prototype is implemented with Unity 2020.3 and the OpenVR XR Plugin 1.1.4, shown in the Unity block (top left, blue). Scripts utilize Libgit2Sharp [25] to access the Git commit history of one or more repositories from within Unity. Thus, data about the repository is not stored in a separate database but accessed on-the-fly, avoiding synchronization, data-loss, storage format, transformation, and other issues. Note that only realization aspects not explicitly mentioned in the evaluation are described in this section to reduce redundancy.

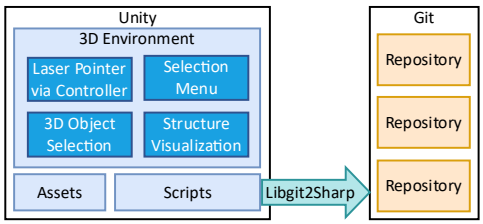


Figure 2. VR-GitCity logical architecture.

Internally, a tree data structure is used to represent a repository. Directory nodes represent a folder characterized by a name and path, and may contain files or subfolders (children). File nodes keep references to the parent directory node that contains them. Dictionaries are used to track the state and the size of a node, with the commit SHA serving as a key and used to identify the time of a change. With each SHA commit stored in the dictionary, the difference via a file compare is performed and the delta as Lines of Code (LOC) is retained.

For our city metaphor visualization, a directory node is a cuboid with equal x and z (depth) lengths. A bit matrix is used to distribute any children (folders or files) as compactly as possible, and these are stacked as cream-colored common height blocks in the y dimension (height). To follow the city metaphor, a building should be used to represent modules or in our case files. However, we wanted to primarily convey the dynamic evolution of the size of repository elements over time, rather than depicting any structural modularization. So, to be intuitive in visually transmitting the size information dynamically over time, we chose to utilize the metaphor of a glass of water and its fullness to represent the size state of any file. However, to not break with the city metaphor, these can be viewed as a glass skyscraper or glass elevator as shown in Figure 3. It is a cuboid in blue glass tone, with the maximum LOC file size ever reached (relative to all others) represented by its height. Its current size is transparent glass with a glass level to show how full it is currently (relative to max), with a more greyish tone above to differentiate anything less than full. The final size is marked by a grey fat slab (like concrete). Selecting a certain commit via the VR-Tablet will depict its changes to the repository by coloring the aqua cuboid green for added files, red for deleted files, and blue for changed files.

As to color choices, transparent (glass) was used as the default for building sides, in order to avoid buildings from hiding objects behind it, to better see the foundation, and for objects involved in a commit (opaque colors), to be more pronounced. This also permits the metaphor of a water glass with its fullness represented by slab levels. However, the building colors could readily be randomly distributed or custom-defined per object or folder by the user via a configuration file or the VR-Tablet. Also, in place of opaque colors, alternatively the border outlines of a building could be color-coded for the commit accordingly.

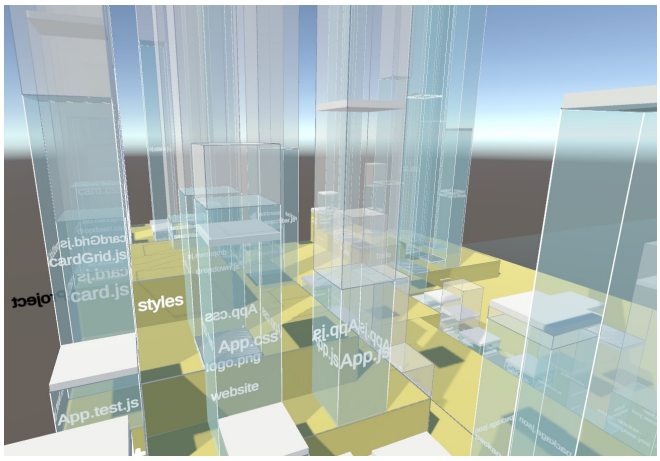


Figure 3. VR-GitCity: files as glass buildings stacked on their containing folders (box height is its all-time max LOC size, aqua slab its current size, and thick grey slab the final size).

The VR-Git view functionality supports detailed commit information visualization, with commits represented on vertical planes. To support interaction on specific commits, an anchor (ball) is placed on one corner of a hyperplane and is an affordance in order to move or expand/collapse an entire hyperplane, as shown in Figure 4. The anchors are also placed at the left bottom corner of all commit planes and colored and aligned with the branch with which they are associated.

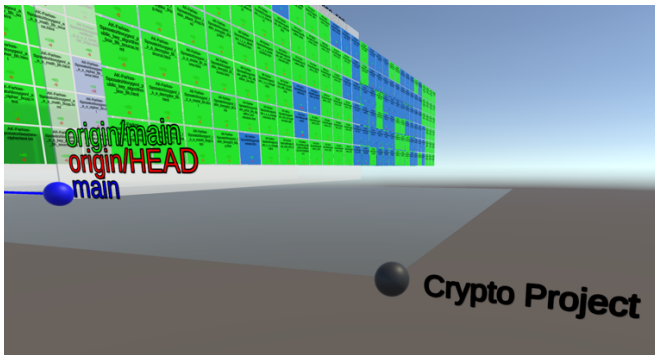


Figure 4. Vertical commit planes on hyperplane with anchor affordances.

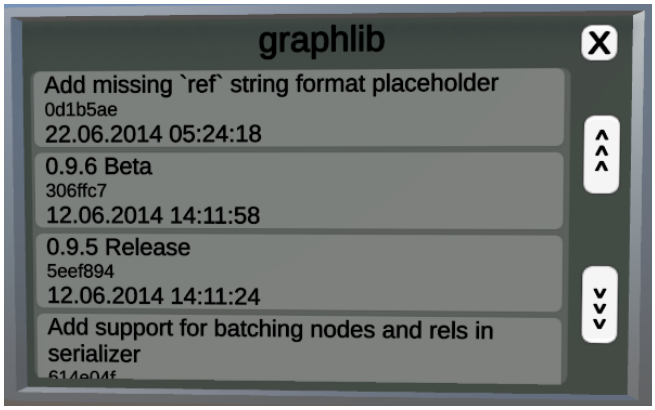


Figure 5. Viewing Git commit messages in VR-Tablet.

To support navigation, projects can be selected via the VR-Tablet and provide a teleporting capability to its project hyperplane or a specific commit plane. A list of Git commit messages including their commit ID (Secure Hash Algorithm 1 (SHA-1)) and the date and timestamp in the VR-Tablet, as shown in Figure 5.

V. Evaluation

For the evaluation of our solution concept, we refer to the design science method and principles [26], in particular, a viable artifact, problem relevance, and design evaluation (utility, quality, efficacy). For this, we use a case study applying four Git repository scenarios in VR:

A. Repository evolution scenario
B. Branch analysis scenario
C. Commit analysis scenario
D. Multi-Repository Analysis Scenario

A. Repository Evolution Scenario

To support repository evolution comprehension and analysis, the VR-GitCity city metaphor depiction is used. Two Repositories, denoted as R1 and R2, were used as the basis for the scenario depictions. Note that the counting utility ignored file types and code unfamiliar to it.

- R1 consists of approximately 23 (sub)folders, 99 files, and at least 47K LOC (across ~50 files with JSON, HTML, JS, XML, CSS, and Markdown), as can be seen in Figure 6.
- R2 consists of approximately 660 (sub)folders, 2700 files, and at least 123K LOC (across ~377 files with C# and JSON), as shown in Figure 7.

```
      65 text files.
classified 65 files
      65 unique files.
      25 files ignored.

github.com/AlDanial/cloc v 1.90  T=0.14 s (395.2 files/s, 353263.6 lines/s)
-------------------------------------------------------------------------------
Language                     files          blank        comment           code
-------------------------------------------------------------------------------
JSON                            15              0              0          39269
HTML                             3            985             84           6108
JavaScript                      17            131             94            974
XML                              4              2              0            640
CSS                              6            116              6            527
YAML                             3              5              0             56
Bourne Shell                     2             11              7             40
Markdown                         1             32              0             38
Dockerfile                       3             13              0             24
SVG                              1              0              0              1
-------------------------------------------------------------------------------
SUM:                            55           1295            191          47677
```

Figure 6. R1 cloc report.

```
|    2720 text files.
classified 2720 files
Duplicate file check 2720 files (1043 known unique)
    2698 unique files.
    2205 files ignored.

github.com/AlDanial/cloc v 1.90  T=1.10 s (841.0 files/s, 344980.1 lines/s)
-------------------------------------------------------------------------------
Language                     files          blank        comment           code
-------------------------------------------------------------------------------
Unity-Prefab                   410             23              0         177356
C#                             344          12318           7747          68221
JSON                            33              2              0          55119
XML                             19             83              1          45408
SVG                             77            138             77           5105
HLSL                            31            931            228           3585
XSD                              5            341             53           2129
Markdown                         4            259              0            683
C# Generated                     4             63             36            368
-------------------------------------------------------------------------------
SUM:                           927          14158           8142         357974
-------------------------------------------------------------------------------
```

Figure 7. R2 cloc report.

In VR, the front and side views of the repository show aligned stacked cream-colored blocks as (sub)folders that provide a single quick overview of the involved (sub)folders and their containing parent, as seen in Figure 8.

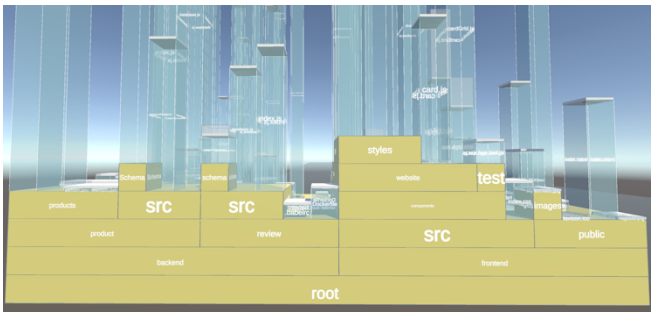


Figure 8. VR-GitCity R1 front view showing aligned stacked (sub)folders.

The deepest path of subfolders is placed on the front, thus the rear view typically shows a declining set of folder blocks with regard to height, as seen in Figure 9. Thus, the greatest depth of folder containment can be readily determined.

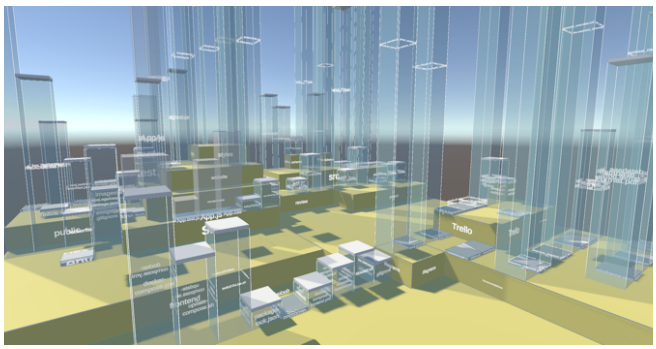


Figure 9. VR-GitCity R1 rear view with declining (sub)folder foundation.

Where applicable, additional parallel subfolders may be viewed from the left or right side based on placement, as seen in Figure 10. A closeup from the rear is shown in Figure 11.

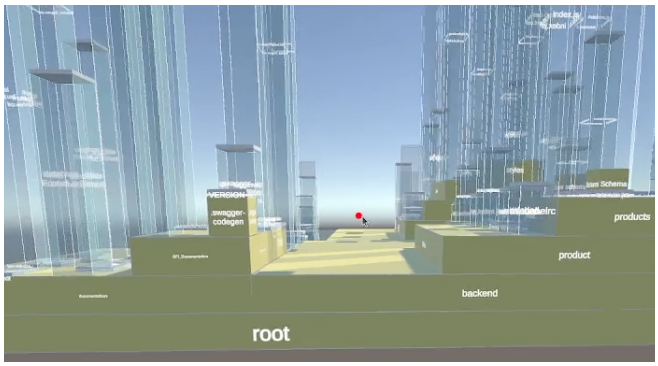


Figure 10. VR-GitCity R1 side view with parallel folder depiction.

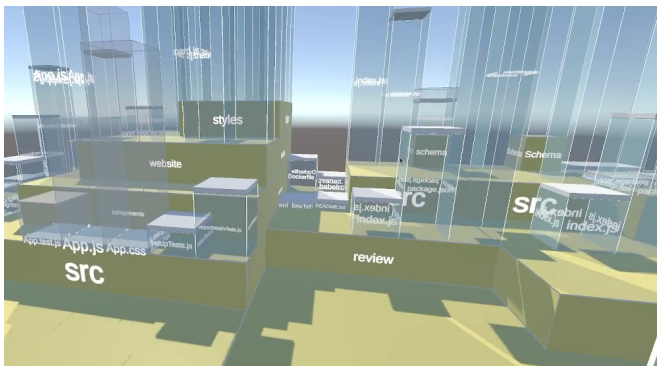


Figure 11. VR-GitCity R1 closeup from the rear.

When conveying the elements affected by a specific commit selected in the VR-Tablet, green is used to indicate that files were added, as shown in Figure 12. Red is used for deleted files, as shown in Figure 13. The glass thin slab level shows the current relative size in LOC compared to other files, with the max size being the cuboid overall height, with a thick grey slab conveying the final file size.

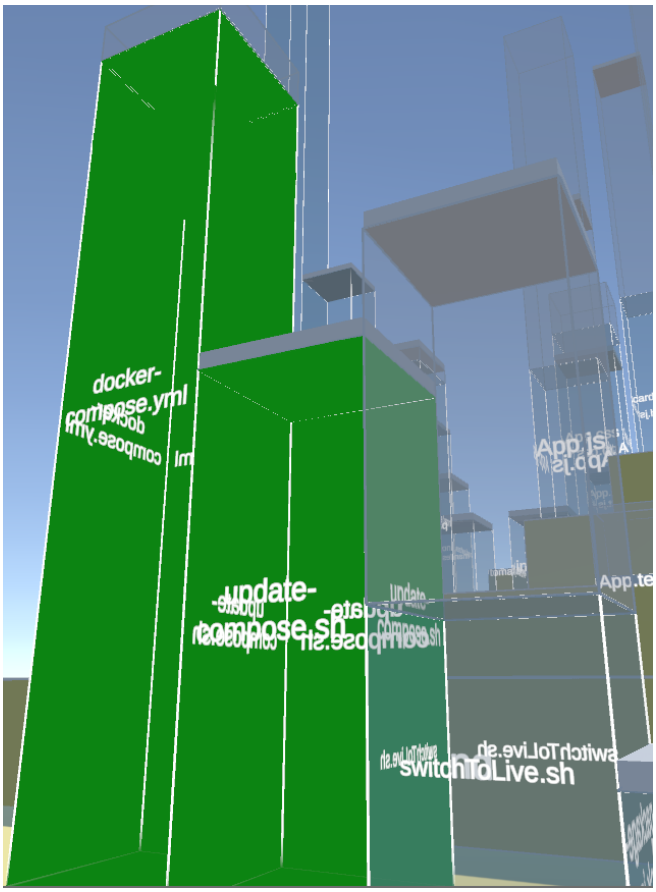


Figure 12. VR-GitCity R1 showing added files in green.

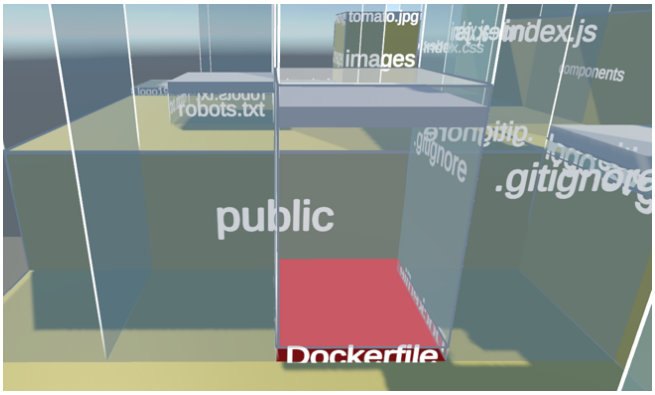


Figure 13. VR-GitCity R1 showing deleted files in red.

Blue is used to convey files that were changed by a commit, as shown in Figure 14. Thus, by scrolling through commits on the VR-Tablet, a dynamic changing picture equivalent to a video of the repository evolution is presented. In portraying the maximum size as well as the end final size, it can be understood to show the evolution of any single element to its maximum as well as end target size, while not forgetting the maximum it once had if it later shrunk (as a type of ghosting) or a file was removed.

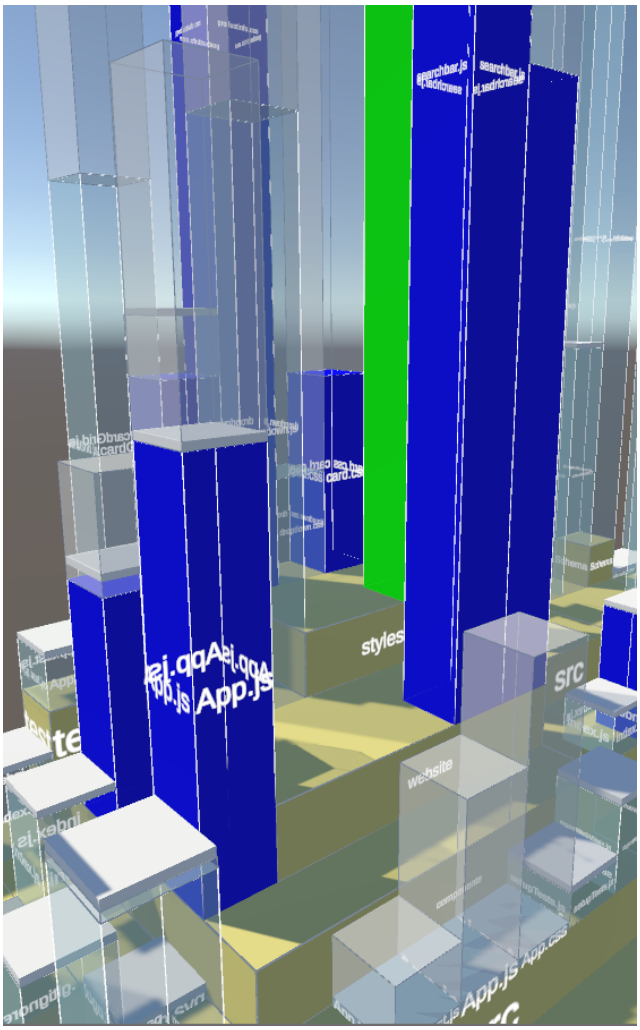


Figure 14. VR-GitCity R1 showing changed files in blue.

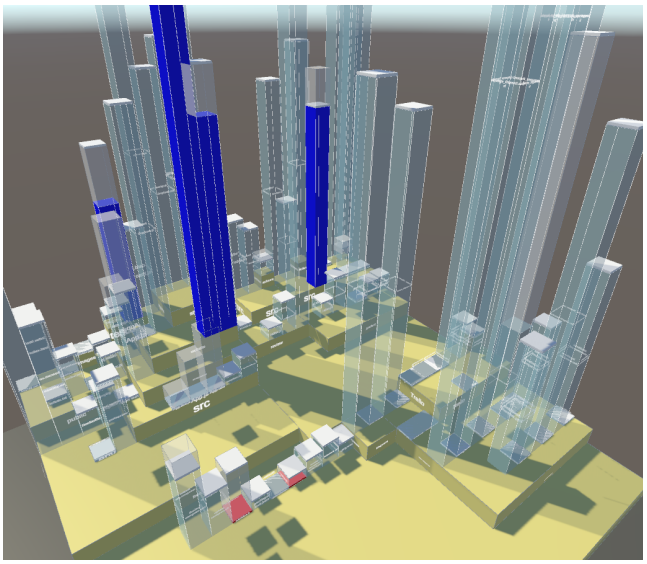


Figure 15. VR-GitCity R1 overview.

An overview of repository R1 can be seen in Figure 15. It visually and immersively conveys the grouping and containment of elements in subfolders, the number of files, the maximum relative sizes achieved, the current size as a fill level, and the affected elements by a specific commit via colors.

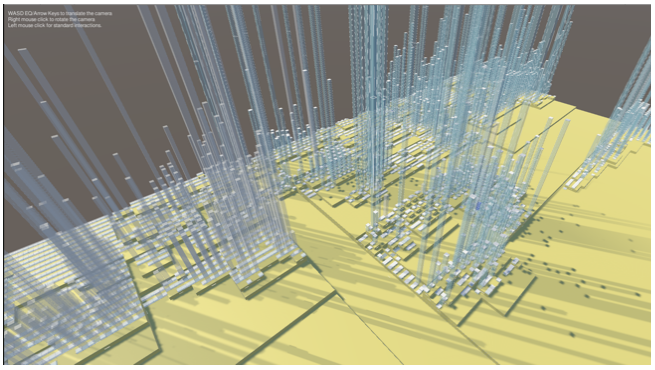


Figure 16. VR-GitCity R2 overview.

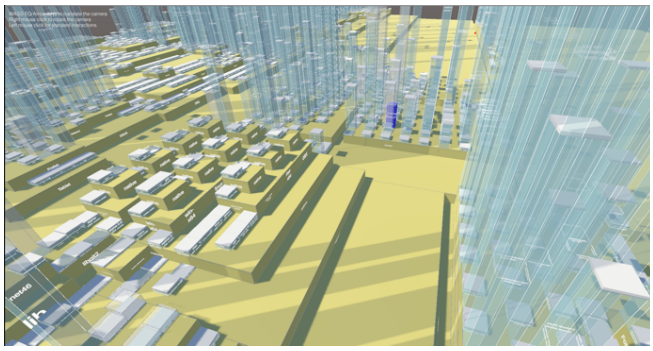


Figure 17. VR-GitCity R2 indicating a changed file in center as blue.

To test the scalability of the concept, the repository R2 was used and is shown in Figure 16 and Figure 17. Note that R2 consists of over 600 subfolders and almost 10K files. While these screenshots are not intended to be legible in this paper, they portray the capability of VR-GitCity to scale to very large repositories and provide a “big picture” of their evolution over time. Up close via immersion, fly-through navigation, selection, and the VR-Tablet, additional detailed information about an element or affected files in a commit can be determined.

B. Branch Analysis Scenario

To support branch analysis, our hyperplane concept with vertical planes is used. To the left side of a hyperplane, an invisible branch graph plane is rendered perpendicular to the hyperplane and a color-coded list of all the branches can be seen next to the first commit plane, seen in Figure 18. These colored labels can be used for orientation. By selecting a branch label, the user can be teleported to the first commit of that branch. We chose not to repeat the branch labels throughout the graph to reduce the textual visual clutter.

Figure 18. VR-Git branch overview.

The branch perspective of the hyperplane (its left side) shows a contiguous color-coded graph of the branches as shown in Figure 19. Commit plane heights offset based on the branch to which they are associated. This can provide a quick visual cue as to how relatively close or far the commit is from the main branch. A merge of two branches is shown in Figure 20.

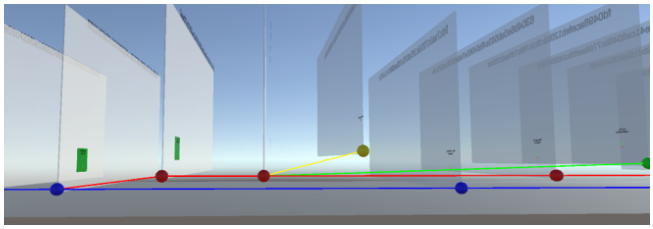


Figure 19. VR-Git branch tree graph.

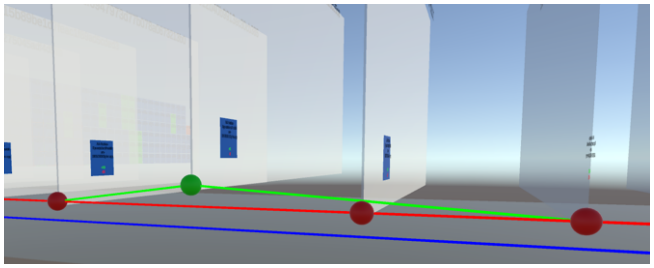


Figure 20. Branch merge.

```
* cdfb4e2 (HEAD -> development, origin/development) fixed header color
* 9ec0274 modified settings page
* 24ec5ab app icon improvement, design improvements
* fb22d80 updated icons
* 5353133 design improvements
* 76d23be #3 display number of tickets
* cbd1866 #6 changed font-size
* 9ec697f fixed package.json
* 5c4d842 improved refresh after redeem
* 144f743 added new icons
*   1fdf948 (tag: 0.1.3-beta, origin/master, origin/HEAD, master) Merge pull request #1 from Jay895/new-ui
|\
| * 357ff25 added ticket variation to list
| * 98b819b regenerated icons, finished initialization
| * fc2a0b3 updated app id
| * fb67deb new app icon
| * 4ddb897 new app name
| * 8e414f3 changed version number
| * 04b0936 added initial screen and some ui changes
|/
* d4d3cce added settings, multi checkin, ticket overview
* 5fde2cf fixed store access, use auth from store
*   80e5bc6 Merge branch 'master' of https://github.com/Jay895/pretix-checkin
|\
| * ff382fe Delete .DS_Store
| * a913fa7 Update .gitignore
* | 3545d0f added webpack config
* | 43ed54b added scan functions, search function
* | 8f83e05 added scan plugin, API functions, store
* | 06d49eb IOS permissions
* | ccd67c5 added packages
|/
* 954e89f initial commit
```

Figure 21. Example Git log terminal output

As a reference, the terminal output in Git is shown in Figure 21. In contrast, VR-Git provides equivalent branch information, providing the labels and also using different branch colors and spatial offsetting to indicate which branch a commit relates to. To reduce visual clutter, commit messages are not shown on the planes, but rather the VR-Tablet, which includes the commit messages, timestamp, and commit ID (SHA). Note that the commit ID is displayed at the top of each commit plane to both differentiate and identify commits.

C. Commit Analysis Scenario

Git commits are a snapshot of a repository. In a typical commit analysis, a stakeholder is interested in what changed with a commit, i.e., what files were added, deleted, or modified. To readily indicate this, tiles labeled with the file pathname are placed on the commit plane to represent changed files, with colors of green representing files added, blue changed, and red for deleted. This is shown in Figure 22. In addition, the number of lines of text are shown at the bottom or a tile, with positive numbers in green indicating the number of lines added, and negative red values below it for the lines removed.

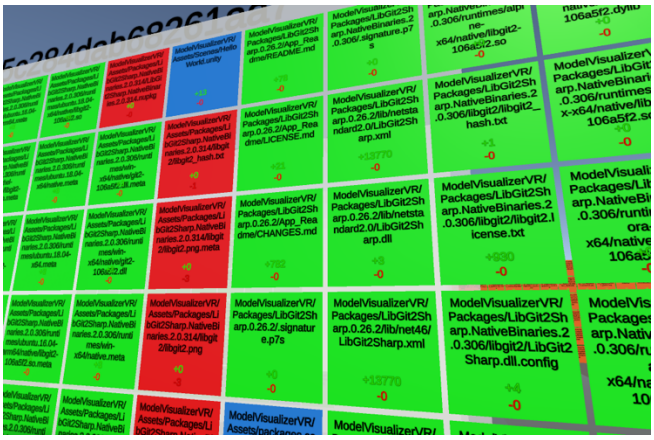


Figure 22. Commit files added (green), changed (blue), deleted (red); number of lines affected indicated in each tile at the bottom.

The ability of VR to visually scale with commits affecting a very large number of files is shown in Figure 23. As we see, there is no issue displaying the data, and VR navigation and the VR-Tablet can be used to analyze the commit further.

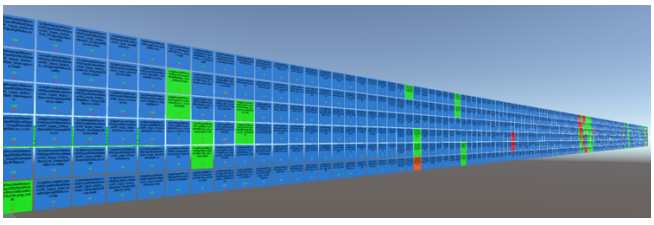


Figure 23. VR-Git commit visual scaling example for a very large file set.

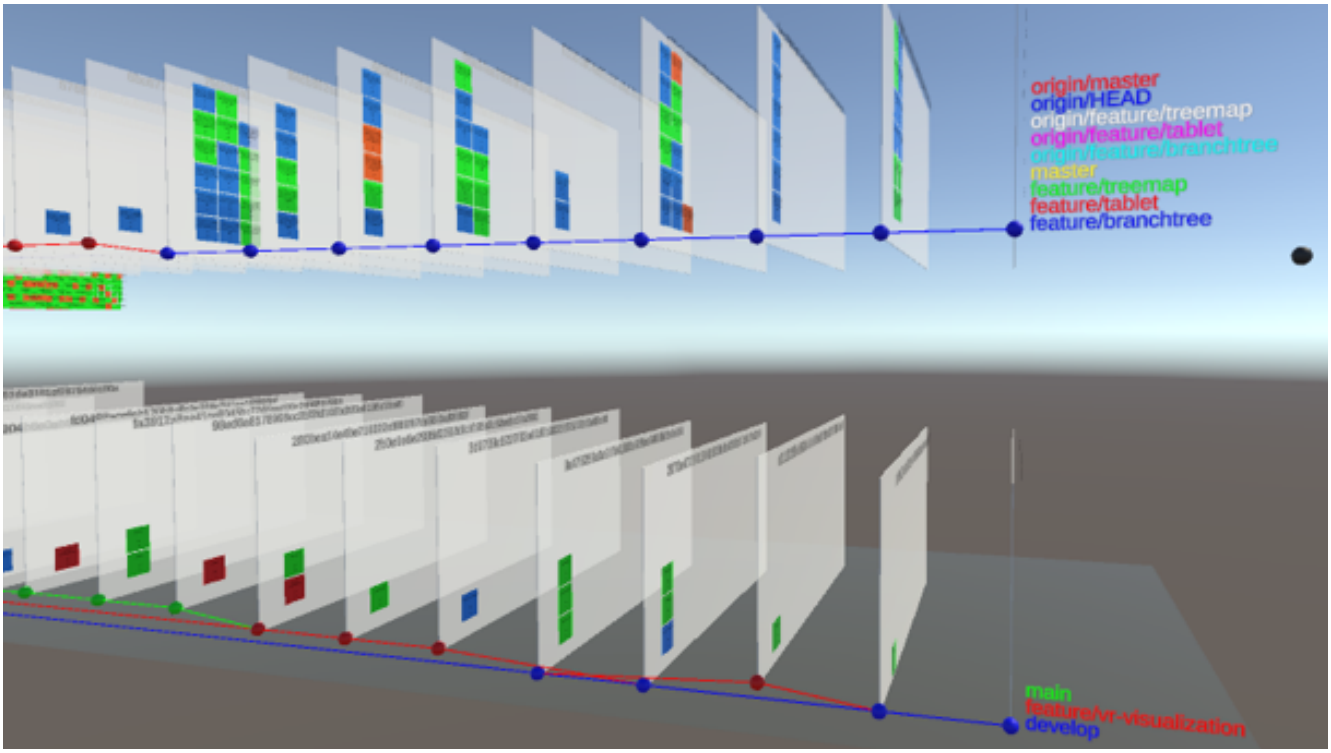


Figure 24. Dual repository comparison with a branch focus.

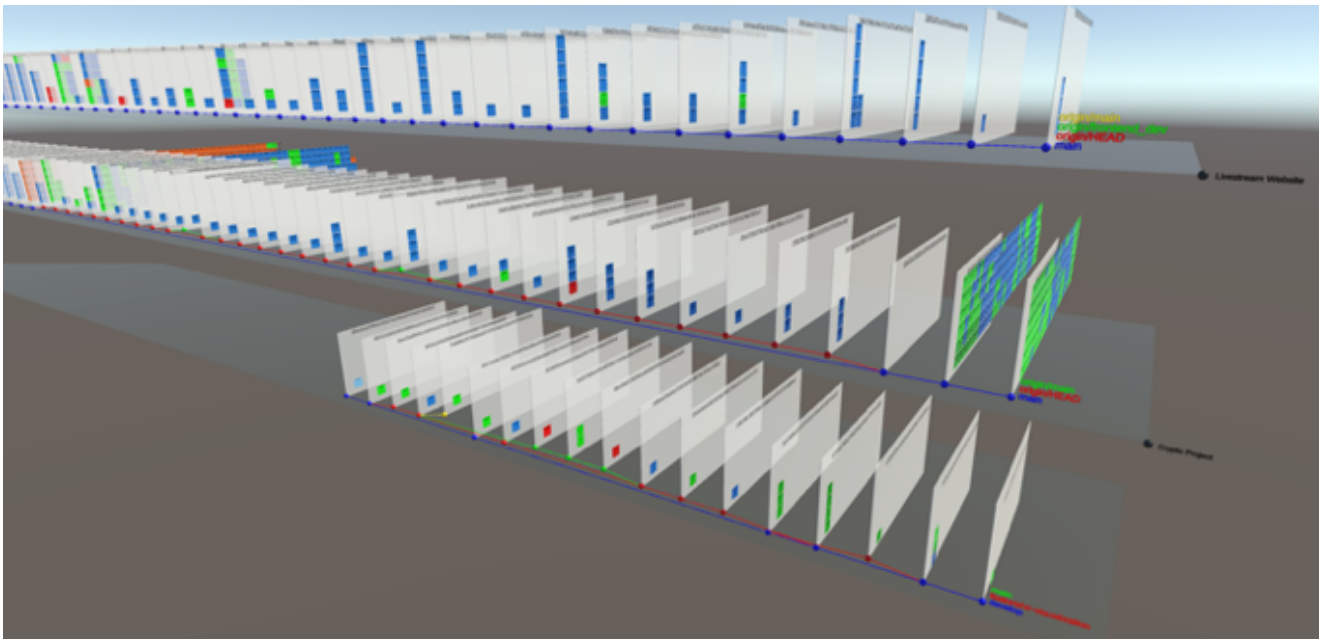


Figure 25. Multiple repositories from a wide perspective.

Commits affecting a large number of files can be readily determined, as seen in Figure 26. This can support analysis to quickly hone in on commits with the greatest impacts.

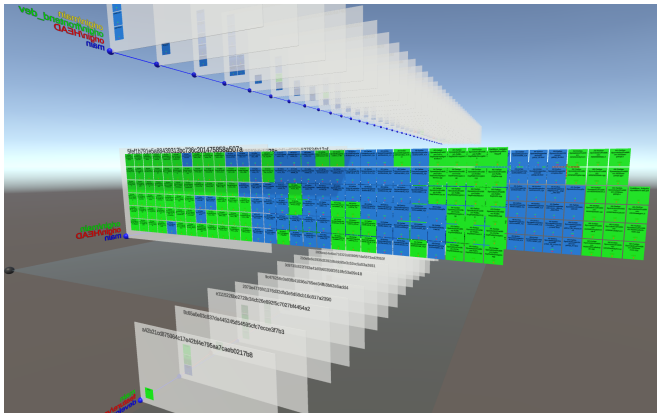


Figure 26. Multiple repositories showing commits affecting a large number of files.

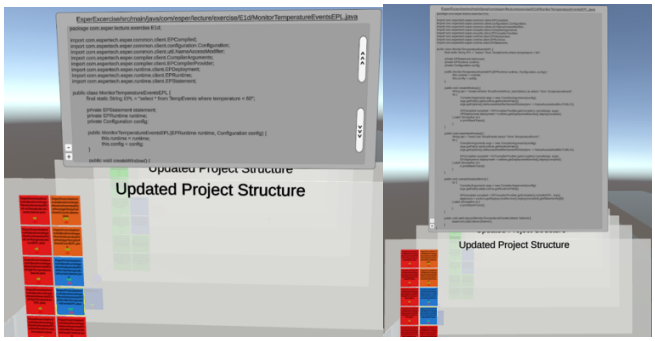


Figure 27. Code View: collapsed and scrollable (left) and expanded (right).

By selecting a specific tile (file), a file contents plane (i.e., code view) pops up displaying the contents of that file for that commit, seen in Figure 27. Since file contents can be too lengthy and wide for practical depiction in our VR-Tablet, we chose to display the plane above the commit plane, providing a clear association. The contents are initially scrollable, and can be expanded with the plus icon to show the entire file contents if desired. Since VR is not limited, one can navigate by moving the VR camera to any part of the code plane to see the code there.

D. Multi-Repository Analysis Scenario

To support multiple repository analysis, hyperplanes are used to represent each separate repository. Via the anchors, these can be placed where appropriate for the user. Branch and commit comparisons can be made from the branch perspective, with visual cues being offered by the element tiles, as shown in Figure 24. Here, one can see how the branches developed with their commits. A larger visual depiction of multiple repositories is shown in Figure 25. This shows how the VR unlimited space can be used, e.g., to determine which ones involved more commits, or where larger commits with more elements were involved via the extended commit planes.

E. Discussion

In summary, the evaluation showed that VR-GitCity supports comprehension and analysis for key Git scenarios in VR, including a repository's evolution, commits, branches, and scalable multi-repository comparisons. The city metaphor is used to convey the dynamic evolution of a repository visually and immersively, depicting the grouping and containment of elements in subfolders, the number of files, the maximum relative sizes achieved, the current size as a fill level, and colors affected elements of a specific commit. Branch comprehension and analysis were supported via the branch plane. Commit comprehension and analysis were supported via the commit planes, which readily showed the number of files involved in a commit (based on the number of tiles) and via their color if they were added, removed, or changed. The metrics in each tile show the number of lines affected. Multi-repository analysis showed the potential of VR to display and compare multiple repositories, where the limitless space can be used to readily focus and hone-in on the areas of interest or differences between repositories. This type of visual, immersive multi-repository analysis could support fork analysis, intellectual property analysis and tracking, forensic analysis, etc.

VI. Conclusion

VR-GitCity contributes an immersive software repository experience for visually depicting and navigating repositories in VR. It provides a convenient way for stakeholders who may not be developers yet have a legitimate interest in the code development to collaborate. This can further the onboarding of maintenance or quality assurance personnel. The solution concept was described and a VR prototype demonstrated its feasibility. Based on our VR hyperplane principle, repositories are enhanced with 3D depth and color. Interaction is supported via a virtual tablet and keyboard. The unlimited space in VR facilitates the depiction and visual navigation of large repositories, while relations within and between artifacts, groups, and versions can be analyzed. Furthermore, in VR additional related repositories or models can be visualized and analyzed simultaneously and benefit more complex collaboration and comprehension. The sensory immersion of VR can support task focus during comprehension and increase enjoyment, while limiting the visual distractions that typical 2D display surroundings incur. The solution concept was evaluated with our prototype using a case study based on typical Git comprehension and analysis scenarios: branch analysis, commit analysis, and multi-repository analysis. The results indicate that VR-Git can support these analysis scenarios and thus provide an immersive collaborative environment to involve and include a larger stakeholder spectrum in understanding Git repository development.

Future work includes support for directly invoking and utilizing Git within VR, including further visual constructs, integrating additional informational and tooling capabilities, and conducting a comprehensive empirical study.

ACKNOWLEDGMENT

The authors would like to thank Nikolas Lindenmeyer, Jason Farkas, and Marie Bähre for their assistance with the design, implementation, figures, and evaluation.

REFERENCES

[1] R. Oberhauser, "VR-Git: Git Repository Visualization and Immersion in Virtual Reality," The Seventeenth International Conference on Software Engineering Advances (ICSEA 2022), IARIA, 2022, pp. 9-14.

[2] GitHub repositories [Online]. Available from: https://web.archive.org/web/20220509204719/https://github.com/search 2023.12.01

[3] GitHub users [Online]. Available from: https://web.archive.org/web/20220529205506/https://github.com/search 2023.12.01

[4] C. Metz, "Google Is 2 Billion Lines of Code—And It's All in One Place," 2015. [Online]. Available from: http://www.wired.com/2015/09/ google-2-billion-lines-codeand-one-place/ 2023.12.01

[5] Evans Data Corporation. [Online]. Available from: https://evansdata.com/press/viewRelease.php?pressID=293 2023.12.01

[6] R. Oberhauser, "VR-UML: The unified modeling language in virtual reality – an immersive modeling experience," International Symposium on Business Modeling and Software Design, Springer, Cham, 2021, pp. 40-58.

[7] R. Oberhauser, "VR-SysML: SysML Model Visualization and Immersion in Virtual Reality," International Conference of Modern Systems Engineering Solutions (MODERN SYSTEMS 2022), IARIA, 2022, pp. 59-64.

[8] H. Bjørklund, "Visualisation of Git in Virtual Reality," Master's thesis, NTNU, 2017.

[9] GitHub Skyline [Online]. Available from: https://skyline.github.com 2023.12.01

[10] J. Feiner and K. Andrews, "Repovis: Visual overviews and full-text search in software repositories," In: 2018 IEEE Working Conference on Software Visualization (VISSOFT), IEEE, 2018, pp. 1-11.

[11] Y. Kim et al., "Githru: Visual analytics for understanding software development history through git metadata analysis," IEEE Transactions on Visualization and Computer Graphics, 27(2), IEEE, 2020, pp.656-666.

[12] S. Elsen, "VisGi: Visualizing git branches," In 2013 First IEEE Working Conference on Software Visualization, IEEE, 2013, pp. 1-4.

[13] A. Ciani, R. Minelli, A. Mocci, and M. Lanza, "UrbanIt: Visualizing repositories everywhere," In 2015 IEEE International Conference on Software Maintenance and Evolution (ICSME), IEEE, 2015, pp. 324-326.

[14] Rogers, B., Cunningham, S. J., & Holmes, G., "Navigating the virtual library: A 3D browsing interface for information retrieval," In: Proceedings of ANZIIS'94-Australian New Zealand Intelligent Information Systems Conference, IEEE, 1994, pp. 467-471.

[15] Chiu, P., Girgensohn, A., Lertsithichai, S., Polak, W., and Shipman, F., "MediaMetro: Browsing multimedia document collections with a 3D city metaphor," In: Proceedings of the 13th annual ACM international conference on Multimedia, 2005, pp. 213-214.

[16] R. Wettel et al., "Software systems as cities: A controlled experiment," in Proc. of the 33rd International Conference on Software Engineering, ACM, 2011, pp. 551-560.

[17] F. Fittkau, A. Krause, and W. Hasselbring, "Exploring software cities in virtual reality," Proc. IEEE 3rd Working Conference on Software Visualization (VISSOFT), IEEE Computer Society, 2015, 130-134.

[18] S. Romano, N. Capece, U. Erra, G. Scanniello, and M. Lanza, "On the use of virtual reality in software visualization: The case of the city metaphor," Information and Software Technology, 114, 2019, pp.92-106.

[19] R. Oberhauser and C. Pogolski, "VR-EA: Virtual Reality Visualization of Enterprise Architecture Models with ArchiMate and BPMN," In: Shishkov, B. (ed.) BMSD 2019. LNBIP, vol. 356, Springer, Cham, 2019, pp. 170–187.

[20] R. Oberhauser, "VR-ProcessMine: Immersive Process Mining Visualization and Analysis in Virtual Reality," The Fourteenth International Conference on Information, Process, and Knowledge Management (eKNOW 2022), IARIA, 2022, pp. 29-36.

[21] R. Oberhauser, C. Pogolski, and A. Matic, "VR-BPMN: Visualizing BPMN models in Virtual Reality," In: Shishkov, B. (ed.) BMSD 2018. LNBIP, vol. 319, Springer, Cham, 2018, pp. 83–97. https://doi.org/10.1007/978-3-319-94214-8_6

[22] R. Oberhauser, P. Sousa, and F. Michel, "VR-EAT: Visualization of Enterprise Architecture Tool Diagrams in Virtual Reality," In: Shishkov B. (eds) Business Modeling and Software Design. BMSD 2020. LNBIP, vol 391, Springer, Cham, 2020, pp. 221-239. https://doi.org/10.1007/978-3-030-52306-0_14

[23] R. Oberhauser, M. Baehre, and P. Sousa, "VR-EA+TCK: Visualizing Enterprise Architecture, Content, and Knowledge in Virtual Reality," In: Shishkov, B. (eds) Business Modeling and Software Design. BMSD 2022. Lecture Notes in Business Information Processing, vol 453, Springer, Cham, 2022, pp. 122-140. https://doi.org/10.1007/978-3-031-11510-3_8

[24] R. Müller, P. Kovacs, J. Schilbach, and D. Zeckzer, "How to master challenges in experimental evaluation of 2D versus 3D software visualizations," In: 2014 IEEE VIS International Workshop on 3Dvis (3Dvis), IEEE, 2014, pp. 33-36

[25] Libgit2Sharp. [Online]. Available from: https://github.com/libgit2/libgit2sharp 2023.12.01

[26] A.R. Hevner, S.T. March, J. Park, and S. Ram, "Design science in information systems research," MIS Quarterly, 28(1), 2004, pp. 75-105

Identification of Critical Groups and Other Supply Chain Vulnerabilities

Tim vor der Brück
Department of Computer Science
Distance University of Switzerland (FFHS)
Brig, Switzerland
email: tim.vorderbrueck@ffhs.ch

***Abstract*—The impact of a supplier or transportation link breakdown in a supply chain can strongly differ depending on which nodes/links are affected. While the breakdown of producers of rarely needed products or backup suppliers might result in no or only minor repercussions, the breakdown of central suppliers or transportation links, also called critical nodes/links, can be fatal and may cause a severe delivery delay or even a complete production failure of certain product lines. Therefore, it is of high importance for a company to identify its critical nodes/links in the supply chain and take precautionary actions such as organizing additional backup suppliers or alternative ways of transportation. In this paper, we describe a novel method to identify critical groups, nodes, and links in a supply chain based on robust optimization, which has the advantage that supply chain risks are considered, and also precise risk cost estimates regarding the possible breakdown of each supplier node are provided. Afterwards, we introduce the concept of *Critical Groups*, which is a generalization of *Critical Nodes* to potentially more than one supplier. Finally, we demonstrate this method on an example supply chain and discuss its distribution of critical nodes, links, and groups.**

***Keywords*—supply chain management; critical nodes; critical groups; critical links; robust optimization; supply chain risks.**

I. Introduction

Note that this paper is an extended version of [1]. In comparison to the original paper, we revised and extended our optimization model to include change-over costs and fixed penalties. In addition, we discuss an important generalization of *critical nodes* that we term *critical groups*.

Supply chain disruption may cause severe loss of sales and revenue. Therefore, a thorough risk analysis of the supply chain is of high importance. Falasca et al. [2] identified three major determinants of supply chain risks, which are:

- Density (cf. Figure 1)
- Complexity (cf. Figure 2)
- Critical Nodes (cf. Figure 3)

The first determinant of supply chain risks according to Falasca et al., supplier density relates to the number of suppliers residing inside a certain region. In this paper, we slightly generalize the concept of density according to Falasca and Craighead [2], [3] and also refer to a high density if a high number of suppliers reside in the same country even if these suppliers might not be geographically located close to each other. A high density increases the probability of joint supplier failures due to similar geological, economic, or political influences on neighboring suppliers. We will call a group of neighboring suppliers with a high joint dropout impact a critical group.

Craighead et al. [3] assess the complexity of a supply chain, which is the second determinant of supply chain risks, by the number of its nodes (suppliers) and edges (transportation links). The more complex the supply chain is, the higher can be the supply chain risk since a highly complex supply chain structure can complicate the logistics as well as the production processes. However, the authors point out that a high complexity can also mean that the supply chain contains redundancies and backup suppliers, which would increase its resilience. Consider as an example the supply chain in Figure 2. If supplier *A* breaks down on the low-complexity supply chain on the left subfigure, the whole chain is interrupted and non-operational because the goods on the left side of the subfigure can no longer be transported to any of the suppliers on the right side. However, on the more complex supply chain on the right subfigure, supplier *A* can partly be bypassed by nodes *B* and *C*, which effectively mitigates a potential failure of node *A*. Thus, the effect that a high complexity has on the supply chain risk is not as clear as for the other two determinants (density and critical nodes). Therefore, we chose not to propose an assessment measure for supply chain complexity and will not discuss this topic any further in the remainder of this paper.

Finally, the third determinant of supply chain risk is given by its critical nodes. Craighead et al. [3] define criticality as the relative importance of a given node or set of nodes within a supply chain (see Figure 3). A breakdown of a critical node has typically severe implications, such as serious delay or even a complete collapse of the production process for certain product lines, which can result in non-fulfillment of customer demand. Consequently, the affected company suffers lost revenue and faces a potential non-delivery contract penalty. Thus, it is of great importance to identify the critical nodes in the supply chain and mitigate their possible breakdown risks by implementing precautionary measures such as organizing backup suppliers.

The concept of critical nodes can also be transferred to important transportation links. A link in a supply chain denotes a certain transport mode (e.g., airplane, truck, or ship transportation) and a route between two suppliers or between a supplier and a customer. Analog to the definition of critical nodes, a critical link denotes a link that is of high importance for the total supply chain. Critical links should therefore be secured by identifying alternative means of transportation.

The rest of the paper is structured as follows. Related work is given in the upcoming section (Section II). The employed optimization model is given in Section III. In Section IV, we

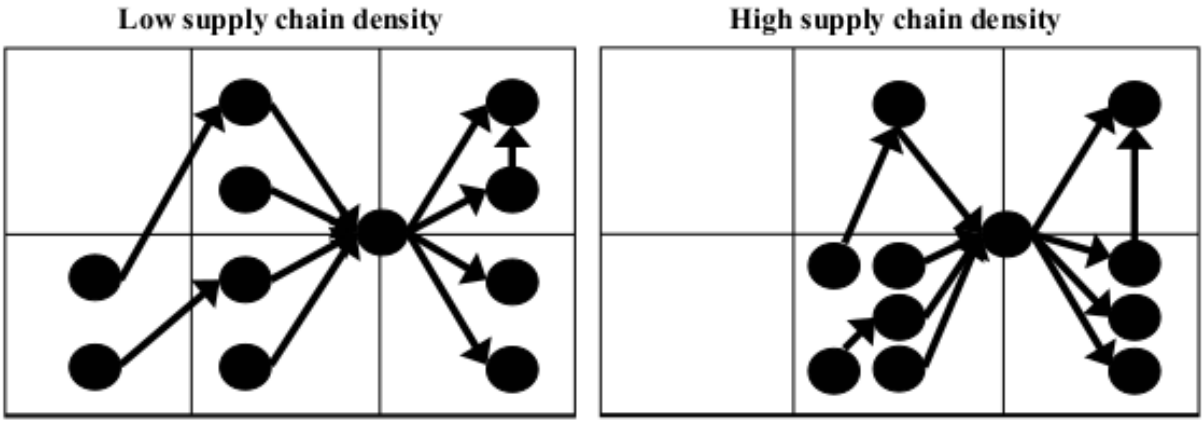


Fig. 1. Different degrees of supply chain density [2].

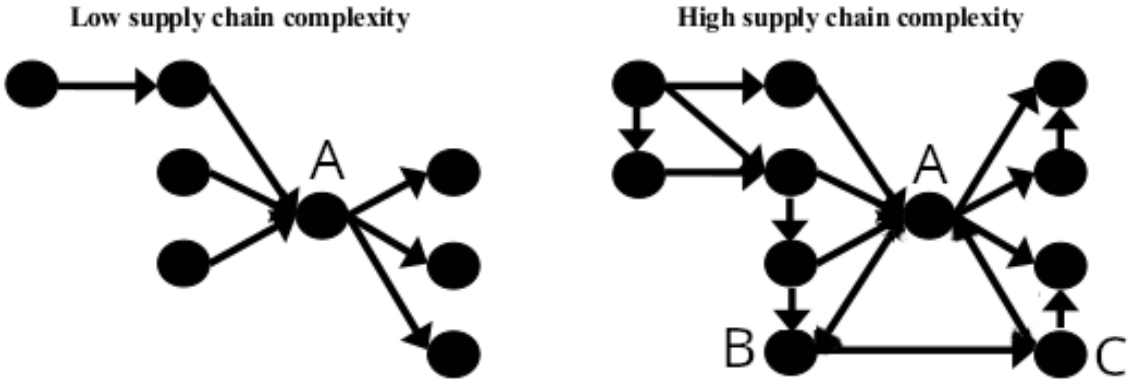


Fig. 2. Different degrees of supply chain complexity, slightly modified from [2].

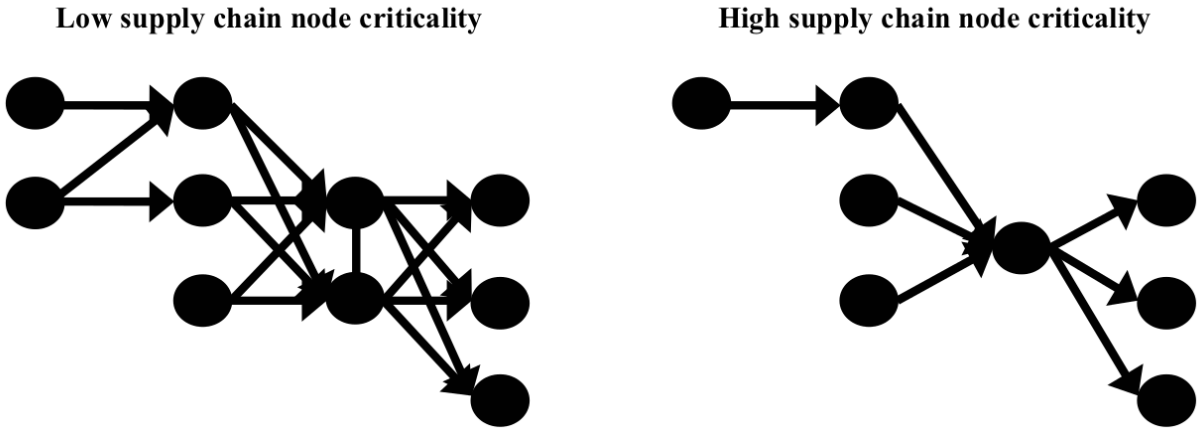


Fig. 3. Supply chain with (right) and without a critical node (left) [2].

describe how the node criticality is assessed and discuss the obtained results. Critical groups are discussed in Section V. Finally, we conclude the paper with Section VI where we summarize our contribution and give potential future work.

II. Related work

Zhang and Han [4] as well as Yan et al. [5] propose to use network centrality (especially degree, betweenness centrality, and eigenvalue centrality) as indicators for the criticality of a node in a supply chain.

Gaura et al. [6] assess the criticality of a certain network node by determining the decrease in network efficiency when this node is removed from the network. The network efficiency is measured by the normalized sum of the reciprocal of graph distances between any two nodes in the network. Prior to applying their approach, nodes with low clustering indices are removed from the network, wherefore the authors termed their approach clustering-based.

The approaches described so far assess a node's criticality alone by topological network measures. In contrast, Falasca et al. [2] propose to also consider throughput through the supply chain but fail to suggest a concrete measure. Sabouhi et al. [7] consider a node as critical, if the throughput through this node as determined by solving a linear optimization problem, exceeds a certain predefined threshold. However, this measure does not take the use of backup suppliers into account as we do here, which can de facto reduce the node criticality of alternative suppliers.

There are also some existing approaches to identify critical links. Scotta et al. [8] introduce the so-called Network Robustness Index (NRI), "for evaluating the critical importance of a given highway segment (i.e., network link) to the overall system as the change in travel-time cost associated with rerouting all traffic in the system should that segment become unusable." Note that the NRI only takes costs into account that are directly transportation-related but disregards repercussions of item non-delivery for downstream production processes as we considered in our proposed method.

We chose to use a robust optimization model as a basis for our risk cost estimation. Such a model is oftentimes employed for supply chain optimization under uncertainties. Kim et al. introduce [9] such a model, which maximizes profit in a closed-loop supply chain scenario also considering repairs and recycling of products and materials. The uncertainty arises since the available budget for uncertainties and repairs is unknown and can assume one of possible three values. Babazadeh and Jafar Razmi [10] propose a mixed integer linear programming model based on robust optimization that minimizes production, inventory, and transportation costs under 4 different economic growth scenarios.

Alternatives to the scenario-based robust/stochastic optimization approach are the use of the value at risk and the conditional value at risk. The value at risk specifies a certain quantile of the probability density function of the production loss. The closer this quantile is to the expected value of the distribution, the less is the variance of the loss and therefore also the risk. In contrast, if this quantile is located far away from the expected value, the probability density curve must be quite flat and therefore the variance and also the supply chain risk are rather high. Thus, the distance between the value at risk and expected value should be minimized for obtaining a low-risk supply chain configuration. Khorshidi and Ghezavati [11] use the value at risk approach to obtain the best possible location of facilities to minimize production loss. The downside of the value at risk-based methods is that they consider only a single location on the probability density curve, which can make this measure unreliable as risk estimate in certain situations. Therefore, nowadays, the value at risk is oftentimes replaced by the conditional value at risk that considers the weighted average of the loss beyond the chosen quantile. A supply chain optimization approach based on the conditional value at risk is introduced by Azad et al. [12]. In particular, they minimize the conditional value at risk of the lost capacity to determine the optimal amount of investment for opening and operating distribution centers. We opted for the scenario-based robust optimization approach since value at risk as well as conditional value at risk require the probability distribution of the production loss/costs, which is difficult to obtain in practice.

III. Employed optimization model

Our approach is based on robust optimization, which itself is based on stochastic optimization, which again is based on a deterministic optimization model.

We describe each of these three models subsequently in the following sections starting with the most basic one.

A. Deterministic optimization model

The deterministic model disregards any potential risk for the supply chain and determines the minimum costs of the so-called "happy flow", which denotes the best-case situation that no supply chain disruption occurs. Since such a model contains no stochastic part, it can be computed very efficiently. Note that we use, due to the computational complexity of the stochastic and robust model, a single period of 12 months for all of our 3 optimization models, over which we aggregate the total customer demand.

The following constants must be specified beforehand:

- d_{jz}: demand at location j for product z
- c_{ij}: cost to move one kg over one km from location i to j
- pc_{iz}: cost to produce one item of product z at supplier location i
- a_{xz}: number of items of product x to produce one amount of product z
- cap_{iz}: production capacity of product z at supplier location i
- in_{iz}: initial number of items of product z contained in the inventory at supplier location i
- ic_{iz}: inventory cost for storing z at location i
- $dist_{ij}$: geographical distance between locations i and j
- $weight_z$: weight of product z

- coc_i: change-over costs of supplier i. These costs arise if the supplier produces at least one item. Later on in the stochastic and robust optimization model, we assign change-over costs to all suppliers that are not part of the *Happy Flow* scenario, i.e., the scenario without any supply chain disruptions.

The following decision variables are to be determined by the optimizer:

- T_{ijz}: number of items z that are moved from location i to j
- IT_{il}: internal transfer of item l from inventory at location i
- P_{iz}: number of items z produced at supplier i
- WT_{iz}: number of items z removed from the warehouse of supplier i
- US_i: use supplier i in the supply chain

Model constraints:

- $d_{jz} \leq \sum_i T_{ijz}$: demand of item z at location j is met
- $\sum_z a_{lz} P_{iz} = IT_{il} + \sum_k T_{kil}$: number of items l required to build items z at location i
- $P_{iz} \leq cap_{iz}$: supplier at node i can at most produce cap_{iz} items for product z
- $P_{iz} + WT_{iz} \geq \sum_j T_{ijz} + IT_{iz}$: produced + removed from the inventory of supplier $i \geq$ number of items transported from supplier i
- $WT_{iz} \leq in_{iz}$ for each item z and supplier i: inventory contents cannot become negative
- $US_i = 1 \Leftrightarrow \sum_z P_{iz} > 0$: A supplier i is considered to be used in the supply chain if it produces at least one item. This constraint is implemented by means of the so-called big-M approach.

The following objective is used:
Min. $costs_{total}$ with:

$$\begin{aligned} costs_{total} := & \sum_{ijz} T_{ijz} c_{ijz} dist_{ij} weight_z \\ & + \sum_{iz} (P_{iz} pc_{iz} + in_{iz} ic_{iz}) + \sum_i US_i coc_i \end{aligned} \tag{1}$$

B. Stochastic optimization model

The stochastic model takes supply chain risks into account and computes the expected value of the supply chain costs ($\mathbb{E}(C)$) determined over all generated risk scenarios. In a stochastic optimization setting, the set of risk scenarios describes the potential hazards for the whole supply chain. Hence, the nine scenarios from our case company's supply network are used as input for the stochastic optimization approach, which are given in Table I.

The stochastic optimization model determines the minimal supply chain costs under these risks and estimates the supply network resilience of the entire supply chain. Note that certain inventory costs are currently still disregarded in our model but may be considered for future work. We have expanded our initial deterministic optimization model as follows. First, each decision variable is assigned an additional index denoting the associated risk scenario. For instance: P_{izs} denotes the number of item z produced at location i in risk scenario s. Furthermore, an additional decision variable named $Missed_{jzs}$ has been included to denote the shortfall of a produced item z at location j for risk scenario s with respect to the actual demand. To represent the effect of a missed demand, we define a variable (per item) non-delivery penalty term pen_{jz}. The penalty is invoked when the demand for item j and location z cannot be met ($Missed_{jzs} > 0$). The non-delivery penalty comprises lost revenue and a possible contract penalty. As a result, the demand constraint changes as follows: $d_{jz} \leq Missed_{jzs} + \sum_i P_{izs} T_{ijzs}$ for every scenario s and the objective function becomes:

$$\text{Min. } \mathbb{E}(C) \tag{2}$$

where

$$\begin{aligned} \mathbb{E}(C) := & \sum_s C_s p_s \\ = & \sum_{ijzs} T_{ijzs} c_{ijz} dist_{ij} weight_z p_s \\ & + \sum_{izs} (P_{izs} pc_{iz} + in_{iz} ic_{iz}) p_s \\ & + \sum_{jzs} vp_{jz} Missed_{jzs} p_s \\ & + \sum_{jzs} \mathbb{1}_{Missed_{jzs} > 0} fp_{js} p_s \\ & \quad (\text{ Missed demand is penalized.}) \\ & + \sum_{is} US_{is} coc_i p_s \end{aligned} \tag{3}$$

with p_s specifying the probability of occurrence of risk scenario s and C_s denoting the total supply chain cost in the broad sense (see Section III-C) as follows

$$\begin{aligned} C_s := & \sum_{ijz} T_{ijzs} c_{ijz} dist_{ij} weight_z \\ & + \sum_{iz} (P_{izs} pc_{iz} + in_{iz} ic_{iz}) \end{aligned} \tag{4}$$

TABLE I
SUPPLY CHAIN DISRUPTION RISK SCENARIOS FOR OUR EXAMPLE SUPPLY CHAIN.

Number	Risk Scenario
1	Product line simplification of supplier 1 - supplier no longer delivers the component due to strategy change
2	Product line simplification of supplier 2 - supplier no longer delivers the component due to strategy change
3	Covid19 pandamic
4	Cyber attack
5	Transport disruption
6	Supplier disruption due to export restrictions
7	Delivery problems of a certain part from supplier 3
8	Delivery problems of a certain part from supplier 4
9	Happy Flow - no disruptions

$$+\sum_{jz} vp_{jz} Missed_{jzs}$$
$$+\sum_{jz} \mathbb{1}_{Missed_{jzs}>0} fp_{js}$$
$$+\sum_{i} US_{is} coc_i$$

The expression $\mathbb{1}_{Missed_{jzs}>0} fp_{js}$ is modeled by means of a big-M approach. The supply chain cost estimate is given by the objective of this stochastic optimization problem formulation.

C. Robust optimization model

A supply chain disruption can cause an unmet demand, which decreases the production, transportation and inventory costs (costs in the narrow sense), since fewer items are produced, transported and stored but increases the sum of the costs in the narrow sense and non-delivery penalties comprising of lost revenues and a potential contractually agreed payment (costs in the broad sense). We differentiate between a variable per-item penalty and a fixed non-delivery penalty that is imposed as soon as a certain number (here 1) of items could not be delivered. The aggregated variable non-delivery penalties and the decrease in costs in the narrow sense are considerably correlated with each other. Therefore, unmet demand causes a non-negative costs variance for both the costs in the narrow and in the broad sense. Furthermore, a supply chain setting that minimizes the variance of the costs in the narrow sense would also have a comparatively small variance of the costs in the broad sense. A high variance of costs means a high unsureness about the actual costs and therefore a high risk. Thus, it is an important aim for a risk-averse decision maker to reduce the unsureness and therefore also the costs variance. We decided aiming to minimize the variance of the costs in the broad sense, since otherwise, the imposition of a fixed (item-independent) non-delivery penalty would not have any influence on the costs variance since the optimization objective already contains the variable non-delivery penalty.

The robust model introduces an additional constant σ that specifies the risk affinity of the decision-maker [10] [13]. Large values of σ cause a considerable increase in the costs accounting for the unsureness about the actual costs. Thus, a risk-averse decision-maker would select a rather high σ, whereas a risk-tolerant decision-maker would select a small value or drop this term altogether. Thus, the objective function changes to:

$$\text{Min. } \mathbb{E}(C) + \sigma \mathbb{V}(C) \tag{5}$$

where $\mathbb{E}(C)$ is defined in Equation 4. Since the computation of the variance requires quadratic programming, we decided to approximate it by the absolute variance [10] [14]:

$$\mathbb{V}_{abs}(C) := \sum_{s} p_s |C_s - \mathbb{E}(C)| \tag{6}$$

The absolute variance can be modeled by linear programming as follows. First, we introduce additional non-negative decision variables : $\phi(s)^+$ und $\phi(s)^-$ with the following two constraints:

$$\begin{aligned} \phi_s^+ &\geq p_s(C_s - \mathbb{E}(C)) \\ \phi_s^- &\geq p_s(\mathbb{E}(C) - C_s) \end{aligned} \tag{7}$$

The objective function is then given by:

$$\text{Min. } \mathbb{E}(C) + \sum_{s} \sigma(\phi_s^+ + \phi_s^-) \tag{8}$$

ϕ_s^+ captures the part of the variance, where the costs exceed their expected value, whereas ϕ_s^+ captures the remaining part, where the costs fall below their expected value. It can be shown that for the absolute variance, both parts must coincide. Thus:

$$\phi_s := \phi_s^+ = \phi_s^- \tag{9}$$

With this, the constraints in (7) simplify to [14]:

$$\phi_s \geq p_s(C_s - \mathbb{E}(C)) \tag{10}$$

and the objective function changes to

$$Min. \ \mathbb{E}(C) + \sum_{s} \sigma \cdot 2\phi_s \tag{11}$$

We call the objective value of this optimization problem the *risk costs* of the associated supply chain in the remainder of the paper.

IV. Assessing node criticality

Thus far, we have explained our robust optimization model, which is the basis for our proposed node criticality assessment. In particular, the robust optimization method as described above estimates the supply chain‘s risk costs that are composed of the expected total supply chain costs considering several disruption risk scenarios and their variance. A large variance implies that the supply chain costs can vary strongly depending on the occurred risk scenarios. In this case, there is high uncertainty about the incurring costs and therefore the overall supply chain risk is quite high. In contrast, low variance means that the supply chain costs do not deviate much across the scenarios. In this case, the overall supply chain risk remains small. The risk costs are leveraged in our approach for identifying the critical nodes of the supply chain.

By using risk costs instead of ordinary deterministic costs, we obtain more accurate criticality assessments of the nodes. Consider for example the case, that an important supplier S is backed up by a second supplier, which is threatened by probable bankruptcy. In a deterministic setup, the supplier S would be assigned a low criticality because of the provided backup supplier. However, in case supply chain risks are considered, the criticality of supplier S remains high due to the foreseeable default of the backup supplier.

In our approach, a supplier node is considered critical, if its complete breakdown causes a high increase in risk costs of the supply chain, which can be estimated by our robust optimization approach. In contrast, a node is considered uncritical, if the total risk costs of the supply chain do not change in case the associated supplier breaks down and can no

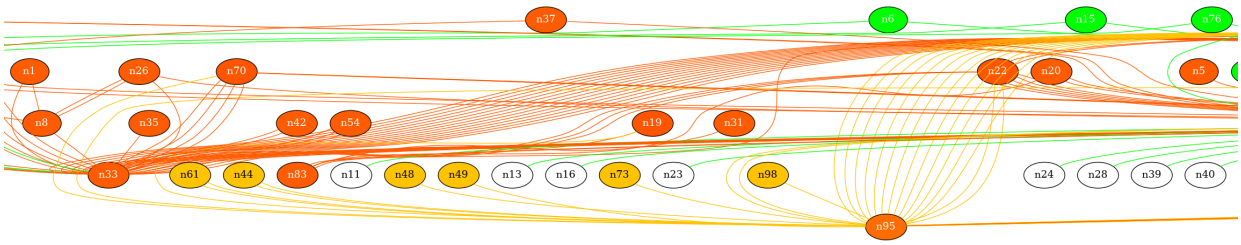


Fig. 4. Part of our example supply chain, where supplier nodes and transportation links are colored according to their criticality.

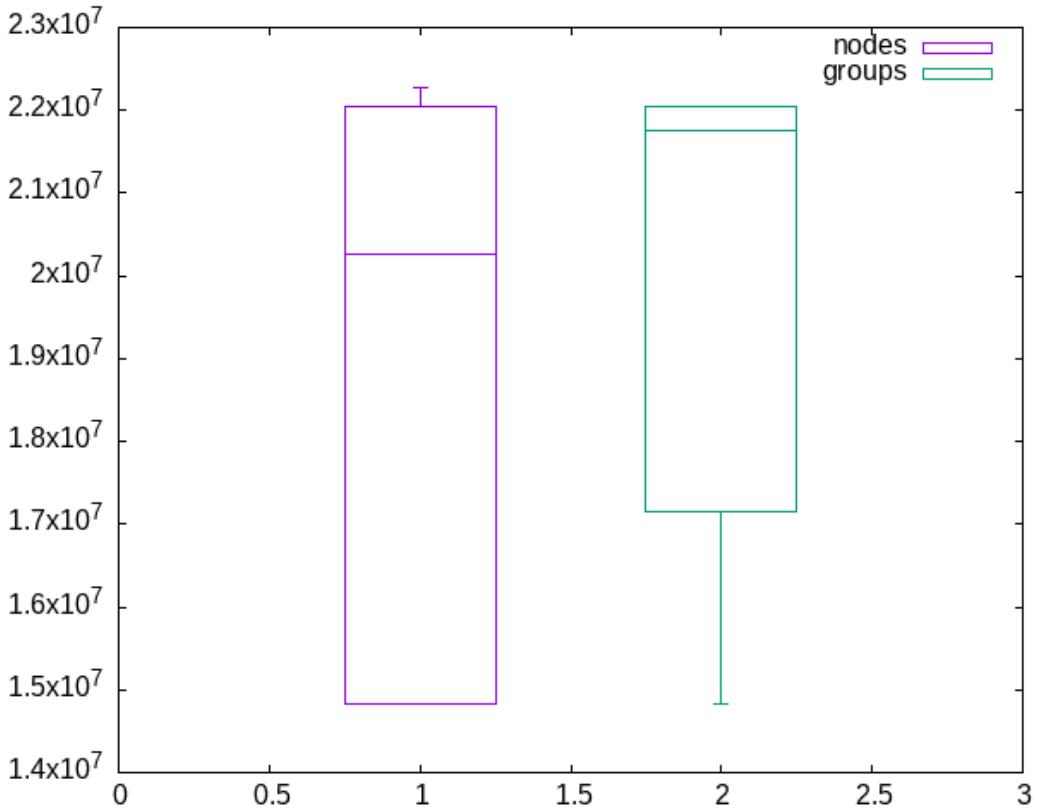


Fig. 5. Boxplot comparing the costs distribution of nodes and groups.

longer produce or deliver any goods. Therefore, we consider the criticality of a node being proportional to the overall risk costs increase of the supply chain when the node in question is removed. We will call in the remainder of the paper the risk costs of the supply chain, in which a certain supplier node n is removed, the risk costs of this node n.

A node in the supply chain network can represent either a supplier or a customer, while the edges represent transportation links either between two suppliers or between a supplier and a customer. We consider in the following an example supply chain with 40 customers, 80 suppliers, 200 components and products, 200 transportation links, and 400 product demands. Due to its size, we only depict a part of the total supply chain in Figure 4, which has similar characteristics in terms of critical links and nodes as the total supply chain.

Each supplier node in this network is colorized according to its criticality. Suppliers are colored green if the risk costs of the supply chain are not increased by its potential breakdown, they are colored yellow if the supply chain risk costs are increased by a certain threshold factor f_1 (we use 30%), and red if the costs were increased by a second larger threshold factor f_2 (we use 60%) or more. Note that the exact values of factors f_1 and f_2 can vary depending on the corporate branch and the degree of competition. For costs increases between 0 and f_1, we interpolate the RGB color values linearly between green (red=0, green=255, blue=0) and yellow (red=255, green=255, blue=0), for costs increased between f_1 and f_2, we interpolate the color values between yellow and red (red=255, green=0, blue=0). Customers are not associated with any production risks and therefore their associated graph nodes are not colored and instead visualized by unfilled circles. The entire process is illustrated in the form of pseudocode in Figure 6.

Like critical nodes, we also visualize critical links in the supply chain. Analog to the node case, they are colored in green if uncritical, in yellow if somewhat critical, and in red if critical. Again, mixtures of the colors red and yellow as well as green and red are possible. In case there are several transportation modes available between two connected nodes, we consider only the most critical mode for the visualization. Note that a link originating from an uncritical supplier node must also be uncritical. However, the opposite does not hold. A link originating from a critical supplier node, can be considered uncritical, if alternative (backup) transportation modes are available.

The most critical node in our example supply chain would increase the risk costs by 50% in case of failure. Furthermore, by far the largest part of the suppliers is considered rather critical by our chosen definition of f_2, which is caused by the fact that backup suppliers are missing in most cases. The remaining suppliers are to the same part either non-critical (visualized in green) or somewhat critical (visualized in yellow). In contrast, the distribution of links is much more balanced. Almost 56% of the links are regarded as critical, the rest is either somewhat critical or uncritical. In particular, transportation links leading to a customer are all considered uncritical due to existing alternative transportation modes, while most of the inter-supplier links are critical. Optimally, the decision-maker should supply backup suppliers/transportation modes for all critical nodes and links so that all critical nodes / links become somewhat critical or uncritical.

V. Identification of Critical Groups

A high supply chain density is not critical per se but only if all suppliers located in a close proximity have a common risk trigger like a natural disaster (see Figure 7) or certain political or economic circumstances (cf. [15] for an overview of major supply chain disruption risks). We call a group of such suppliers critical if their common failure would have a strong impact on the total supply chain costs. A critical group is in principle an extension of the concept of a critical node. Members of the same critical group are oftentimes located in a geographical neighborhood, although this is not a strict requirement. The criticality of a group is determined analogous to the criticality for nodes or link as presented earlier by risk costs. In particular, the risk costs of a group are given by the risk costs of the supply chain (including lost revenue and potential contract penalties due to missed demand) in which all the individual group members are blocked and are not able to produce (and potentially also deliver) any goods. A group is considered critical if the failure of the entire group considerably increases the supply chain costs so that they exceed a certain predefined threshold value.

The identification of critical groups gives the decision maker another criteria to identify potential weaknesses and bottlenecks in the supply chain. After their identification, s(he) has the following options to mitigate potential repercussions of a complete group failure.

1: **procedure** GET_RISK_COSTS_COLOR($nodes$,$costs_{hf}$)
2: Input $nodes$: list of total supply chain nodes
3: Input $costs_{hf}$: "happy flow" costs
4: $red := (255, 0, 0)$
5: $green := (0, 255, 0)$
6: $yellow := (255, 255, 0)$
7: $hm := \{\}$ # associated risk costs of a node
8: $hm\_color := \{\}$ # associated RGB values for a node
9: **for** $n \in nodes$ **do**
10: **if** type(n)==Supplier **then**
11: $costs := obj\_value(min_{costs}(nodes \backslash \{n\}))$
12: $hm[n] := costs$
13: **if** $costs < (1 + f_1)costs_{hf}$ **then**
14: $w := (costs - costs_{hf})/(f_1 \cdot costs_{hf})$
15: $hm\_color[n] := w \cdot yellow + (1-w) \cdot green$
16: **else if** $costs < (1 + f_2)costs_{hf}$ **then**
17: $df := f_2 - f_1$
18: $dcosts := costs - (1 + f_1)costs_{hf}$
19: $w := dcosts/(df \cdot costs_{hf})$
20: $hm\_color[n] := w \cdot red + (1 - w) \cdot yellow$
21: **else** $hm\_color[n] := red$
22: **end if**
23: **end if**
24: **end for**
25: return hm, hm_color
26: **end procedure**

Fig. 6. Identification of risk costs and node criticality for all suppliers.

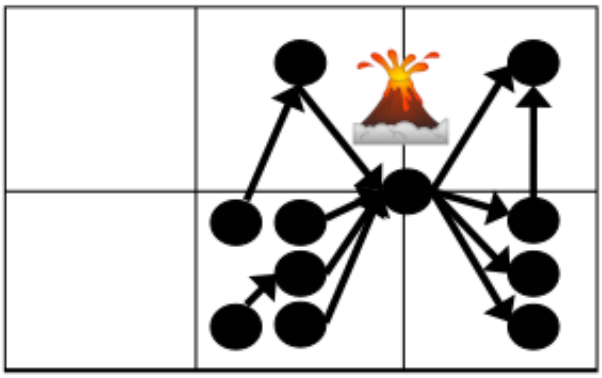


Fig. 7. Supplier group with a volcano as a common risk trigger.

- (S)he can reduce the impact of an occurred risk, usually by providing a backup supplier for each member of a critical group.
- (S)he can reduce the probability that the risk occurs. Note that this option might not always be available, since the decision maker can oftentimes not directly influence the underlying cause of the risk.
- (S)he can resolve the risk scenario altogether by replacing all group members with equivalent suppliers that are not affected by the risk in question. For instance, if a certain set of suppliers is located in the immediate neighborhood

TABLE II
RISK SCENARIO AND ASSOCIATED SUPPLIER GROUPS.

Risk Scenario	Potential group members
Labour Strike	suppliers in the same country and of identical corporate branch
Natural Disaster	suppliers in the vicinity of where the natural disaster is expected to occur
Political Instability (e.g., danger of coups) in a certain country	all suppliers in this country
Vulnerability to cyber attacks	suppliers with a strong dependencies on IT infrastructure that is accessible from outside, e.g., companies conducting e-commerce and selling their goods and services directly to the end-cunsumer over the internet.

of an active volcano, the decision maker can replace them with suppliers in a different geographical area.

Note that different types of the same risk (e.g., political instabilities of two different countries) should be modeled as separate scenarios, since they usually have different repercussions and occurrence probabilities.

In Table II we give some examples of risk scenarios and associated potential supplier groups. The groups and supplier we used in our evaluation are specified in Table I. Note that all risk scenarios, affected supplier groups, and risk occurrence probability are currently manually specified. In principle, the affected supplier groups could be at least partially determined automatically by statistical analysis if enough background data is available.

The 8 groups in our supply chain are displayed in Figure 8. The nodes are colored from light to dark according to their criticality ranking ranging from 1 (most uncritical) to 8 (most critical) whereas darker color means a higher criticality. In case, a supplier belongs to several groups, the supplier is drawn in the color of its most critical group (associated colors: yellow, light orange, dark orange, green, blue, purple, brown, black). In this figure, the coloring only depends on the ranking from 1 to 8, while the costs are not directly reflected. In addition, all groups with a criticality level nearer to *critical* than to *somewhat critical* are displayed in an increased size. The figure shows that most groups in our example supply chain are critical, that the group size is rather moderate ranging from 1 to 4 suppliers and that several times a supplier belongs to different groups, thus the most uncritical group colored in yellow does not show up at all in the graph. In addition, it can be perceived that suppliers belonging to the same group show up in neighboring locations of the supply chain.

We also compared the risk costs distribution of critical nodes and critical groups. As can be seen in Figure 5, the risk costs of the groups usually exceed the ones of the nodes but not by a high margin. This might seem slightly surprising at first sight, since a group usually contains of several suppliers, thus a group failure should normally have a higher impact than a failure of a single supplier. Furthermore, the interquartile range (IQR) of the risk costs is considerably lower for the groups than for the nodes, i.e., for the former, the risk costs are more concentrated around the median of the distribution. This effect is mainly caused by the fact that there are some uncritical nodes, whose failure do not cause a considerable risk cost increase, while a group failure has almost always a large impact on the supply chain.

However, only our example scenarios 3 and 4 involve the failure of several suppliers and these suppliers are highly dependent of each other since a supplier as well as its direct upstream suppliers are affected by the risk scenario.

VI. CONCLUSION

We described a method for identifying critical groups, nodes, and groups in a supply chain based on robust optimization. In contrast to other state-of-the-art methods, our method is very precise since it not only considers network topology but also network throughput as well as possible supply chain disruption risks. Furthermore, our method provides a concrete risk costs estimate for the breakdown of each supplier, group, and transportation link. In addition, we provided a representation as a risk graph that allows for easily pinpointing the supply chain vulnerabilities by a decision maker.

Furthermore, we applied our method on a real-world supply chain to analyze its vulnerbilities. The analysis revealed that most of the supplier nodes were considered critical with our employed threshold. Moreover, it could be shown that in average, the groups were only slightly more critical than node in terms of cost increase, which is caused by our specific risk scenario setup.

Currently, risk scenarios, supplier groups, and risk occurrence probabilities are all specified manually. A potential future work is to obtain the supplier groups affected by a certain risk by using statistical analysis of available background data.

ACKNOWLEDGMENT

Hereby we thank all who supported us in this work, especially Gunter Krell for a lot of fruitful discussions and Uta Jüttner who provided us with fitting risk scenarios. Furthermore, we want to thank the Swiss Innovation Agency Innosuisse for funding the work leading to this publication.

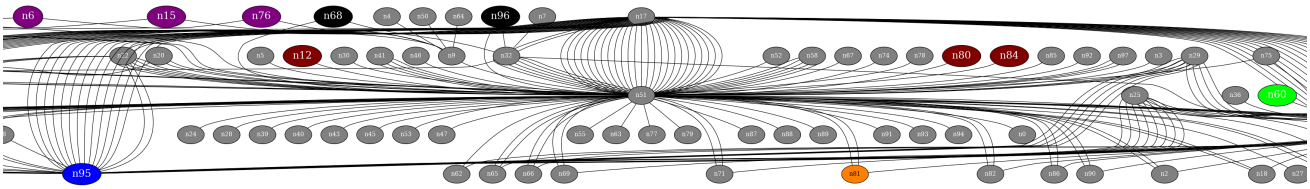


Fig. 8. Identified critical groups in our supply chain (selected part of the total supply chain contains all group nodes).

REFERENCES

[1] T. vor der Brück and R. Meier, "Identification of critical nodes and links in a supply chain by robust optimization," in *Sixteenth International Conference on Advanced Engineering Computing and Applications in Sciences (ADVCOMP)*, Valencia, Spain, 2023.

[2] M. Falasca and C. W. Zobel, "A decision support framework to assess supply chain resilience," in *Proceedings of the 5th International ISCRAM Conference*, 2008, pp. 596–605.

[3] C. W. Craighead, J. Blackhurst, M. J. Runtusanatham, and R. B. Handfield, "The severity of supply chain disruptions: Design characteristics and mitigation capabilities," *Decision Sciences*, vol. 38, no. 1, pp. 131–156, 2007.

[4] X. Zhang and J. Han, "Analysis of the importance of nodes in supply chain network," in *Proceedings of the 2011 International Conference on Business Computing and Global informatization*, Shanghai, China, 2011, pp. 387–388.

[5] T. Yan, T. Y. Choi, Y. Kim, and Y. Yang, "A theory of the nexus supplier: A critical supplier from a network perspective," *Journal of Supply Chain Management*, vol. 51, no. 1, 2015.

[6] V. Gaura, O. Yadavb, G. Sonia, and A. Rathorea, "Identification of critical nodes and edges in a network based on clustering," in *Proceedings of the 30th International Conference on Flexible Automation and Intelligent Manufacturing (FAIM2021)*, Athens, Greece, 2021, pp. 1298–1304.

[7] F. Sabouhi and M. S. Jabalameli, "A stochastic bi-objective multi-product programming model to supply chain network design under disruption risks," *Journal of Industrial and Systems Engineering*, vol. 12, no. 3, pp. 196–209, 2019.

[8] D. M.Scotta, D. C. Nova, L. Aultman-Hall, and F. Guo, "Network robustness index: A new method for identifying critical links and evaluating the performance of transportation networks," *Journal of Transportation Geography*, vol. 14, pp. 215–227, 2006.

[9] J. Kim, B. D. Chung, Y. Kang, and B. Jeong, "Robust optimization model for closed-loop supply chain planning under reverse logistics flow and demand uncertainty," *Journal of Cleaner Production*, vol. 196, pp. 1314–1328, 2018.

[10] R. Babazadeh and J. Razmi, "A robust stochastic programming approach for agile and responsive logistics under operational and disruption risks," *International Journal of Logistics Systems and Management*, vol. 13, no. 4, pp. 458–482, 2012.

[11] E. Khorshidi and V. R. Ghezavati, "Application of mathematical modeling value-at-risk (var) to optimize decision making in distribution networks," *SN Applied Sciences*, vol. 12, no. 1, 2019.

[12] N. Azad, H. Davoudpour, G. K. D. Saharidis, and M. Shiripour, "A new model to mitigating random disruption risks of facility and transportation in supply chain network design," *The International Journal of Advanced Manufacturing Technology*, vol. 70, no. 9-12, pp. 1757–1774, 2014.

[13] J. M. Mulvey, R. J. Vanderbei, and S. A. Zenios, *Operations Research*, vol. 43, no. 2, pp. 264–281,

[14] C.-S. Yu and H.-L. Li, "A robust optimization model for stochastic logistic problems," *International Journal of Production Ecnonomics*, vol. 64, pp. 385–397, 2000.

[15] S. DuHadway, S. Carnovale, and B. Hazen, "Understanding risk management for intentional supply chain disruptions: Risk detection, risk mitigation, and risk recovery," *Annuals of Operation Research*, vol. 283, pp. 179–198, 2019.

An IoT Stereo Image Sensor System for Agricultural Application

Bruno M. Moreno[1,2], Paulo E. Cruvinel[1,2]

[1] Embrapa Instrumentation (CNPDIA), P.O. Box 741, 13560-970, São Carlos, SP, Brazil
[2] Postgraduate Program in Computer Science, Federal University of São Carlos, São Carlos, SP, Brazil
Emails: bruno.moreno@estudante.ufscar.br; paulo.cruvinel@embrapa.br

***Abstract*—Sensors and Internet of Things systems have become quite important to support decision-making in agriculture. In such a context, smart farming has emerged as a new opportunity for food production based on a sustainable development concept, since the rational use of agricultural inputs is now a reality. One of these opportunities is the application of precision agriculture for weed control. This paper presents the characterization of an embedded stereo system using camera sensors, Internet of Things principles for computational intelligence tasks. For validation, it has been used the Modular Transfer Function concept, that is, taking into account not only the calibration of the sensors, but also of the 3D system, memory use and energy consumption for a long term operation. Furthermore, the results clarify details related to the implementation and construction of such a 3D system, which in fact aims to control invasive plants in agricultural crops.**

***Keywords*—camera sensor; stereo vision; embedded platform; IoT sensors; agricultural industry.**

I. Introduction

Agriculture is a very important source of food, feed, fiber and even fuel. Despite this, agriculture currently faces the challenge of increasing its production in response to the demand of continued population growth, taking precautions against the various adversities caused by the climate and minimizing the impact of man on nature.

Recently, a previous study regarding an Internet of Things (IoT) system for agricultural application was presented at the Eighth International Conference on Advances in Sensors, Actuators, Metering and Sensing (ALLSENSORS 2023) [1].

IoT devices have been used in agriculture, mainly in tasks that aim to reduce waste of resources. As examples of applications, IoT can help with the storage of agricultural products, smart irrigation, soil monitoring, nutrient management, precision agriculture, intelligent livestock management and crop monitoring. In irrigation, devices can automate the process intelligently, collecting data from the soil with temperature and humidity sensors, and using the collected and historical information to train a model to decide the best time to activate irrigators. Other information can be collected from the soil by sensors, such as pH and nutrient content, allowing the choice of the best plant breed for certain soil parameters. This information can be controlled and monitored remotely via web or mobile applications. The sensors can also track the farm and with the data collected, farmers can plan their farming activities such as seed selection, sowing, amount of fertilizer used, harvest date and expected yield amount [2].

One of the approaches aimed at increasing productivity in the field is the reduction of losses due to factors exogenous to crops, such as competition resulting from the presence of invasive plants. The presence of weeds in the cultivation area can decrease crop yield by more than 50% just by competing with the moisture present in the soil, causing more damage than invasive animals, diseases and other pests [3]. Therefore, weed control is essential so that the nutrients present in the soil, the development space and the reception of sunlight remain exclusively for the plant of interest [4].

Moreno and Cruvinel presented previous studies related to a stereo camera's system [5], and the development of a software based on semantic computing concepts for the segmentation of weed plants [6]. Even though there are systems that perform plant phenotyping [7], none have combined the information generated by stereo images, as the system developed can also provide the height of the plants as data to assist in the task of deciding the correct quantity of product in the region. Of the works that use more than one camera to obtain images, there is a greater focus on 3D reconstruction of plants, which allows generating a point cloud representation of the plant from a depth map [8].

Although the use of pesticides has already been established to deal with this problem, technological applications aimed at the rational use of inputs are desired. Among such technologies, Computer Vision stands out, which works in two stages: image acquisition and image processing. The acquisition is made exclusively from camera sensors, capturing the environment and patterns present in digital images. Such sensors can then capture the visible or thermal spectrum, and be coupled to vehicles, devices, robots, drones and even satellites. On the other hand, affordable single-board computers have made onboard image processing possible [9].

Image processing, a task of computacional intelligence, can be summarized in five steps. In the first, the raw data are preprocessed, removing noise and selecting only the object of interest. In another step, pattern features are extracted, whereas in the case of plant images, such parameters are related to color, shape and texture. In the third stage, the features go through a selection process, decreasing the dimensionality of the data. Afterwards, the data are classified, grouping them based on their similarities. Finally, in the decision-making stage, new input data can be classified from the already trained model, thus identifying which group it belongs to [10][11].

To ensure that the input data are of good quality, validating and using good camera sensors have become extremely impor-

tant. Allied to this, other points of consideration in the application of such techniques in agriculture are the management of the volume of data generated, the data analysis techniques that need to deliver interpretable and understandable results due to the interdisciplinarity of workers in the area, and the mobile systems that need to be able of handle scarce resources such as limited battery life, low computational power and limited bandwidths for data transfer [12].

As examples of the use of camera sensors in the field, there are applications coupled to vehicles to operate during preplanting and analyze the height and density of vegetation from the images [13] and identify the location of invasive plants for manual control via weeding machine [14]. Plant images can also be acquired to create a database on an external server for further processing, for training a future classifier [15].

This paper is structured as follows. Section II presents the materials and methods used, including the IoT system description, camera sensor specifications, stereo vision basics, and embedded board specifications. Section III presents the results of the validation of the sensor and of the stereo system, the power supply and memory limitations and the final prototype, with the final conclusions in Section IV.

II. Materials and Methods

The developed system aims to capture stereoscopic images in a real environment of plantations, so that the presence and concentration of weeds present in the region of interest can be identified from an embedded algorithm. The capture of stereo images requires two camera sensors, generating two images of the same area that will be the input of the system. The images are then processed and grouped into classes, and the data will be prepared for sending to a module external to the system, which will be responsible for spraying the site.

A. High-level IoT architecture

Embedded systems have a potential in agricultural use due to their mobility, low cost and computational power, allowing the performance of complex tasks in a more practical way. Raspberry Pi (RPi) is being used in several applications and it is the leading candidate for hardware implementation due to its powerful processor, rich I/O interface and compatibility that allows most projects to run on it [16]. Its wireless communication also makes the RPi capable of working with IoT projects, allowing objects to be sensed or controlled remotely across existing network infrastructure and reducing human intervention [17].

IoT systems in agriculture are separated into three modules: farm side, server side and client side. The farm side usually consists of detecting local agricultural parameters, identifying the location and sensor data, transferring crop fields data for decision-making, decision support and early risk analysis based on recent data, and action and control based on the monitoring of the crop [18]. As can be seen from the block diagram in Fig. 1, the farm side is represented by the developed IoT Stereo System that can gather image data in the field, pre-process, segment, create feature extraction and depth information vector, classify and interpret the collected data, while being controlled and monitored via Bluetooth serial communication by a mobile app.

On the server side, the network layer is responsible for reliable transformation to the application layer. A Wireless Personal Area Networks (WPAN) network can be mounted on a single board computer, with its own unified control and monitoring console for various wireless networks. Data transport and storage become essential, with data that can be saved on an external server or in the cloud, and then transferred to other devices, including the equipment responsible for product spraying on the plantation. The last module, the client side or application layer, collects and processes information, providing an environment where users can monitor data processed by the system via a web browser, anywhere and anytime. In Fig. 1, the server side is represented by the Bluetooth and Wi-Fi communication of the system to the farm server and by the server management of local and remote network, while the client side is represented by the mobiles devices and by the cloud environment.

Communication between all devices can be carried out via the Bluetooth protocol, which supports up to 7 devices connected simultaneously. The Bluetooth 4.2 connection can reach the transfer limit of 1 Mbps and the signal can reach 10 m away from the board indoors and 50 m outdoors. One of the protocols used is radio frequency communication (RFCOMM). The RFCOMM protocol is an important layer that provides a serial interface to the Bluetooth transport layer, emulating an RS-232 interconnect cable. RFCOMM is based on the ETSI 07.10 standard, which allows the emulation and multiplexing of multiple serial ports on a single transport [19]. The OBEX protocol (OBject EXchange) is also used for file transfer, which is a software implementation of the File Transfer Protocol (FTP) network protocol, which runs on top of RFCOMM.

To ensure system security, it only connects to trusted equipment and specific ports. An RPi is then defined as master, responsible for receiving commands sent by an application on an Android cell phone and using this command to carry out its actions and inform the other board what it should also do. The other RPi is defined as a slave, receiving commands from the master and obeying them.

The pseudocode of the algorithm developed for the system communication between all components is described below, where $addr\_master$, $addr\_slave$ and $addr\_mob$ are the MAC addresses of the master RPi, slave RPi and mobile controller, respectively, and prt_1 and prt_2 are the ports enabled for serial communication:

---

function MASTER COMMUNICATION($addr\_slave$, $addr\_mob$, prt_1, prt_2)
begin function
$s1$ = create_socket_bluetooth(RFCOMM)
$s2$ = create_socket_bluetooth(OBEXFTP)

```
    connect(s1,(addr_slave, prt_1))
    bind(s2,(addr_mob, prt_2))
    while mobile connection is not interrupted do
        comd = get_data(mobile)
        send(comd,slave)
    end while
    return comd
end function
function SLAVE COMMUNICATION(addr_master, prt1)
begin function
    s = create_socket_bluetooth(RFCOMM)
    bind(s,(addr_master, prt_1))
    accept_conection(s)
    while master connection is not interrupted do
        comd = get_data(master)
    end while
    return comd
end function
```

The system is built in such a way that it can be operated in the field without the need for an Internet or 5G connection, which allows the data collection stage to work in more isolated locations, requiring only the existence of a local network. The processed data can be used by other devices connected to the local network, but can also be transmitted to external servers later when the connection to the World Wide Web is available. In this way, specific commands can be sent remotely by the user to the system, which will perform procedures such as image capture and data transfer.

B. Embedded System and Camera Sensor Specifications

RPi is a series of mini-embedded computers developed in the United Kingdom by the Raspberry Pi Foundation in association with Broadcom. The model used was the RPi 3 B+, where its specifications can be seen in Table I. It is important to note that the board must be powered with a nominal voltage of 5 V capable of delivering 2.5 A of current, with operating temperature between 0 °C and 50 °C.

The internal memory is defined from a micro Secure Digital (SD) card, where the kernel of the operating system is also present, being recommended the use of at least 8 GB of memory. The RPi 3 B+, unlike previous family models, enables BCM43438 wireless Local Area Network (LAN) and Bluetooth Low Energy (BLE) communication, allowing wireless data exchange.

The RPi has its own camera sensor alternatives, including the Pi Camera v1, with specs shown in Table II. Among the most important parameters, stand out the fixed focal length of 3.60 mm, the maximum sensor resolution of 2592 x 1944 pixels, and the camera opening angle of 53.50º horizontally and 41.41º vertically.

C. Modular Transfer Function as Camera Sensor Validation

Lens and camera designers face challenges in developing systems with high image quality. The problem of greatest concern is how to optimize lens parameters such as curvatures and thicknesses to obtain high image resolution. A set of optimizations were proposed to improve the aberrations of lens systems, using as a metric the Modular Transfer Function (MTF), which is the amplitude term of the Optical Transfer Function (OTF) which is similar to the transfer function of the linear system [20]–[22].

To evaluate the quality of the images acquired by the cameras, the MTF is used from each one of them and from the set, expressing how well an optical system preserves the contrast of spatial frequencies of the object in the image and is a well-established performance method [23].

A simple method to obtain the transfer function is to generate the system response when the input is a pure impulse signal, therefore obtaining the impulse response of the function. Using the same procedure, a point source is considered as the impulse signal to help estimate the image response in a lens system.

The point source image shown on the image plane is called the Point Spread Function (PSF), which is the inverse Fourier transform of the OTF. The projection of the PSF in 1D is called the Line Spread Fuction (LSF), measurement preferable because it can be obtained simply and equally valid for cases where there are no distortions between the axes.

Then, the camera sensor can be defined taking into account the calculation of the LSF of the camera lens and the MTF, which represents the magnitude response of the optical system to sinusoids of different spatial frequencies, that is, recovered by the Fourier transform of the LSF. Several key aspects of optical instrumentation relate to the implementation of a linear source for a given optical system, the impact of finite source size on measurement, and the choice of optical elements for imaging the response of specific patterns and their relationship to the lens used in the camera sensor.

Taking a linear source, the solution to measure the MTF is in 1D orthogonal to the direction of the line. This can be proven considering a given source $S(x, y) = \delta(x)C$ and a lens with a diameter equal to a, obtaining the objective response $R(k_x, k_y)$, described in Equation (1).

$$R(k_x, k_y) = \int \int_{-a/2}^{a/2} \delta(x) C e^{j(k_x x + k_y y)} dx dy \qquad (1)$$

Thus, the spatial frequencies associated with the spatial coordinator (x, y) can be expressed as the square of the Fourier transform of the product of the source and lens aperture $R^2(k_x, k_y)$, with (k_x, k_y). Therefore, looking for the solution of (1) and solving the integral by parts, it is possible to arrive at:

$$R^2(k_x, k_y) \alpha \frac{sin^2(ak_y)}{(ak_y)^2} \qquad (2)$$

Equation (2) corresponds to the LSF. The Fourier Transform of the LSF then gives the 1D MTF in the yy direction. Considering that the lens has circular symmetry, using this function it is now possible to characterize the entire lens.

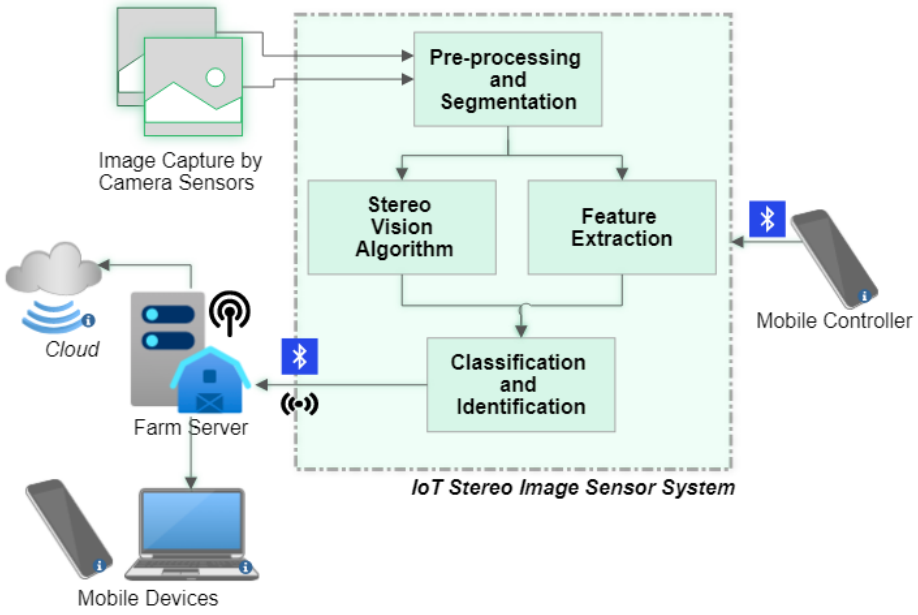


Figure 1. High-level system architecture diagram.

TABLE I
RASPBERRY PI 3 MODEL B+ CHARACTERISTICS

Processor	BCM2837B0 Cortex-A53 (ARMv8) 64-bit		
Clock	1.4 GHz	GPIO	40 pins
Memory	1 GB SDRAM	Gigabit Ethernet	1 connector
USB Port	4 USB 2.0	HDMI	1 connector
Camera serial interface (CSI)		Display serial interface (DSI)	
Wireless (dual band)		Bluetooth 4.2/BLE	
3.5mm 4 Jack output		Micro SD card slot	
Support Power-over-Ethernet		Input DC 5V/2.5A	

TABLE II
PI CAMERA CHARACTERISTICS

Size	25 x 24 x 9 mm
Resolution	5 MP
Video modules	1080p30, 720p60, 640x480p60/90
Sensor	OmniVision OV5647
Sensor resolution	2592 x 1944 pixels
Sensor image area	3.76 x 2.74 mm
Pixel size	1.4 μm x 1.4 μm
Optical size	1/4"
Full-frame SLR equivalent	35 mm
S/N Ratio	36 dB
Dynamic range	67 dB @ 8 times gain
Fixed focus	1 m - ∞
Focal length	3.60 ± 0.01 mm
Horizontal field of view (HFOV)	53.50° ± 0.13°
Vertical field of view (VFOV)	41.41° ± 0.11°
Focal ratio (F-stop)	2.9

A popular way of estimating the MTF curve for spatial frequency is called the inclined knife-edge method, in which the curve is obtained from a region of the image where there is a transition from a very dark tone to a very light tone [24]. An Edge Spread Function (ESF) is calculated from the recorded knife edge, giving the unidirectional response of the imaging system to an edge object, replacing the PSF. The LSF can then be obtained in the same way from the derivative of the ESF and finally the MTF is calculated from the Fourier Transform.

In stereo systems, the system MTF is generally summarized as a set of curves for each sensor used, or just the curve of the lowest quality sensor [25]. In this research, the response of all sensors is considered, performing the convolution of the sensors' responses, based on the multiplication of the MTFs in the frequency domain, as illustrated in Equation (3).

$$MTF_{sistema} = \mathcal{F}(LSF_1 * LSF_2) = MTF_1 \times MTF_2 \quad (3)$$

To qualify a sensor, three points of the MTF are usually analyzed: the frequency at which it drops by 50% (at which the image contrast is degraded by half), the frequency at which it drops by 10% (at which the image contrast is image is degraded by 90%) and the MTF value at the Nyquist frequency, which should preferably be greater than 0 [26]. Considering these aspects, the MTF becomes fundamental in analyzing image contrast, so that the impact of spatial resolution and lighting variations can be analyzed. If contrast is compromised, texture and edge details of plants may be damaged to the point of making it impossible to extract features correctly.

Figure 2 shows an example of the typical images where the weed identification task can be performed and the expected size of the plants present. For such situations, the MTF itself can be used in image enhancement processes, based on the deconvolution of the signal based on a Wiener filter [27]. The characterization of the MTF is then useful to define the spatial response of the vision system, considering its detection capacity from a minimum dimension in pixels of the object of interest.

The pseudocode of the system MTF calculation algorithm developed, with the left image I_L, right image I_R and

number of samples n as inputs, can be described as:

function SYSTEM MTF CALCULATION(I_L,I_R,n)
begin function
  for each image I_L and I_R **do**
    Form n *subimages* from regions where there is an inclined knife edge
    for each *subimages* n **do**
      $ESF(n)$ = read_value_pixels(centered horizontal line)
    end for
    ESF = average($ESF(n)$)
    ESF = normalize(ESF)
    LSF = derivative(ESF)
    MTF = Fourier_transform(LSF) ▷ from I_L obtain MTF_L, and from I_R obtain MTF_R
  end for
  MTF_system = $MTF_R \times MTF_L$
return MTF_R, MTF_L and MTF_system
end function

With the three MTFs, it is then possible to validate the sensors individually and together in the system.

D. Stereo Vision Principles

Stereo vision systems are usually based on the use of two cameras with the aim of simulating the human vision system and obtaining depth of objects, with the camera plane as a reference. The depth is acquired through the comparison of the object's position between each captured image [28]. The simplest way of comparing both images is guaranteed when the cameras are coplanar and aligned, as shown in Fig. 3. The variables defined by the camera system are the baseline b and the focal distance f. The $P(X, Y, Z)$ represents a point that would be recorded by the two cameras and $u_L = (X_L, Y_L)$ and $u_R = (X_R, Y_R)$ are the projections of this point in each image. From the concepts of geometry and similarity of triangles, it is possible to obtain:

$$Z = \frac{bf}{X_L - X_R} = \frac{bf}{d} \tag{4}$$

The d variable is called disparity. Thus, with two images as inputs in a calibrated and synchronized stereo architecture, depth information is obtained by finding the corresponding pixels in both images (u_L and u_R) by a matching algorithm and subtracting their X-axis coordinates. By performing this operation for all paired pixels in the image, the disparity map is obtained, which contains all the depth information in the image.

It is also important to note the distortion that variations in the disparity map can cause in the depth estimation, i.e., verify the measurement obtained accuracy. So, for a variation in depth, it is possible to find:

$$\Delta Z = Z - \frac{bf}{d + \Delta d} = \frac{Z^2 \Delta d}{bf + Z\Delta d} \approx \frac{Z^2 \Delta d}{bf} \tag{5}$$

Therefore, when designing a stereo system, attention must be paid to a baseline value at which objects at the distance of interest can be correctly differentiated while the measurement depth distortion must be small. Another important factor when designing such a system is the calibration of and between cameras.

For camera calibration, the set of internal parameters is considered to validate the method. Every camera can be described based on intrinsic and extrinsic parameters, which contribute to how the image is formed from the scene in the real world.

The intrinsic parameters are those related to internal biases, due to the sensor and its shape, the lens and its distortions and other characteristics involved in the manufacture of the camera, while the extrinsic parameters refer to the position of the camera in space in relation to the world. The extrinsic parameters can be simplified by a rotation matrix $\mathbf{R_m}$ and a translation matrix $\mathbf{T_m}$ [29].

The focal length f is an intrinsic parameter, as it is the distance between the center of the camera and the image plane, i.e., from the lens to the censor. Many cameras use a Charge-Coupled Device (CCD) sensor, a semiconductor sensor formed by an integrated circuit that contains a matrix of coupled capacitors, capable of generating electrical stimuli from the light received. As the pixel on a sensor of this type may not be perfectly square, there is the possibility of a small distortion in the number of pixels per unit length. In this way, the focal length of the camera lenses will be different in each direction, resulting in the variables f_u and f_v, with the aspect ratio being defined by f_v/f_u.

Another camera parameter is the optical center, defined by the coordinates (u_0, v_0), which represents a translation factor of the image origin in relation to the center of the sensor, such that the image origin is correctly on the upper edge left of her. There is also the skew coefficient (τ) that corrects the image in cases where the CCD sensor does not have a perpendicular orientation between the length and width axes. As this situation is rare for most sensors, it is common to assume that $\tau = 0$.

Finally, due to the curved nature of lenses, the last intrinsic parameters to be considered when modeling a camera are the distortion coefficients [30]. The tangential distortion coefficients are defined by two variables, k_{p1} and k_{p2}, while the second, fourth and sixth order radial distortion coefficients are respectively represented by k_{q1}, k_{q2} and k_{q3}.

Therefore, the process of capturing a digital image by a sensor can be described in a simplified way using Equation (6), based on the projection of space onto the sensor, where u_d and v_d represent the coordinates of a point in the image without distortion correction, s the scale or resolution factor and X_w, Y_w and Z_w the coordinates of a point in the world.

Figure 2. Typical images of plants in crops for phenotyping task.

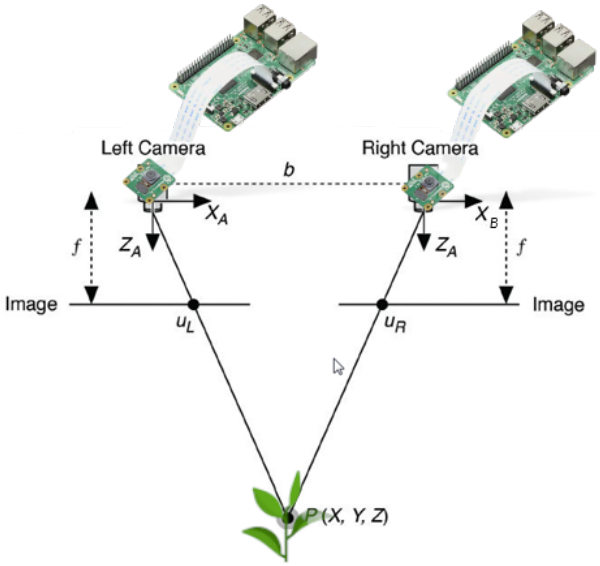


Figure 3. Stereo vision model.

$$s\begin{bmatrix} u_d \\ v_d \\ 1 \end{bmatrix} = \begin{bmatrix} f_u & \tau & u_0 & 0 \\ 0 & f_v & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} \mathbf{R_m} & \mathbf{T_m} \\ 0^T & 1 \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} \quad (6)$$

To find the undistorted coordinates (u, v) of the image, correcting the projection, the system of equations (7) must be solved.

$$\begin{cases} x_d = \dfrac{u_d - u_0}{f_u} \\ y_d = \dfrac{v_d - v_0}{f_v} \\ r^2 = x_n^2 + y_n^2 \\ x_k = x_n(1 + k_{q1}r^2 + k_{q2}r^4 + k_{q3}r^6) \\ x_d = x_k + 2k_{p1}x_n y_n + k_{p2}(r^2 + 2x_n^2) \\ y_k = y_n(1 + k_{q1}r^2 + k_{q2}r^4 + k_{q3}r^6) \\ y_d = y_k + 2k_{p2}x_n y_n + k_{p1}(r^2 + 2y_n^2) \\ u = f_u x_n + u_0 \\ v = f_v y_n + v_0 \end{cases} \quad (7)$$

In addition, when characterizing the intrinsic parameters of any camera, the information can be summarized from two matrices, the camera matrix $\mathbf{M_{cam}}$ and the distortion coefficient matrix $\mathbf{K_{cam}}$, as can be seen in Equations (8) and (9).

$$\mathbf{M_{cam}} = \begin{bmatrix} f_u & \tau & u_0 \\ 0 & f_v & v_0 \\ 0 & 0 & 1 \end{bmatrix} \quad (8)$$

$$\mathbf{K_{cam}} = \begin{bmatrix} kq_1 & kq_2 & kp_1 & kp_2 & kq_3 \end{bmatrix} \quad (9)$$

The process of obtaining such parameters is called camera calibration. Calibration methods depend on the model used to approximate actual camera behavior. The most used models are the linear models of Hall and Faugeras–Toscani, developed respectively in 1982 and 1986, and the non-linear models of Tsai and Weng, implemented in 1987 and 1992, which generally present fewer errors [31].

From Equation (6), the projection matrix between the real world and the image universe will have dimension equal to 3x4, which results in 11 parameters that must be obtained. Commonly, to calibrate stereo systems and obtain matrix

values, images of chessboards with known dimensions (number of squares and size of their side in the real world) are used, in which the calibration points are the internal vertices of the squares on the board. As each point corresponds to two equations (one in the x coordinate and the other in y), five and a half points are needed to calibrate the system, but experiments have shown that 5 times more points than necessary gave better results [32]. A greater number of images for calibration also reduces the total location error in mm, with 13 images in which at least 30% of them were composed of the chessboard, in random orientations, already shows good results. Calibration methods other than the board can also be used, such as calibration using a laser [33] or using spherical objects [34].

The entire stereo vision system must also be calibrated, where are obtained the rotation factor $\mathbf{R_{stereo}}$ and the translation factor $\mathbf{T_{stereo}}$ between the left and right image. For this calculation, the previously calculated camera parameters and the simultaneously captured chessboard images are used. The process of correcting the orientation of stereo images is called rectification. Note that, unlike the camera matrix and distortion coefficients which depend only on the camera, the $\mathbf{R_{stereo}}$ and $\mathbf{T_{stereo}}$ matrices must be recalculated if any stereo system settings change such as, for example, the baseline distance.

III. Results and Discussions

Experimental results were focused on the instrumentation's characterization, i.e., including both the sensors and hardware associated with signal and image processing. So far, the images for such a characterization were collected at laboratory level only. The system is based on eight elements, as follows: 12 V battery; 12 Vdc to 220 Vac voltage inverter; Light Emitting Diode (LED) lamp; 110-220 Vac to 5 Vdc rectifier; two RPis and two Camera Pi, as the schematic presented in Fig. 4. All components are fixed on a metallic structure, with adjustable distance between cameras, angle of inclination (0º, 90º, 180º, 270º) and height of the cameras in relation to the ground (10 to 100 cm). The constructed system can be seen in Fig. 5.

The system is controlled by an Android App via Bluetooth serial communication, where commands can be sent: synchronous image capture on the two RPis, send the images to the cell phone to check the quality of the capture, check the amount of images saved on memory, and board reboot or shutdown command. The RPis also communicate with each other via Bluetooth protocol, that supports up to 7 accessory devices, and uses RFCOMM Bluetooth protocol in data transfer with the cell phone. To ensure system security, it connects only to trusted equipment from their MAC address on specific designated ports.

The elements that most impact the cost of the system are those related to the power supply, sensors and the embedded board. The advantage of the RPi is that it is cheaper when compared to other boards such as the PC/104, although a more detailed analysis should take into account local and freight costs and component availability.

A. Energy consumption management

A RPi can have power consumption of up to 12.5 W, but in laboratory tests the usual value during the application of the image capture software was only 3 W. As the system was designed with an inverter, the power consumed by this equipment must also be considered for system evaluation and possible improvements. In this case, the inverter in question presented a spent power of around 8.4 W, significantly higher than the sum of the RPis. To deal with such power, a battery of 12 V and 60 Ah was chosen.

To measure the energy expenditure of the system, current and power were calculated in different situations, according to Table III, with battery voltage fixed at 12.0 V. To evaluate the battery capacity, a test was carried out in the most extreme situation, with the system in continuous operation with the 18 W LED lamp on, which resulted in the maintenance of operation for approximately 15 hours. When the battery was discharged to 11.7 V, the inverter stopped as a safety precaution. It was observed that, in this operating mode, the peak current at system startup was close to 3.2 A, while with the same configuration but with the less potent LED lamp the peak was 2.0 A.

B. Memory management

For each RPi a 32 GB SD memory card was selected. After the initial settings, the necessary programs installed and the capture algorithm developed, about 23.1 GB of memory was free for general use. To ensure that the program can handle the amount of data written and stored, it is good to know how long the embedded system takes to save files. In testing, it was found that the SD card sequential memory write rate is 14833 KB/s or 14.8 MB/s.

Such information is important to define the resolution in which the images will be captured, as they define the size of the files saved in memory. Following the dimension of the camera sensor, it is preferable to define the resolution of the captured images to take advantage of the entire sensor size, that is, in which the 4:3 ratio is preserved. The maximum file size can be calculated by multiplying the resolution by the pixel depth, but since the Pi Camera doesn't have the option to format a RAW image file, the images are compressed, resulting in smaller files. So, it was tested in five resolutions, 640 x 480, 800 x 600, 1024 x 768, 1280 x 960 and the maximum 2592 x 1944. Early test results can be seen on Fig. 6, where five images in each resolution were taken and saved in the PNG format.

Considering the future application in image processing, in which the computational cost of operations tends to grow exponentially according to the number of pixels present, and the available SD memory, the resolution of 1280 x 960 was then chosen. With this resolution, at least 6,000 images can be saved in memory, although it is possible to store them later in the cloud, from the system's communication with an external network, freeing up space on the board.

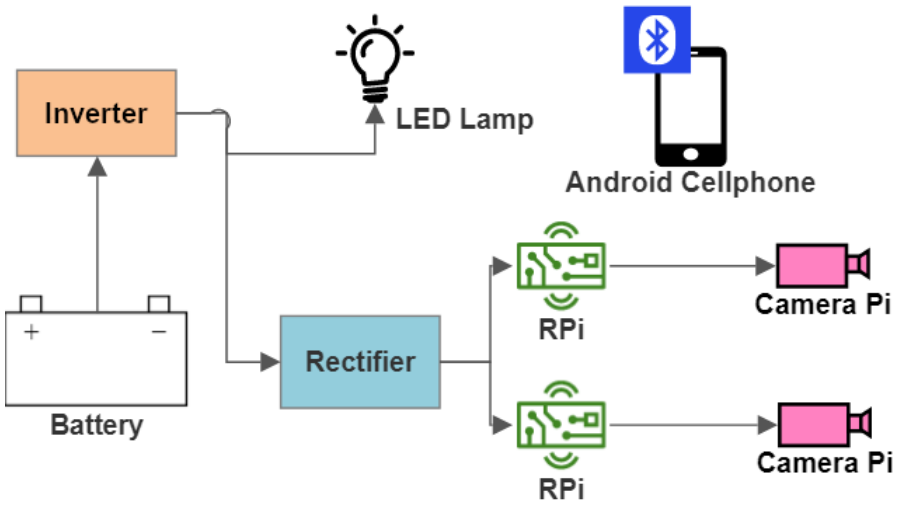
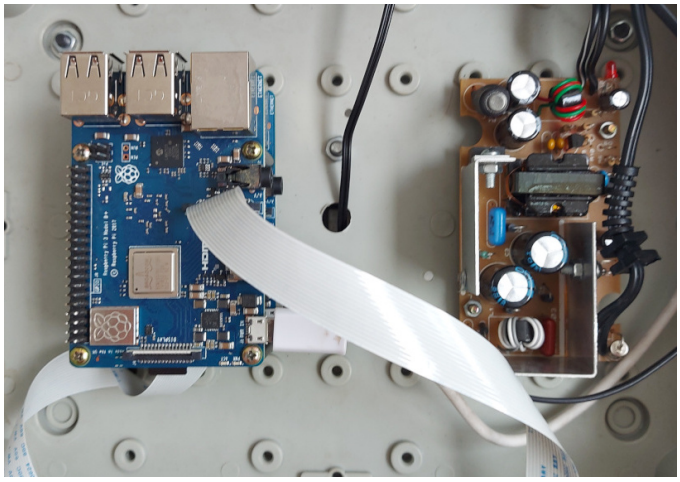


Figure 4. Diagram of the connection between the components.

(a) Details of the camera, stereo rig and lamp.



(b) Interior of protective case, with RPis and rectifier.

Figure 5. Developed system.

TABLE III
SYSTEM POWER AT DIFFERENT SETTINGS

Mode of operation	Current (A)	Power (W)
Standard	1.3	15.6
With active camera sensors	1.4	16.8
With active camera sensors and 4.5 W LED lamp	1.9	22.8
With active camera sensors and 18 W LED lamp	3.0	36

It should be noted that for future applications, if the maximum resolution is used, the memory writing time must be taken into account as a limiting factor.

C. Camera sensor validation

The first step in calculating the stereo MTF was to capture an image of the chessboard with both cameras at the same time, as can be seen in Fig. 7. For each image, five random regions were selected where there are knife edges recorded in the same location for both cameras. The ESF and LSF for a sample of the left and right camera can be exemplified in Fig. 8. The normalized MTF was calculated for each point and averaged between them.

The MTF for the stereo system was then calculated from the convolution in the frequency domain of both partial MTFs, i.e., each one obtained for the cameras used in the developed stereoscopic image system (Fig. 9). For the left camera, 50% of contrast reduction was observed for the normalized special frequency equal to 0.327 cycles/pixel; and 90% of contrast reduction at the 0.551 cycles/pixel. For the right camera, the values of reduction in such frequencies were respectively 0.286 and 0.673 cycles/pixel. Besides, for the entire system, the 50% of contrast could be found in the 0.224 cycles/pixel and the 10% in the 0.367 cycles/pixel. Therefore, the MTF value at the Nyquist frequency was equal to 14.31% for the left camera, 8.97% for the right, and 1.28% for the entire stereoscopic

system. As the MTF value was greater than 5% (contrast reduction that still allows the recovery of the edges of the objects in low noisy images), as well as greater than 0% for both cameras in the system. Such a result qualifies the CCD's sensors, which meet the needs of the developed prototype.

By using the MTF concept, it has become possible to know whether the image will have enough contrast to differentiate the leaves of weed plants when applied in a real agricultural situation. Therefore, considering average values of areas for both weed plants, narrow leaves (monocotyledons) and broad leaves (dicotyledons), the frequencies, in cycles/pixel, could be characterized as 0.053 and 0.100 respectively. Likewise, considering the highest frequency of leaves as a critical point, the MTF presented a value of 97.23% for the left camera, 91.74% for the right and 89.21% for the entire stereoscopic system. Contrast loss values were approximately 10%, which did not interfere with the results, validating the sensor arrangement as suitable to weed family's patterns recognizing.

To evaluate the camera's SNR ratio, only the regions of the converted grayscale image where black blocks were presented, which have a uniform color on the original chessboard, were used, and the mean and standard variation of the signal were evaluated. For the right camera, the calculated value was 19.7 dB, while for the left camera it was 17.9 dB, below the 36 dB specified by the manufacturer.

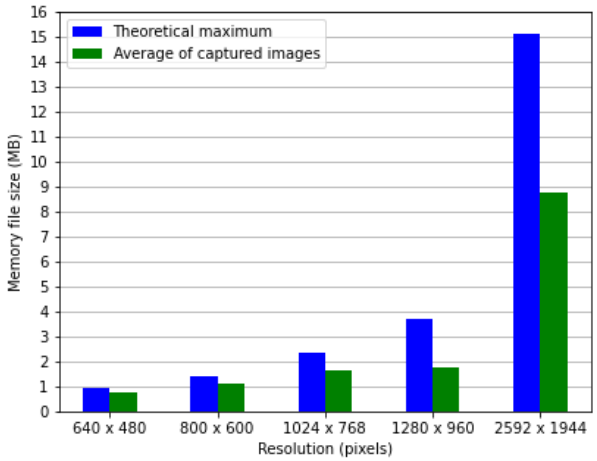


Figure 6. Image file size experimentations.

D. Stereo vision parameters

The first step in tuning the stereo system is to define the baseline distance that will be used to capture the images. The developed prototype has a minimum possible baseline of 6 cm and a maximum of 24 cm, which makes it capable of simulating human vision, which has this value in the range of 5.4 to 7.4 cm, in addition to allowing the exploration of other scenarios. For this, considering (4) and (5), the expected disparity for an object up to 1 m away from the camera and the expected distortion error at such distance were calculated, for four values of baseline, 6 cm, 12 cm, 18 cm and 24 cm, as can be seen in the Fig. 10, considering the resolution of 1280 x 960.

When setting the baseline distance, it is always preferable to use the lower values to ensure greater interpolation between the two generated images, which allows closer objects to have their distance calculated. For example, according to the graph shown, for b = 24 cm, objects up to 23.8 cm away from the camera would not be present in both images, making it impossible to calculate the disparity, while for b = 6 cm such a situation is only valid for objects less than 5.9 cm away. As for objects of up to 1 m, the distortion error proved to be small for all cases, including for the scenario with the smallest baseline, so it can be defined that the best use of the stereo system occurs for values close to 6 cm.

Thus, for b = 6 cm and height of 1 m (value chosen so that, due to the height of the growing plants, the object under analysis is not too close to the sensors), the calibrated parameters results of the left and right cameras, and of the stereo system, were:

$$\textit{Left camera matrix} = \begin{bmatrix} 736 & 0 & 582 \\ 0 & 735 & 464 \\ 0 & 0 & 1 \end{bmatrix} \quad (10)$$

$$\textit{Left distortion coefficients} = \begin{bmatrix} 0.0589 \\ -0.169 \\ 0.00139 \\ 0.00198 \\ 0.142 \end{bmatrix}^{\mathsf{T}} \quad (11)$$

$$\textit{Right camera matrix} = \begin{bmatrix} 1480 & 0 & 681 \\ 0 & 1480 & 480 \\ 0 & 0 & 1 \end{bmatrix} \quad (12)$$

$$\textit{Right distortion coefficients} = \begin{bmatrix} -0.0728 \\ 3.98 \\ 0.00117 \\ 0.00630 \\ -22.6 \end{bmatrix}^{\mathsf{T}} \quad (13)$$

$$\mathbf{R_{stereo}} = \begin{bmatrix} 0.960 & -0.0133 & -0.281 \\ 0.0159 & 1.00 & 0.00721 \\ 0.280 & -0.0114 & 0.960 \end{bmatrix} \quad (14)$$

$$\mathbf{T_{stereo}} = \begin{bmatrix} -0.787 \\ -0.0670 \\ 5.65 \end{bmatrix} \quad (15)$$

Note that if the baseline distance is changed, it is necessary to calibrate the system again, recalculating only the $\mathbf{R_{stereo}}$ and $\mathbf{T_{stereo}}$ matrices, but it is expected that $\mathbf{R_{stereo}}$ will not change significantly, as the mounted structure does not allow the cameras to yaw, pitch or roll.

From these calibration matrices, images can then be correctly rectified, eliminating distortions characteristic of the sensors during image capture and preparing them for use in stereo vision matching algorithms.

(a) Left Camera. (b) Right camera.

Figure 7. Images of a calibration chessboard, captured synchronously and without being processed.

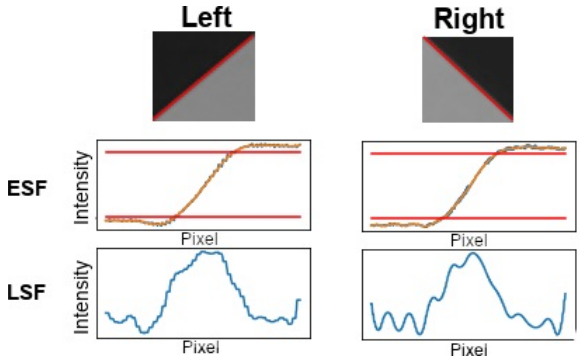


Figure 8. ESF and LSF of a left and right camera sample.

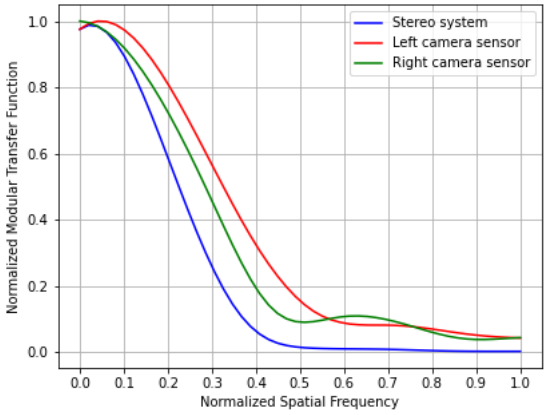


Figure 9. MTF of each camera sensor and combined system.

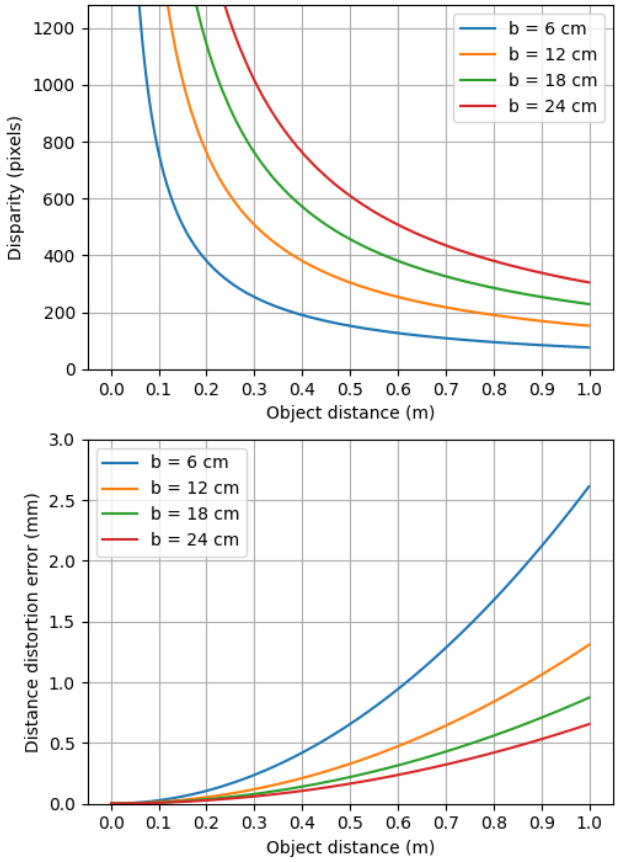


Figure 10. Baseline distance disparity and distortion error evaluation.

IV. Conclusion and Future Work

The results showed a characterization process of an IoT stereo image sensor system, capable of capturing and transferring validated images via wireless commands from serial communication protocols, ready to be used in real agricultural field conditions. In this way, one of main contribution of this work is the construction of a system considering all parameters of casing, structure, power supply, communication, storage memory and hardware and software specifications, ready for use in a real field environment, while previous works only delved into software specifications for application in a controlled laboratory environment.

The developed system can be used in agricultural plantations, with a casing that protects the electrical components from sunlight, wind and light drizzle. It is necessary for a person to control the commands sent to the boards and help move the system, although the device can be adapted to be attached to a vehicle such as a tractor. With dedicated software

for identifying weed families, the device can then be used to generate detailed information, such as a distribution map of the occupancy of a given species in the cultivation area.

The MTF validation principles have demonstrated their importance in ensuring that captured images have sufficient contrast and are capable of observing details of plants with narrow and broad leaves, which allows the correct extraction of real-world data from the information generated by the sensors. Likewise, camera sensor distortions and 3D system calibration are essential so that the data can be used correctly.

Such developed embedded vision system can be useful for applications in 3D image processing, with several variable parameters that allow the adaptation of the system to different situations, although the power supply can be simplified to reduce the weight and power spent of the system, allowing the use of smaller batteries and fewer components (for example, with only a 12 Vdc to 5 Vdc converter and 9 W 12 V LED lamp).

For future steps, it is desired to carry out agricultural analyzes, considering weed families, as well as the inclusion of AI-based weed image process to identify plant species for agricultural control. In addition, an expansion of system's connectivity with other devices will also be realized.

ACKNOWLEDGMENT

This work has been supported by the Brazilian Corporation for Agricultural Research (Embrapa) and the Coordination for the Improvement of Higher Education Personnel (CAPES).

REFERENCES

[1] B. M. Moreno and P. E. Cruvinel, "Characterization of an IoT Stereo Image Sensor System for Weed Control," in ALLSENSORS, International Conference on Advances in Sensors, Actuators, Metering and Sensing, 8th edition, pp. 1–7, 2023.

[2] V. R. Pathmudi, N. Khatri, S. Kumar, A. S. H. Abdul-Qawy and A. K. Vyas, "A systematic review of IoT technologies and their constituents for smart and sustainable agriculture applications," in Scientific African, Vol. 19, p. e01577, 2023.

[3] H. Abouziena and W. Haggag, "Weed control in clean agriculture: a review," Planta daninha, SciELO Brasil, Vol. 34, pp. 377-–392, 2016.

[4] S. C. Bhatla and M. A. Lal, "Plant physiology, development and metabolism," Springer, 2018.

[5] B. M. Moreno and P. E. Cruvinel, "Sensors-based stereo image system for precision control of weed in the agricultural industry," SENSORDEVICES 2018, The Ninth International Conference on Sensor Device Technologies and Applications, pp. 69–76, 2018.

[6] B. M. Moreno and P. E. Cruvinel, "Computer vision system for identifying on farming weed species," 2022 IEEE 16th International Conference on Semantic Computing (ICSC), USA, pp. 287–292, 2022.

[7] N. Higgs, B. Leyeza, J. Ubbens, J. Kocur, W. Kamp, T. Cory, C. Eynck, S. Vail, M. Eramian and I. Stavness, "ProTractor: a lightweight ground imaging and analysis system for early-season field phenotyping," in Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), pp. 2629–2638, 2019.

[8] D. Li, L. Xu, X. Tang, S. Sun, X. Cai and P. Zhang, "3D imaging of greenhouse plants with an inexpensive binocular stereo vision system," in Remote Sensing, Vol. 9, pp. 508, 2017.

[9] M. I. Sadiq, S. M. P. Rahman, S. Kayes, A. H. Sumaita and N. A. Chisty, "A review on the imaging approaches in agriculture with crop and soil sensing methodologies," 2021 Fifth International Conference On Intelligent Computing in Data Sciences (ICDS), Morocco, pp. 1–7, 2021.

[10] M. Rajoriya and T. Usha, "Pattern recognition in agricultural areas," Journal of Critical Reviews, Vol. 7, pp. 1123–1127, 2020.

[11] J. Wäldchen, M. Rzanny, M. Seeland and P. Mäder, "Automated plant species identification—trends and future directions," PLoS computational biology, Vol. 14, pp. 1–19, 2018.

[12] M. P. Raj, P. R. Swaminarayan, J. R. Saini and D. K. Parmar, "Applications of pattern recognition algorithms in agriculture: a review," International Journal of Advanced Networking and Applications, Vol. 6, pp. 2495–2502, 2015.

[13] D. Mcloughlin, "Image processing apparatus for analysis of vegetation for weed control by identifying types of weeds," EP1000540, May 17, 2000.

[14] J. Gao and Z. Jin, "Bionic four-foot walking intelligent rotary tillage weeding device, has weeding system installed on back side of machine body, where gear in gear rotating mechanism transmits power to rotary shaft that is provided with weeding cutter," CN114794067, Jul 29, 2022.

[15] X. Jin, Y. Chen and J. Yu, "Precise weeding method for lawn and pasture based on cloud-killing spectrum, involves receiving images uploaded by each weeding robot, completing weed identification and outputting spraying instructions, and collecting and organizing massive weed data for big data applications," CN113349188, Sep 7, 2021.

[16] S. E. Mathe, M. Bandaru, H. K. Kondaveeti, S. Vappangi and G. S. Rao, "A survey of agriculture applications utilizing raspberry pi," 2022 International Conference on Innovative Trends in Information Technology (ICITIIT), Kottayam, India, pp. 1–7, 2022.

[17] C. Balamurugan and R. Satheesh, "Development of raspberry pi and IoT based monitoring and controlling devices for agriculture," pp. 207–215, 2017.

[18] K. A. Patil and N. R. Kale, "A model for smart agriculture using IoT," 2016 International Conference on Global Trends in Signal Processing, Information Computing and Communication (ICGTSPICC), Jalgaon, India, pp. 543–545, 2016.

[19] C. Bisdikian, "An overview of the Bluetooth wireless technology," in IEEE Communications Magazine, Vol. 39, pp. 86–94, 2001.

[20] Y. Fang, C. Tsai, J. MacDonald and Y. Pai, "Eliminating chromatic aberration in Gauss-type lens design using a novel genetic algorithm," in Applied Optics, Vol. 46, pp. 2401–2410, 2017.

[21] Y. Fang and C. Tsai, "Miniature lens design and optimization with liquid lens element via genetic algorithm," in Journal of Optics A: Pure and Applied Optics, Vol. 10, pp. 075304, 2008.

[22] C. C. Chen, C. M. Tsai and Y. C. Fang, "Optical Design of LCOS Optical Engine and Optimization With Genetic Algorithm," in Journal of Display Technology, Vol. 5, pp. 293–305, 2009.

[23] O. van Zwanenberg, S. Triantaphillidou, R. Jenkin and A. Psarrou, "Edge detection techniques for quantifying spatial imaging system performance and image quality," in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, pp. 1871–1879, 2019.

[24] N. Kawagishi, R. Kakinuma and H. Yamamoto, "Aerial image resolution measurement based on the slanted knife edge method," in Optics Express Vol. 28, pp. 35518–35527, 2020.

[25] A. A. Naumov, A. V. Gorevoy, A. S. Machikhin, V. I. Batshev and V. E. Pozha, "Estimating the quality of stereoscopic endoscopic systems," in Journal of Physics: Conference Series, Vol. 1421, pp. 012044, 2019.

[26] J. L. Alió, P. Schimchak, R. Montés-Micó and A. Galal, "Retinal image quality after microincision intraocular lens implantation," in Journal of Cataract & Refractive Surgery, Vol. 31, pp. 1557–1560, 2005.

[27] E. Oh and J.-K. Choi, "GOCI image enhancement using an MTF compensation technique for coastal water applications," in Opt. Express, Vol. 22, pp. 26908–26918, 2014.

[28] L. Yang, B. Wang, R. Zhang, H. Zhou and R. Wang, "Analysis on location accuracy for the binocular stereo vision system," in IEEE Photonics Journal, Vol. 10, no. 1, pp. 1–16, Art no. 7800316, Feb. 2018.

[29] R. Hartley and A. Zisserman, Multiple View Geometry in Computer Vision, 2nd ed. Cambridge: Cambridge University Press, 2004.

[30] Y. Wang, X. Wang, Z. Wan, and J. Zhang, "A method for extrinsic parameter calibration of rotating binocular stereo vision using a single feature point," in Sensors, Vol. 18, Art no. 3666, pp. 1–16, 2018.

[31] J. Salvi, X. Armangué and J. Batlle, "A comparative review of camera calibrating methods with accuracy evaluation," in Pattern Recognition, Vol. 35, Issue 7, pp. 1617–1635, 2002.

[32] Y. M. Wang, Y. Li and J. B. Zheng, "A camera calibration technique based on OpenCV," The 3rd International Conference on Information Sciences and Interaction Sciences, Chengdu, China, pp. 403–406, 2010.

[33] S. Yang, Y. Gao, Z. Liu and G. Zhang, "A calibration method for binocular stereo vision sensor with short-baseline based on 3D flexible

control field," in Optics and Lasers in Engineering, Vol. 124, pp. 105817, 2020.

[34] J. Sun, X. Chen, Z. Gong, Z. Liu and Y. Zhao, "Accurate camera calibration with distortion models using sphere images," in Optics & Laser Technology, Vol. 65, pp. 83–87, 2015.

A UAV Based System for Real-Time Near-Infrared Monitoring of Small-Scale Wildfires

Edwin Magidimisha
Optronic Sensor Systems, Defence and Security,
Council for Scientific and Industrial Research
Pretoria, South Africa
e-mail: emagidimisha@csir.co.za

Seelen Naidoo
e-mail: snaidoo7@csir.co.za

Zimbini Faniso-Mnyaka
e-mail:zfaniso@csir.co.za

Muhammad Ahmed Nana
e-mail: mnana@csir.co.za

Shrikant Virendra Naidoo
e-mail: svnaidoo@csir.co.za

Vusi Skosana
e-mail: vskosana@csir.co.za

***Abstract*—Wildfires are a global threat that is becoming more severe and widespread due to climate change. These fires not only pose a significant risk to human life, firefighters, and infrastructure, but also endanger forest resources, increase greenhouse gas emissions, and cause huge economic losses. Several researchers have been working to find dedicated solutions for early wildfire detection, tracking, and firefighting assistance. Traditional methods of fire detection have mainly been from fire lookouts in towers, infrared sensors on elevated platforms, surveillance of fires from aircraft, and remote sensing from satellites. Although these techniques have been proven to work in other areas, they are unsuitable or are limited in performance due to various reasons, e.g., human accuracy, sensor field of view limiting coverage to smaller areas, sensor cost-effectiveness, and re-visit time on a satellite. To counteract the problem, a real-time wildfire monitoring system that can detect small-scale wildfire events and that can be used for tactical forest firefighting operations is proposed. The concept takes advantage of vegetation biomass combustion by-products such as the alkali element Potassium (K) that is emitted at the flaming phase of the fire. The technique is specific to the flaming phase of the fire and is not affected by the fire size. It employs two high-resolution, cost-effective complementary metal-oxide-semiconductors (CMOS) with high quantum efficiency within the near-infrared (NIR) spectrum. The sensor uses ultra-narrow-band filtering and target-to-background rationing techniques for the detection of vegetation fires. The system is designed to be self-contained, having its supporting power, compact, and lightweight for easy integration on different types and sizes of unmanned aerial vehicles (UAV) to provide real-time detection and support to firefighters while airborne. UAVs can provide a low-cost alternative for the reduction of fire disasters through early detection, reporting, and real-time support for firefighters. This paper presents the experimental results of an NIR optical sensor mounted on a UAV carrier that was used to collect data while flying at low to 200m above ground at the Centurion Grassland Flying Club. The results provide evidence of the presence of K in small-scale actively burning vegetation fires observed at different angles and detectable from a UAV. The results support the use of NIR sensor payload for the detection of small-scale fires from a UAV platform.**

Keywords - Climate; CMOS; Near-infrared; Potassium (K); UAV; Wildfires.

I. INTRODUCTION

The fire incidences and severity are expected to increase in response to climate change [1, 2, 3, 4]. Fire prevention, detection, monitoring, and suppression of wildland vegetation are key economic and public safety concerns in many parts of the world [5]. These wildfires further exacerbate climate change due to CO_2 and black aerosol emissions. This serves as a strong motivation for the development of an optical surveillance system that can detect and monitor wildfires on a small scale. Classical remote sensing of vegetation fires has been through the detection of Planckian emission in the medium wave infrared (3-5 µm, MWIR) and the long-wave infrared (LWIR) band of the electromagnetic spectrum [6,7,8]. The short wave (1 – 2.5µm, SWIR) infrared band was exploited and deployed on the Airborne Visible Infrared Imaging Spectrometer (AVIRIS) platform [9] for the detection of wildfires. IR-based systems, whether cooled or uncooled, can be costly and significantly affected by other heat-emitting sources, leading to clutter or false alarms [10].

With the advancements in passive imaging sensors and filter technologies, reliable commercial-off-the-shelf (COTS) products are now available and more affordable. New sensor technologies such as high-resolution charge-coupled device (CCD) and complementary metal-oxide-semiconductor (CMOS) sensors provide an opportunity to enhance wildfire detection, monitoring, and reporting. As an alternative to other fire detection techniques, this study proposes the use of a compact and cost-effective system for the detection of wildland vegetation fires by observation of the Potassium (K) spectral line. An initial concept study was performed to characterise the various vegetation species inside and outside the laboratory at the Council for Scientific and Industrial Research (CSIR) campus in Pretoria [11]. The study was made to ensure the relevance of the concept to local conditions by investigating the use of atomic lines emission lines in

burning South African vegetation. Vegetation plant species contain a series of trace elements (Na, K, Mg) that present unique narrowband spectral emission lines in the visible and near-infrared (NIR) wavelength range when biomass is heated to high temperatures during the combustion process [12]. Potassium spectral lines can be discriminated against any other background by detector systems that are less costly than the longer wavelength, actively cooled instruments most used in Earth Observation (EO) systems [12]. The K spectral line doublet located within the NIR at 766.5 and 769.9 nm is of particular interest for this application [10,17,19,20]. The current study integrates the NIR optical payload and operates it from an unmanned aerial vehicle (UAV) using remote sensing techniques.

II. BACKGROUND

In recent years, we have seen great progress in the use of UAVs with advanced software for forest fire monitoring, detection, and firefighting. Integration of UAVs with remote sensing techniques aims to provide rapid, mobile, low-cost, and powerful solutions for various fire tasks [13]. Firefighting agencies typically use fixed detection platforms such as towers, aerial patrols, and satellite imagery to directly detect forest fires, rather than relying on reports from the public. However, high-elevation platforms are not well suited for area coverage and can result in some areas developing fires unnoticed. Although aircraft are considered efficient in firefighting, they are expensive to keep airborne for constant monitoring. Compared to fixed ground-based wildfire detection systems, UAVs can provide a broader and more accurate perception of fire from above, especially in areas that are inaccessible or considered too dangerous for firefighting crews. During firefighting, UAVs provide eyes from above, operators can use them from a safe place and can provide important information on the progression of the fire.

In [14], a vision-based UAV-mounted system for the detection of forest fires that uses both the motion and the chroma characteristics of the fires was proposed. The two characteristics were used for the decision rules to improve the reliability and accuracy of fire detection. A method to detect forest fires using a UAV equipped with an optical and an infrared (IR) camera has been proposed [15]. The method uses a LAB colour model and a motion-based algorithm, followed by a maximally stable extremal region (MSER) extraction module. For better visualisation, forest fire detections were combined with landscape information and meteorological data. In a study in [16], a convolutional neural network (CNN) model was trained using optical and infrared sensor data to detect smoke and fire.

III. DETECTION PRINCIPLE

A simplified schematic of the fire detection principle is shown in Figure 1. The figure illustrates a comprehensive outline of the fire detection system and the principle of operation. The principle relies on the abundant nature of Potassium element in vegetation species. The system incorporates a dual camera to capture and record images of burning biomass fires, specifically vegetation fires containing the Potassium element radiometric signature. One of the sensors is optimized for the detection of the K-line and the other for the detection of the background. The captured images are processed using the in-house developed CSIR algorithm applied during the image processing stage to analyse the pair of images and establish whether a fire has been detected.

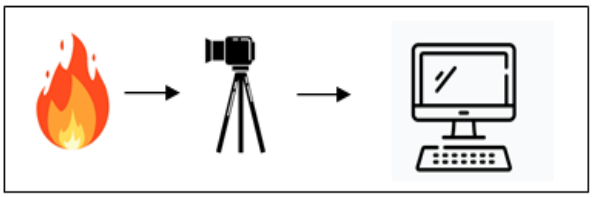


Figure 1: Simplified schematic depicting the overview of the fire detection system.

A. The Potassium Element

Potassium belongs to the alkali metal group and is in the first column of the periodic table. It is one of the most abundant elements in vegetation species [17, 18]. It has a single valence electron that presents unique narrowband spectral emission lines within the visible and NIR wavelength range when biomass is heated to high temperatures in the flaming phase of the fire [19]. The spectral emission of K appears as a doublet at the 766.5 nm and 769.9 nm spectral bands [20]. With advances in optical filter design, filters can now detect low-level signals while suppressing almost all emissions within the outer band by targeting specific elemental emissions from a source signature. These advances in technology open the door for the development of compact sensors capable of detecting narrow spectral lines that can be advanced to compete with other passive sensors operating in other bands. In this study, ultra-narrow band imaging is used for the detection of K using CMOS detectors. The integration of COTS, and ultra-narrow band imaging allows the design of compact and less power-hungry systems, which can be easily integrated on a weight, size, and energy-constrained UAV platform. The CSIR-designed payload weighs 1.8 kg including power support.

B. Fire detection system

A detailed description of the current and futuristic practices in the context of fire detection and monitoring strategies is described in a review paper by F. Khan et al. [21]. Traditional fire detection mechanisms have been through thermal sensors, but other researchers are developing other methods to improve the detection and monitoring of fires for both indoor and outdoor conditions. There are also two broad approaches to fire detection algorithms. The first is using machine learning, which is still in its early stages. The second is to use colour, form, flicker frequency, and the dynamic structure of fire. The fire detection method presented here would fall in the second category. Using radiometric

principles to separate the background from the target (fire) the aspiration is to have a very low false positive rate.

The NIR fire detection sensor presented in this study is made up of two NIR imaging systems placed side-by-side with a common (overlapping) field of view (FOV). These cameras are fitted with ultranarrow band filters with 1 nm bandwidth sensitivity at 769.9 nm, referred to as the K-line band, and 757 nm, referred to as the reference band. The target and reference channels are temporally synchronised at the electronic level so that pairs of images (one from the K-line and the other from the reference band) are obtained at the same instant. Fires are detected by comparing the K-line channel image with the reference channel image. Pixels that are much brighter in the target channel relative to the reference channel are candidate fire detections.

C. Image processing algorithm

The system's image processing begins after the two images are captured, the image with K-line emission, and the other with the background or reference. The images from the two sensors are captured synchronously. The images are not modified with any image enhancement algorithm and are not compressed to preserve the fire front K-line signal emissions. The reference image is resampled to align with the K-line image pixels. This is done by mapping and using a Lucas-Kanada optical flow algorithm [22]. Sections of the individual images that are not common in both are then cropped out, leaving two images of the exact same scene. The fire detection algorithm is applied to the matched cropped K-line and reference images. Fire detection is done using the image ratio technique [23]. Figure 2 illustrates a block diagram that gives an overview of the algorithm.

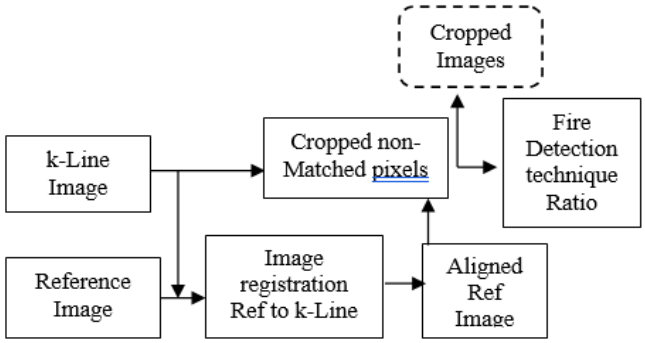


Figure 2: Overview of the K-line fire detection principle.

The K-line and reference images will have the same nominal FOV but will not be pixel-aligned. This is due to the following:

a) The difficulty of perfectly mechanically aligning the optical axes and image plane rotations of the two channels and
b) A possible slight mismatch in the effective focal length (EFL), which means that the two channels will have slightly different image scales and not an exact FOV.
c) Instability of both optical channels due to vibrations during flight.
d) Although the optical systems are identical, there will be minor differences in the image sensor and lens (which need to be corrected).

It may not be possible to rely on a fixed relationship between the pixels of the reference channel and those of the K-line channel from a pre-flight calibration due to the instability from vibrations during flight that could shift the camera's perspectives slightly. Image registration or alignment per image pair is performed in the following way:

a) Feature detection is done by obtaining good features to track as described in [24]. This is done for each image individually, to produce two lists of features.
b) These lists of features are passed to a Lucas-Kanade optical flow algorithm to find and order the features that exist in both images.
c) The features that do not co-exist in both images are pruned and removed from the lists.
d) The perspective transform between the two lists of pruned features is then calculated using the RANSAC method [25, 26].
e) The reference image is then perspectively warped using the previously calculated perspective transforms.

The image ratio technique is simple and is implemented as follows:

a) Compute the ratio image, that is, the K-line image divided by the reference image.

$$im_{Ratio} = im_{kline}/im_{reference}$$

b) Compute the global mean (μ) and variance (σ) of the image ratio.
c) Compute the variant for each pixel in the ratio image as:

$$\sigma_p = \sqrt{(p(i,j) - \mu)^2}$$

where p is at location (i, j).

d) If the variance of a pixel is greater than the global variance multiplied by a user-defined sensitivity integer value i.e., $\sigma_p > k\sigma$,the pixel gets classified as a fire front pixel.
e) Otherwise, the pixel is classified as a non-fire pixel and is discarded.

The result or output of the image ratio technique is a binary mask image that has a value of 1 when fire was detected on that pixel, and a value of 0 when no fire was present. The mask image is then passed onto a simple blob detector [27] to filter out any noise or false detections and automatically indicate when a fire was detected. Automatic flagging is possible since no blobs will be found when there is no fire present.

The entire image processing process was implemented in Python programming language using the OpenCV library.

Processing speed can be trivially improved by using the C++ or CUDA implementations of the OpenCV library.

IV. METHOD

The field measurements test was conducted on the 18th of March 2022 at the Grasslands Flying Club in Pretoria West, South Africa. The purpose of the test was to evaluate the aerial performance of the NIR optical fire detection sensor onboard a UAV. Shown in Figure 3 is a photograph of the NIR imaging sensor system during its lab testing phase.

Figure 3: A closer look at the NIR sensor with two CMOS optical sensors placed side by side and furnished with ultra-narrow filters. A third wide field-of-view visible camera is also inserted and placed above the two cameras.

The UAV Payload uses a development board (Raspberry Pi4 8GB) to control the capturing of images, communication with a ground station, and storage of captured images. The captured images were stored on board a micro-SD card and removed after the completion of a sortie. When the memory card is removed from the payload and the data is retrieved for archival, the data is inspected while the next mission is ongoing. Fire detection is performed on a post-processing basis by automatically analysing the images stored in the memory card.

The basic NIR sensor payload consists of the following components:

- A processor module with storage
- the K-line dual camera system,
- a viewfinder camera,
- a telemetry radio downlink,
- an analog video downlink,
- high-definition video downlink,
- a power source, and
- wiring harnesses.

Figure 4 is a representation of the K-line NIR UAV system and its supporting systems in the operational environment. The list of systems and supporting systems follows with a brief description of the context of a typical operational scenario.

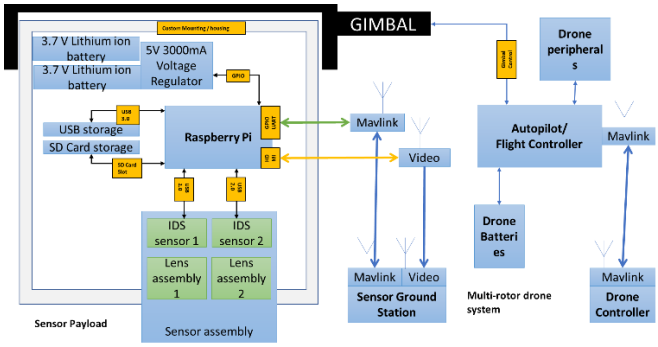


Figure 4: UAV with functional block diagram of the NIR sensor payload

The NIR UAV Payload system (required system) consists of the sensor modules, optics, and processing package in a configuration that accommodates data logging, transmission, and telemetry with the dedicated ground station. The video and telemetry transmission links are separate and isolated from the UAV's communication and control system. The NIR UAV Payload system collectively refers to the physical payload packaging, as well as the ground station and the communication interfacing modules. The use of the system entails the responsibilities of the operator.

The UAV system (supporting system) refers to the UAV airframe (in this instance, a rotary-wing drone) and its gimbal. It includes the UAV pilot's ground station (typically mission planner / ardu pilot). UAV control and gimbal control are designated responsibilities of the UAV pilot. The experiments required coordination between the UAV and payload controller personnel.

The payload system operates in a free-running mode that is triggered by the ground station operator. In the typical context of a fire surveillance exercise for large, restricted areas or where accessibility is challenging, a UAV system is ideal for creating situational awareness of the fire and its spread. The intended mode of operation is illustrated in Figure 5.

The processing module posed significant limitations when implementing onboard processing, making the effective framerate unusable. More limiting was the thermal impact of processing onboard with the processor exceeding its rated threshold. For this reason, the ground station triggered a recording of relevant data, captured to the storage device. Upon the UAV's return to the ground station, the captured data was manually retrieved and post-processed on the ground station system. The video transmission modules were not reliable enough to transmit processable data during flight, hence the decision was taken to post-process data in between each flight path cycle which for the DJI 600 drone was limited to 30 minutes.

A UAV-licensed groundskeeper was tasked to pilot the UAV into a strategic position to capture visual data regarding the fire. The ground station controller has access to trigger the various operational modes of the system. This iteration can

trigger free running record modes, swap between video transmission feeds, and provide general status feedback and control during the flight path. The state mode model of the system is illustrated below.

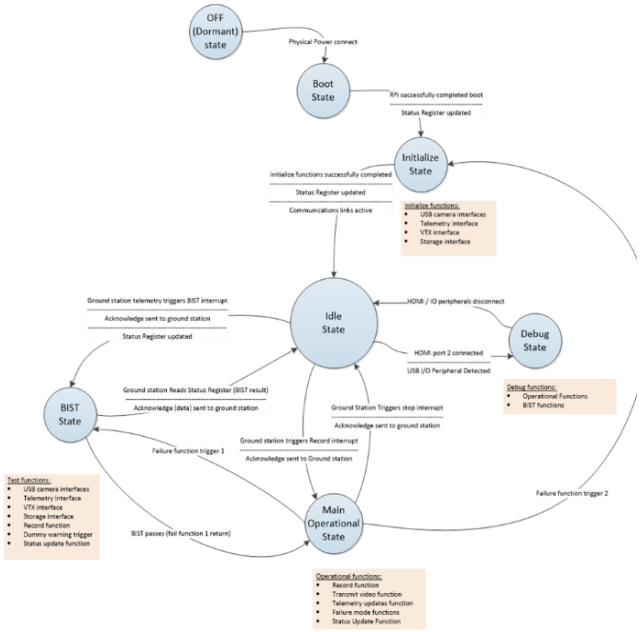


Figure 5: Modes of operation for payload

Shown in Figure 6 is the NIR fire detection payload onboard the UAV taken during the deployment experiment at the Grasslands Flying Club.

Figure 6: UAV with NIR sensor payload on the DJI 600 drone during a field fire detection test of the sensor.

Several sorties were carried out during deployment to test the new NIR payload aboard an airborne UAV. The purpose of the test was to determine whether the new NIR sensor can detect ground wildfires from the air at relatively low altitudes (approximately 200 m above ground level) and at different aspect angles from the fire. The size of the fire on the ground was approximately 500 cm by 500 cm.

The UAV and the Payload systems are completely isolated with respect to power distribution and telecommunications, with the Payload system including its own battery and independent telemetry transceivers. The UAV employs a proprietary gimbal (3 degrees of Freedom) with a manual rotary clamp system. No adhesives, custom mounting brackets, or specialised tools were required for the mechanical coupling of the two systems. The following equipment was used during the test:

- M600 UAV with RONIN gimbal provided and piloted by UAV Industries (UAVI),
- UAV NIR Payload sensor,
- UAV Ground Control Station,
- FieldSpec 3 Max Analytical Spectral Device (ASD) with spectral range 350-2500 nm,
- Weather Station.

A. *Atmospheric Conditions*

During field measurements, the scenario demands that atmospheric computations be made to accommodate the atmospheric effects, caused by molecular absorption and emission (mainly water and CO_2, as well as atmospheric scattering processes by aerosols). Atmospheric modelling codes such as MODTRAN, HITRAN, and others can be used to simulate atmospheric transmission as described below. The radiative transfer is conducted to confirm the detectability of the Potassium lines within the atmosphere.

Atmospheric transmission was calculated using the HITRAN Radiation Transfer Model (RTM) in the NIR region, as shown in Figure 7. The downloaded HITRAN data were on a vacuum scale and converted to air using the Edlen equation (NIST). The following parameters were used: 20°C air temperature, 101325 Pa air pressure, and 50% humidity [18]. The red lines show the K doublet at 766.5 nm and 769.9 nm. The 766.5 nm is absorbed by atmospheric Oxygen (O_2) located at the O_2 absorption line and therefore cannot be detected remotely. The K emission lines are within the range of the sequence of the atmospheric absorption lines that peak at about 762 nm [24].

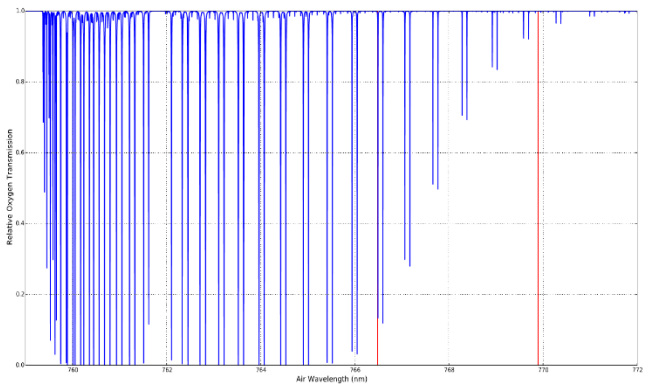


Figure 7: The high spectral resolution Oxygen atmospheric transmittance near the wavelength location of the two Potassium emission spectral lines, data from HiTRAN (http://. iao.ru).

The positions of the K-lines are indicated by two vertical red lines, and the deep lines show the absorption effects arising from the atmospheric Oxygen gas. The filter position choice of 769.9 nm is based on the transmission data above, showing that the 766.5 nm is absorbed by atmospheric Oxygen.

B. Field UAV measurement

The test consisted of a controlled ground fire using wood and dried grass as fuel. An analytical spectral device was placed on the ground close to the fire (approximately 3 m), which was used to record the spectral signature of the fire as it burned. It provided reference spectral data of the fire from the ground to check whether the NIR signature was contained within the fire. The range at which the detection tests were conducted was approximately 200 m (radially) from the fire over various elevation angles with a centered perspective at the burn zone:

- Test point 1: *The elevation angle is 0 degrees, 200 m from the burn zone.*
- Test point 2: *Elevation angle of 45 degrees, 200 m from burn zone.*
- Test point 3: *Elevation angle of 90 degrees (perpendicular to ground level), 200m from the burn zone.*

At these test points, the UAV pilot was unable to maintain rotation orientation (yaw) for data capture due to wind conditions. The position was confirmed through a video stream to the ground station with effort placed in centering the burn zone in the field of view only. The yaw orientation of the sensors had no impact on the detection. These test points provided sufficient data to prove the initial success of the fire detection system. Results are highlighted in Section V. Figure 8 provides an illustrative overview of the mission profile test points used during the fire detection tests.

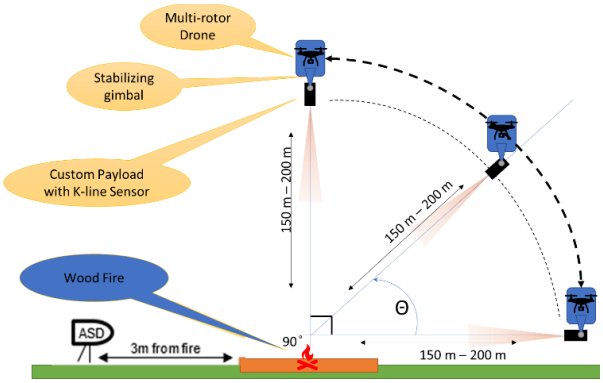


Figure 8: Illustrative overview of flight mission profiles.

C. Hardware Setup

UAV, gimbal and Payload preparations required the UAV operator contractor to provide swappable alternating sets of UAV batteries for the M600 UAV and their control station. The M600 guaranteed a maximum flight time of 30 minutes, of which 20 was allocated to the experiment flight paths, the alternating battery sets allowed for experiment continuity. Similarly, the Payload battery system was designed with two sets of alternating batteries to facilitate the same objective during the experiment. The payload ground station consisted of two laptops in a ruggedized case requiring two operators, viz: a gimbal operator (laptop 1), and a Payload operator (laptop 2). The experimental hardware configuration for the experiment is illustrated in Figure 9 below.

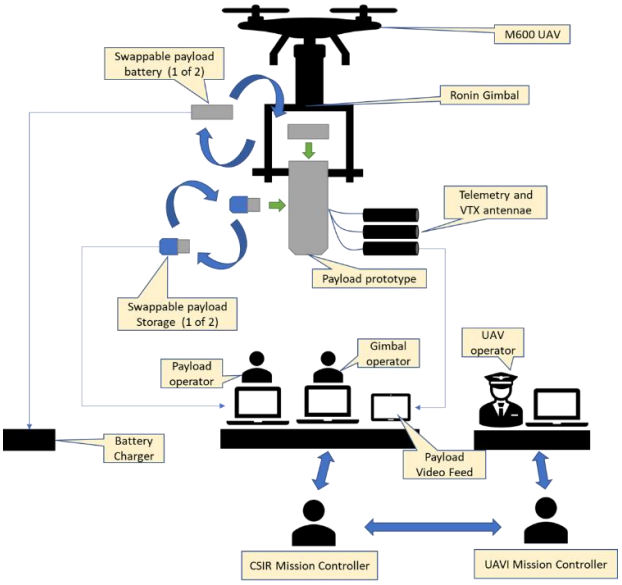


Figure 9: Experimental hardware configuration setup.

V. RESULTS

In this section, the results obtained from the field measurements detection tests, which encompass data collected through both UAV NIR image sensors and spectral measurements recorded by the ASD spectroradiometer, are presented.

A. UAV NIR image sensor data

When examining the results derived from the UAV NIR image sensors, we display them in pairs for clarity. The image on the left represents the masked image, while the image on the right image showcases the target image, which exhibits the distinctive K-line emission signature. Following the application of image processing techniques, the K-line signature is highlighted in red as an overlay, while the black and white target image emphasizes the masking process, effectively isolating the K-line signature.

B. ASD FieldSpec 3 Spectroradiometer Data

The ASD collects a spectrum covering a broad wavelength region (350 nm to 2400 nm) almost instantaneously and has an absolute radiance calibration traceable to NIST. The ASD FieldSpec 3 spectroradiometer data is presented in groups of three images. The top image zooms in on the K-line doublet, offering a detailed view. The image in the middle displays zoomed spectra of several instances of the fire captured at different times, the third figure is a complete spectral image of the fire across the 350-2500 nm spectral band. In these figures, the emission spectrum of the fire becomes prominently visible, with the spectral radiance generally increasing with wavelength. It is clear from the results that the resolution of the ASD is too low and was unable to resolve the K-line doublet.

For this deployment, we conducted controlled burns of dried grass to capture both NIR images from the UAV and spectroradiometer data from the ASD FieldSpec 3. These results contribute to a comprehensive understanding of fire detection mechanisms, spectral signatures, and atomic compositions. Various flight profiles were flown to test the sensor performance at different angles as shared below.

C. Test Point 1:0 Degree Aspect Angle Fire Detection

The image below shows the setup of the NIR imaging sensor at zero degrees relative to the fire.

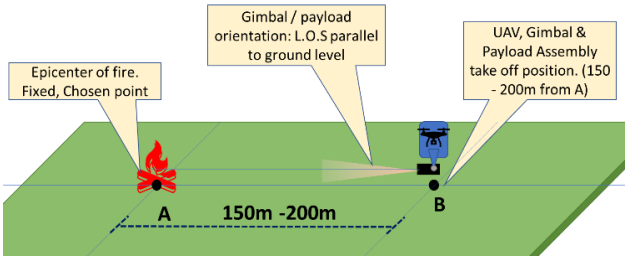
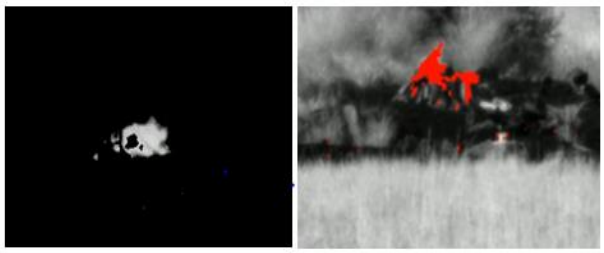


Figure 10: Illustration of the drone viewing the fire at 0 degrees.

The sensor was able to detect fire from an angle (in this scenario, the angle 0° is used). The images were captured while the drone was at 0°, as shown in the image Figure 10.



(a) (b)

Figure 11: NIR sensor images during the lower angle of 0 degrees detection.

The sensor images are as shown above. On Figure 11(a), is the masked image and on the right, Figure 11(b) is the detection image showing the K-line detections in red.

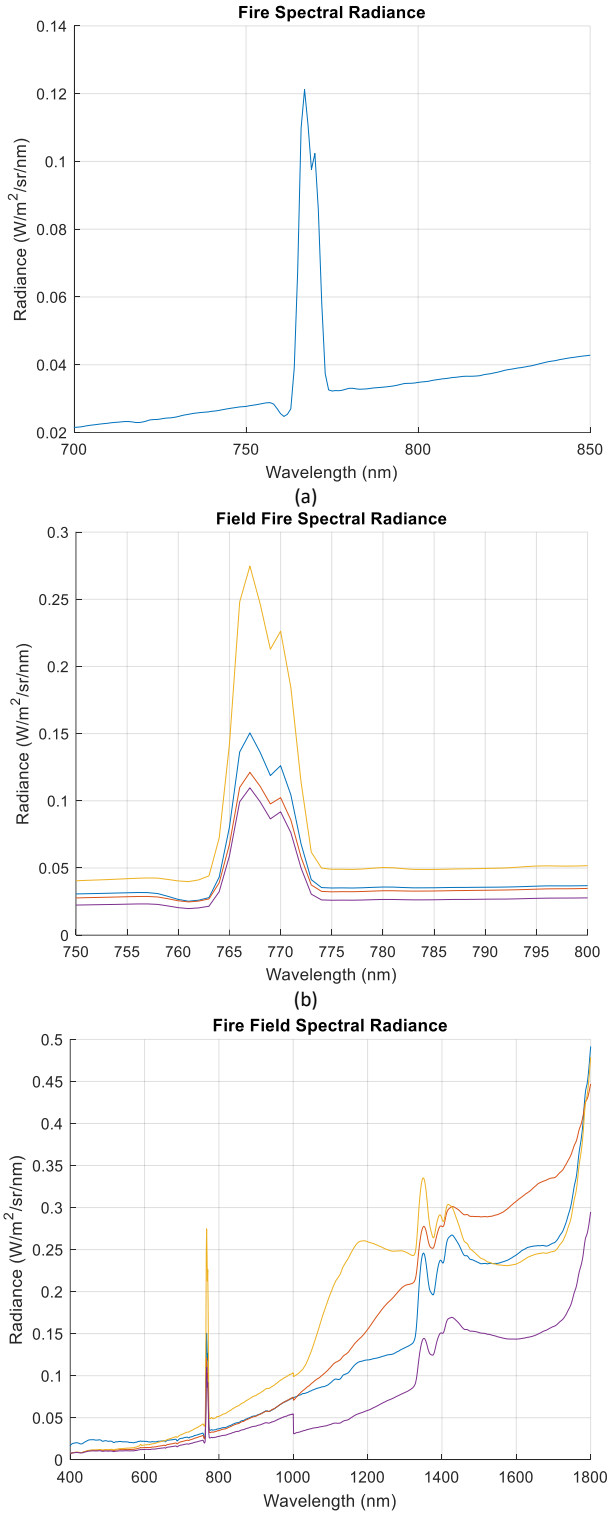


Figure 12: ASD spectral data with NIR-zoomed K-line doublet (a) and (b) and (c) is the full spectral set of the measurements. The Figure shows the spectral radiance of the fire within the NIR region.

The NIR signature was successfully detected in its entirety by the ASD spectral sensor, strategically placed near the fire

scene. The corresponding ASD data is presented in Figure 12. Throughout the airborne operation, multiple spectral measurements were meticulously collected, as visually illustrated in Figures 12(a), Figure 12(b), and Figure 12(c). A similar kind of information is shown in Figure 15 and Figure 18.

D. Test Point 2:45 Degree Aspect Angle Fire Detection

Figure 13 shows the UAV carrying the NIR sensor payload at 45 degrees from the fire.

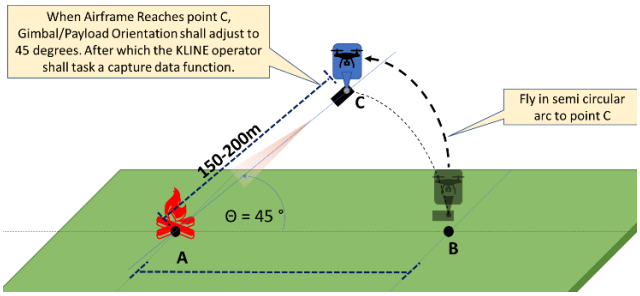
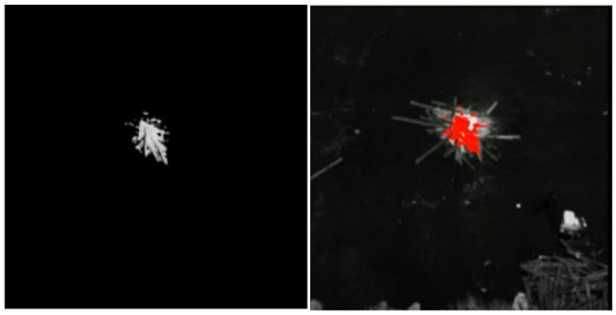


Figure 13: Illustration of the UAV sensor at a 45-degree aspect angle

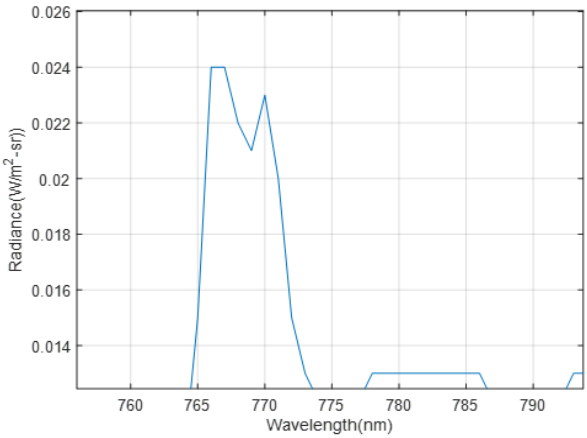
The sensor demonstrated the ability to detect fires from an oblique angle, specifically at 45 degrees in this scenario. Images were acquired during the drone's operation at a 45° angle, as visually depicted in Figure 14. In Figure 14(a), we present the masked image that highlights the K-line emission originating from the fire. Meanwhile, Figure 14(b) presents the unmasked image, with the K-line emission accurately delineated in red for enhanced visibility.



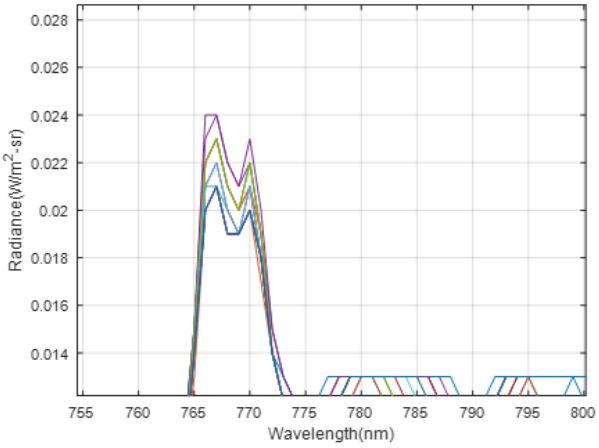
(a) (b)

Figure 14: NIR sensor images during angular (45 degrees) fire detection.

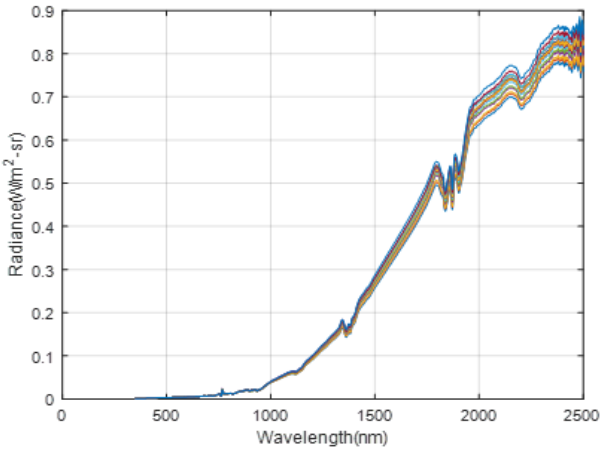
The ASD spectral sensor was strategically positioned near the fire scene, to characterize the flaming vegetation fire spectrally and effectively within the NIR region. While the UAV was in flight or airborne, we conducted multiple ASD spectral measurements, which are illustrated in Figures 15(a), Figure15(b), and Figure 15(c). In particular, Figure 15(a) offers a close-up view of the spectra, highlighting the unresolved K-line doublet, a consequence of the ASD's modest resolution of 3nm.



(a)



(b)



(c)

Figure 15: ASD spectral data with NIR-zoomed K-line unresolved doublet. The fire shows the spectral radiance of the fire within the NIR region.

The full fire spectrum was taken at various instances during the fire progression and it shows an unzoomed K-line presence at 769.9 nm. Visible is the continuous background black body spectrum that rises rapidly with increasing

wavelength. This is purely due to the thermal excitation of all atomic and molecular species within the flaming region. Provided that the fire was flaming (as opposed to smoldering), the burning vegetation within the FOV of the ASD, the K-line doublet was readily evident in the collected spectra.

E. Test Point 3: Flying directly above the fire (90 degrees aspect angle)

Figure 16 shows the UAV carrying the NIR sensor payload at a 90-degree aspect angle from the fire.

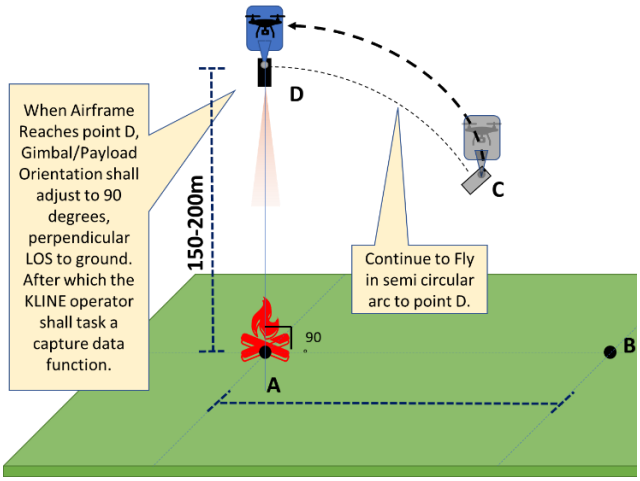
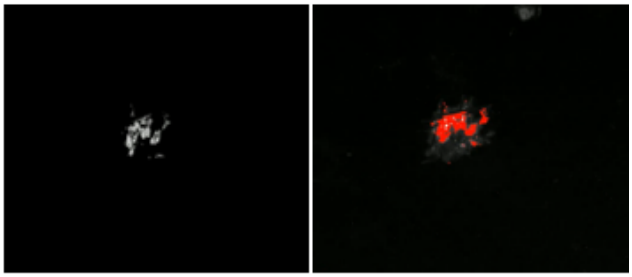


Figure 16: Image depicting the flight path of the UAV from point C TO point D above the fire.

The sensor was able to detect from directly above, as shown in Figure 17. The figure shows the NIR images detected by the K-line band.



(a) (b)

Figure 17: NIR sensor images showing fire detection from directly above (90 degrees).

Successfully data logging was achieved with the ASD sensor, as depicted in Figure 18. Shown in Figure 18(a) are the zoomed and unresolved K-line doublet spectra with a peak at 766.5 nm and 769.9 nm respectively. The lower K-line at 766.5 nm will be absorbed at an increased range as described in section IV. Figure 18(b) is a zoomed ASD spectra of the fire taken at different instances during the flaming phase of the fire. Figure 18(c) is the complete ASD spectra of the fire from 350 nm to 2500 nm taken at various instances during fire progression. A small step at 1000 nm, is a measurement artifact of the ASD, which switches from one internal spectrograph to another at this wavelength.

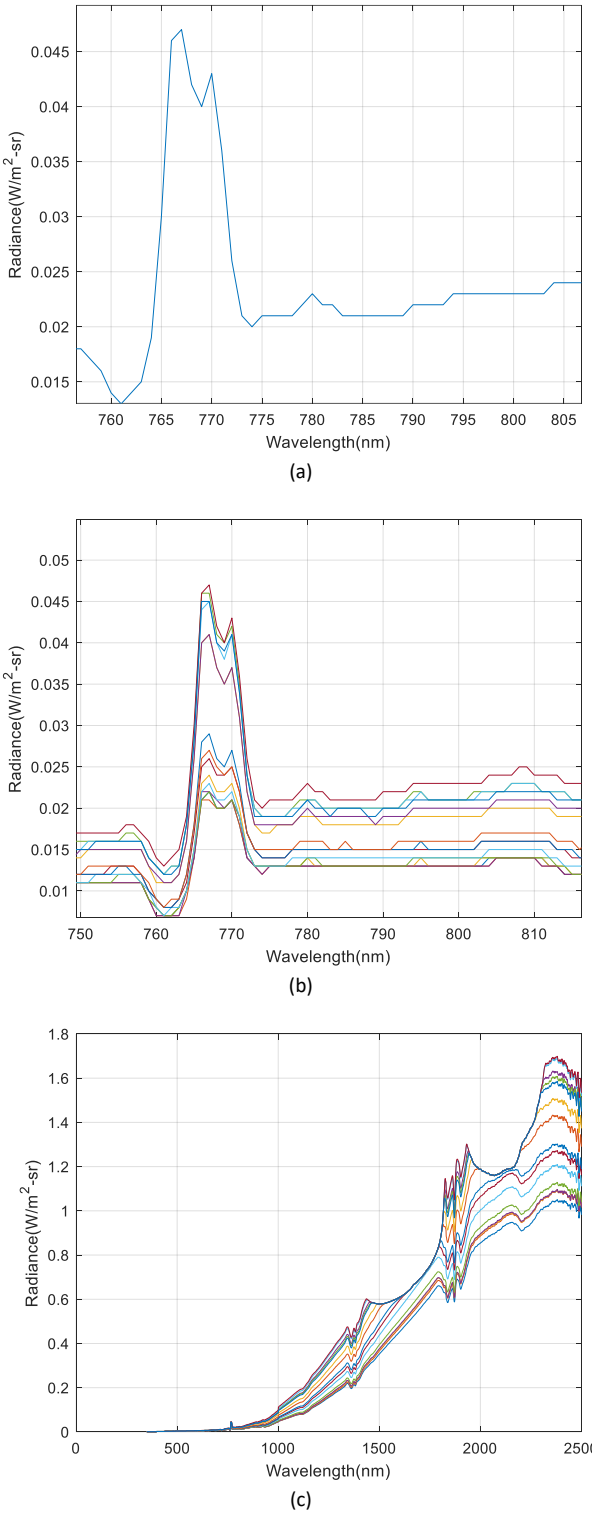


Figure 18: ASD spectral data with NIR-zoomed K-line unresolved doublet (a), zoomed spectra captured at different instances (b), and (c), the ASD spectra from 350nm to 2500nm. The Figure shows the spectral radiance of the fire within the NIR region and the short-wave band.

VI. CONCLUSION

Small-scale fires were captured using a K-line-based fire detection sensor mounted on an unmanned aerial vehicle during a field trial at the Centurion Flying Club, Pretoria, South Africa. The imaging results present strong evidence of the K-line signature within vegetation fires detectable by compact CMOS cameras operating within the NIR spectrum. The ASD spectral measurement confirmed the elemental composition of the vegetation species with a very dominant alkali metal Potassium that is embedded on the spectrum curve. The K element emissions are released at the temperature of the fire at the combustion phase.

This study demonstrates the possibility of performing early fire detection of vegetation biomass using low-cost, higher-resolution NIR sensors integrated into unmanned aerial vehicles coupled with advanced image processing algorithms. This work is recommended as a work in progress to develop a system that will not only detect but track, and geolocate fires, enable fire progression monitoring in areas that are not easily accessible, and finally, facilitate the evolution estimates of fires in real-time.

The limitations on the current development board (RaspberryPi8) such as overheating, and inability to perform onboard processing are targeted as some of the improvements to be considered for the next version of the payload.

ACKNOWLEDGMENT

The Optronic Sensor Systems (OSS) together with all the team members involved in this work owes thanks to the Aeronautical Sciences Impact Area at the CSIR for the logistical assistance and technical inputs to the project. Also, our special thanks to the Department of Science and Innovation for their financial support and funding, for enabling this research effort, as well as UAV Industries (UAV-I) for support in UAV operations during the measurement exercise.

REFERENCES

[1] E. Magidimisha, S. Naidoo, Z. Faniso-Mnyaka, and M. Nana, "Detecting Wildfires Using Unmanned Aerial Vehicle with Near Infrared Optical Imaging Sensor", The Fifteenth International Conference on Advanced Geographic Information Systems, Applications, and Services, IARIA, 978-1-68558-079-7, 2023.

[2] Y. Liu, J. A. Stanturf, and S.L. Goodrick, "Trends in global wildfire potential in a changing climate", Forest Ecology and Management, 259(2010):685–697, 2009. http://dx.doi.org/10.1016/j.foreco.2009. 09.002 (accessed Oct. 25, 2022).

[3] R. Kelly, L. Melissa. C. Philip, E. Higuera, I. Stefanova, B. L. Brubaker, and F. Sheng Hu, "Recent burning of boreal forests exceeds fire regime limits of the past 10 000 years", Proceedings of the National Academy of Sciences, 110(32):13055–13060, 2013. http://dx.doi.org/10.1073/pnas. 1305069110 (accessed Oct. 25, 2022).

[4] P. E. Dennison, D. A. Roberts, and L. Kammer, "Wildfire Detection for Retrieving Fire Temperature from Hyperspectral Data", In ASPRS 2008 Annual Conference, vol. 1, pp. 139–146, 2008. http: //www.asprs.org/a/publications/proceedings/portland08/0015. pdf (accessed Oct. 25, 2022).

[5] S. A. Robert, M. M. Joshua, J. G. Craig, and S. Jennings, "Airborne Optical and Thermal Remote Sensing for Wildfire Detection and Monitoring", MDPI open access article, Sensors 2016.

[6] J. M. Robinson, "Fire from space: global fire evaluation using infrared remote sensing", International Journal of Remote Sensing, vol. 12, pp. 3-24, 1991.

[7] D. O. Fuller, "Satellite remote sensing of biomass burning using optical and thermal sensors", Progress in Physical Geography, vol. 24, pp. 543-561, 2000.

[8] L. B. Lentile, Z. A. Holden, A. M. Smith, M. J. Falkowski, A. T. Hudak, P. Morgan, S. A. Lewis, P. E. Gessler, and N. C. Benson, "Remote sensing techniques to assess active fire characteristics and post fire effects", International Journal of Wildland Fire, vol. 15, pp. 319-345, 2006.

[9] P. J. Thomas and O. Nixon, "Near-infrared forest fire detection concept", Applied Optics, vol. 32, pp. 5348-5355, 1993.

[10] Z. Wang, "Modelling Wildland Fire Radiance in Synthetic Remote Sensing Scenes", PhD thesis, 2007.

[11] E. Magidimisha and D. Griffiths, "Remote optical observations of actively burning biomass fires using potassium line emission", Proceedings of the SPIE, vol. 10036, pp. 331-336, 2016.

[12] A. Stefania, J. Martin, B. Wooster, and A. Piscini, "Multi-resolution spectral analysis of wildfire potassium emission signatures using laboratory, airborne and spaceborne remote sensing", Remote Sensing of Environment, vol. 115, pp. 1811–1823, 2011.

[13] R. S. Allison, A. J. M. Johnston, G. Craig, and S. Jennings, "Airborne Optical and Thermal Remote Sensing for Wildfire Detection and Monitoring", Sensors, 2016.

[14] C. Yuan, Z. Liu, and Y. Zhang, "Vision-based Forest Fire Detection in Aerial Images for Firefighting Using UAVs", Proceedings of 2016 International Conference on Unmanned Aircraft Systems (ICUAS), Arlington VA, USA, 7-10 June 2016.

[15] S. Sudhakar, V. Vijayakumar, C. S. Kumar, V. Priya, L. Ravi, and V. Subramaniya, "Unmanned Aerial Vehicle (UAV) based Forest Fire Detection and monitoring for reducing false alarms in forest-fires", Comput. Commun, vol. 149, pp. 1–16, 2020.

[16] Y. Chen, Y. Zhang, J. Xin, Y. Yi, D. Liu, and H. Liu, "A UAV-based Forest Fire Detection Algorithm Using Convolutional Neural Network", Proceedings of the 37th IEEE Chinese Control Conference, Wuhan, China, 25–27 July, pp. 10305–10310, 2018.

[17] A. Vodacek, R. L. Kremens, A. J. Fordham, S. C. Vangorden, D. Luisi, J.R. Shott, and D. J. Latham, "Remote optical detection of biomass burning using a potassium emission signature", International Journal of Remote Sensing, 23(3), pp. 2721 - 2726, 2002.

[18] Nist Atomic Spectral Database. URL.http://Physics.nist.gov, 2001 (accessed Sep. 21, 2023).

[19] S. Amici, M. J. Wooster, and A. Piscini, "Multi-resolution spectral analysis of wildfire potassium emission signatures using laboratory, airborne and spaceborne remote sensing", Remote Sensing of Environment, vol. 115, no. 8, pp. 1811-1823, Aug. 2011.

[20] D. Latham, "Near-infrared spectral lines in natural fires", Proceedings of the III International Conference on Forest Fire Research/14th Conference on Fire and Forest Meteorology, pp. 513–515, 1998.

[21] F. Khan, Z. Xu, J. Sun, F. M. Khan, A. Ahmed, and Y. Zhao, "Recent Advances in Sensors for Fire Detection," *Sensors*, vol.

22, no. 9, pp. 3310 – 3333, Apr. 2022, doi: 10.3390/s22093310.

[22] "*OpenCV Tutorial Optical Flow"*, docs.opencv.org. https://docs.opencv.org/4.5.1/d4/dee/tutorial_optical_flow.html (accessed Apr. 4, 2023).

[23] A. E. Ononye, A. Vodacek, and R. Kremens, "Fire temperature retrieval using constrained spectral unmixing and emissivity estimation, Algorithms and Technologies for Multispectral, Hyperspectral, and Ultraspectral Imagery", XI 5806, pp. 352 – 360, [doi: 10.1117/12.603440], 2005.

[24] J. Shi and C. Tomasi, "Good features to track", Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, pp. 593–600, IEEE, June 1994.

[25] "*OpenCV Camera Calibration and 3D Reconstruction"*, docs.opencv.org. https://docs.opencv.org/3.4/d9/d0c/group__calib3d.html#ga4abc2ece9fab9398f2e560d53c8c9780 (accessed Apr. 4, 2023).

[26] M. Zuliani, "Ransac for dummies with examples using the ransac toolbox for matlab & octave and more", 2014.

[27] "*OpenCV SimpleBlobDetector Class Reference",* docs.opencv.orghttps://docs.opencv.org/4.x/d0/d7a/classcv_1_1SimpleBlobDetector.html (accessed Apr. 4, 2023).

[28] R. W. B. Pearse and A. G. Gaydon, The identification of Molecular Spectra (London: Chapman and Hall), 1976.

Adding Confidence Intervals to the NESMA Functional Size Estimation Method

Luigi Lavazza
Università degli Studi dell'Insubria
Varese, Italy
email:luigi.lavazza@uninsubria.it

Angela Locoro
Università degli Studi di Brescia
Brescia, Italy
email:angela.locoro@unibs.it

Geng Liu
Hangzhou Dianzi University
Hangzhou, China
email:liugeng@hdu.edu.cn

Roberto Meli
DPO
Rome, Italy
email:roberto.meli@dpo.it

***Abstract*—In many projects, software functional size is measured via the IFPUG (International Function Point Users Group) Function Point Analysis method. However, applying Function Point Analysis using the IFPUG process is possible only when functional user requirements are known completely and in detail. To solve this problem, several early estimation methods have been proposed and have become *de facto* standard processes. Among these, a prominent one is the 'NESMA (NEtherlands Software Metrics Association) estimated' (also known as High-level Function Point Analysis) method. The NESMA estimated method simplifies the measurement by assigning fixed weights to Base Functional Components, instead of determining the weights via the detailed analysis of data and transactions. This makes the process faster and cheaper, and applicable when some details concerning data and transactions are not yet known. The accuracy of the mentioned method has been evaluated, also via large-scale empirical studies, showing that the yielded approximate measures are sufficiently accurate for practical usage. However, a limitation of the method is that it provides a specific size estimate, while other methods can provide confidence intervals, i.e., they indicate with a given confidence level that the size to be estimated is in a range. In this paper, we aim to enhance the NESMA estimated method with the possibility of computing a confidence interval. To this end, we carry out an empirical study, using data from real-life projects. The proposed approach appears effective. We expect that the possibility of estimating that the size of an application is in a range will help project managers deal with the risks connected with inevitable estimation errors.**

***Index Terms*—Function Point Analysis; Early Size Estimation; High-Level FPA; NESMA estimated; Confidence intervals.**

I. Introduction

This paper illustrates the extension of an initial study on the enhancement of the NESMA method with the computation of confidence interval [1].

In the late seventies, Allan Albrecht introduced Function Points Analysis (FPA) at IBM [2], as a means to measure the functional size of software, with special reference to the "functional content" delivered by software providers. Albrecht aimed at defining a measure that might be correlated to the value of software from the perspective of a user, and could also be useful to assess the cost of developing software applications, based on functional user requirements.

FPA is a functional size measurement method, compliant with the ISO/IEC 14143 standard, for measuring the size of a software application in the early stages of a project, generally before actual development starts. Accordingly, software size measures expressed in Function Points (FP) are often used for cost estimation.

The International Function Points User Group (IFPUG) is an association that keeps FPA up to date, publishes the official FP counting manual [3], and certifies professional FP counters. Unfortunately, in some conditions, performing the standard IFPUG measurement process may be too long with respect to management needs, because standard FP measurement can be performed only when relatively complete and detailed requirements specifications are available, while functional measures could be needed much earlier for management purposes.

To tackle this problem, the IFPUG proposes Simple Function Points (SFP). This is an alternative way of measuring the functional size of software: while the SFP method is based on the same concepts as FPA, it requires less detailed information than FPA, so that it is applicable before complete and detailed requirements specifications are available; besides, if is faster and cheaper to apply. As such, it is often presented as a lightweight functional measurement method, also suitable for agile processes. Although the SFP method provides measures that are quantitatively similar to those yielded by FPA, it is not an approximation method for FPA; instead, it is a different measurement method that yields different measures.

Before SFP was proposed, many methods were invented and used to provide *estimates* of functional size measures, based on fewer or coarser-grained information than required by standard FPA. These methods are applied very early in software projects, even before deciding what process (e.g., agile or waterfall) will be used. Among these methods, one of the most widely used is the "NESMA estimated" method [4], which was developed by NESMA [5]. Using this method for size estimation was then suggested by IFPUG [6], which renamed the method High-Level FPA (HLFPA).

The NESMA estimated method has been evaluated by several studies, which found that the method is usable in practice to approximate traditional FPA values, since it yields reasonably accurate estimates, although it has been observed that the NESMA method tends to underestimate size, which is potentially dangerous.

Many estimation methods provide a "confidence interval", meaning that instead of providing a single value, they compute

an interval in which the actual size is expected—with a given confidence level—to be. The greater the required confidence, the greater the interval. Knowing the confidence interval is considered very useful by project managers, because it helps managing the risk deriving from inevitable estimation errors and the inherent uncertainty of estimates. Unfortunately, the NESMA estimated method does not provide a confidence interval. Recently, a proposal for enhancing the NESMA estimated method with a mechanism confidence intervals has been published [1]. Specifically, that paper provided two main contributions: the correction of the NESMA method to eliminate underestimation, and the introduction of confidence intervals. The original study was based on a single dataset, and reported the hypothesis that different datasets may require different corrections and support different confidence intervals.

This paper extends the initial study [1] by verifying that a different dataset actually requires a different correction of the NESMA method and provides different confidence ranges. Therefore, the proposed numerical method is an instrument that lets software project managers get corrected size estimates, equipped with confidence intervals, which apply to specific data, in a context-aware manner.

In addition, this paper illustrates how to take advantage of confidence intervals in real-life situations, by introducing an example of how the proposed technique can be used in practice for effort estimation.

The remainder of the paper is organized as follows. Section II provides an overview of FPA and the NESMA method. Section III describes the empirical study and its results, which are discussed in Section IV. Section V illustrates the usage of the proposed techniques in practical project management, namely, for effort estimation. In Section VI, we discuss the threats to the validity of the study. Section VII reports about related work. Finally, in Section VIII, we draw some conclusions and outline future work.

II. Background

Function Point Analysis was originally introduced by Albrecht to measure the size of data-processing systems from the point of view of end-users, with the goal of the estimating the value of an application and the development effort [2]. The critical fortunes of this measure led to the creation of the IFPUG (International Function Points User Group), which maintains the method and certifies professional measurers.

The "amount of functionality" released to the user can be evaluated by taking into account 1) the data used by the application to provide the required functions, and 2) the transactions (i.e., operations that involve data crossing the boundaries of the application) through which the functionality is delivered to the user. Both data and transactions are counted on the basis of Functional User Requirements (FURs) specifications, and constitute the IFPUG Function Points measure.

FURs are modeled as a set of Base Functional Components (BFCs), which are the measurable elements of FURs: each of the identified BFCs is measured, and the size of the application is obtained as the sum of the sizes of BFCs. IFPUG BFCs are: data functions (also known as logical files), which are classified into Internal Logical Files (ILF) and External Interface Files (EIF); and Elementary Processes (EP)—also known as transaction functions—which are classified into External Inputs (EI), External Outputs (EO), and External inQuiries (EQ), according to the activities carried out within the considered process and the primary intent.

The complexity of a data function (ILF or EIF) depends on the RETs (Record Element Types), which indicate how many types of variations (e.g., sub-classes, in object-oriented terms) exist per logical data file, and DETs (Data Element Types), which indicate how many types of elementary information (e.g., attributes, in object-oriented terms) are contained in the given logical data file.

The complexity of a transaction depends on the number of FTRs—i.e., the number of File Types Referenced while performing the required operation—and the number of DETs—i.e., the number of types of elementary data—that the considered transaction sends and receives across the boundaries of the application. Details concerning the determination of complexity can be found in the official documentation [3].

The core of FPA involves three main activities:

1) Identifying data and transaction functions.
2) Classifying data functions as ILF or EIF and transactions as EI, EO or EQ.
3) Determining the complexity of each data or transaction function.

The first two of these activities can be carried out even if the FURs have not yet been fully detailed. On the contrary, activity 3 requires that all details are available, so that FP measurers can determine the number of RET or FTR and DET involved in every function. Activity 3 is relatively time- and effort-consuming [7].

Note that IFPUG defines both unadjusted FP (UFP) and adjusted FP. The former are a measure of functional requirements. The latter are obtained by correcting unadjusted FP to obtain an indicator that is better correlated to development effort. Noticeably, the ISO standardized only unadjusted FP, recognizing UFP as a proper measure of functional requirements [8]. Following the ISO, in this paper we deal only with UFP, even when we speak generically of Function Points or FP.

The NESMA estimated method does not require activity 3, thus allowing for size estimation when FURs are not fully detailed: it only requires that the complete sets of data and transaction functions are identified and classified.

The SFP method [9] does not require activities 2 and 3: it only requires that the complete sets of data and transaction functions are identified.

Both the NESMA estimated method and SFP methods let measurers skip the most time- and effort-consuming activity, thus both are relatively fast and cheap. The SFP method does not even require classification, making size estimation even faster and less subjective (since different measurers can sometimes classify differently the same transaction, based on the subjective perception of the transaction's primary intent).

NESMA defined two size estimation methods: the 'NESMA Indicative' and the 'NESMA Estimated' methods. IFPUG acknowledged these methods as early function point analysis methods, under the names of 'Indicative FPA' and 'High-Level FPA,' respectively [6]. The NESMA Indicative method proved definitely less accurate [10], [11]. Hence, in this paper, we consider only the NESMA Estimated method.

The NESMA Estimated method requires the identification and classification of all data and transaction functions, but does not require the assessment of the complexity of functions: ILF and EIF are assumed to be of low complexity, while EI, EQ and EO are assumed to be of average complexity. Hence, estimated size is computed as follows:

$$EstSize_{UFP} = 7\ \#ILF + 5\ \#EIF + 4\ \#EI + 5\ \#EO + 4\ \#EQ$$

where *#ILF* is the number of data functions of type ILF, *#EI* is the number of transaction functions of type EI, etc.

III. Empirical study

In this section, the empirical study is described: Section III-A described the datasets used for the reported analysis; Section III-B illustrates some considerations concerning the accuracy of the NESMA method that affect the study, and introduces the correction of the NESMA method, to avoid size underestimation; Section III-C describes how the study was performed; Section III-D describes the obtained results. While the aforementioned sections use the same dataset used previously [1], the following Sections III-E and III-F use a second dataset, to replicate the previous study.

A. The datasets

In the empirical study, we used two datasets. One is the ISBSG dataset [12], which has been extensively used for studies concerning functional size [13]–[18].

The ISBSG dataset contains many data concerning software development projects. Of the many available data, we considered only the project size, expressed in UFP, and the components used to compute the size, i.e., *#ILF*, *#EIF*, *#EI*, *#EO* and *#EQ*.

The ISBSG dataset contains several small project data. As a matter of fact, estimating the size of small projects is not very interesting. Based on these considerations, we removed from the dataset the projects smaller than 100 UFP (Unadjusted Function Points). The resulting dataset includes data from 140 projects having size in the [103, 4202] range. Some descriptive statistics for this dataset are given in Table I.

TABLE I
Descriptive statistics for the ISBSG dataset (after removing small projects).

	UFP	*#ILF*	*#EIF*	*#EI*	*#EO*	*#EQ*	NESMA
Mean	801	22	20	35	37	37	730
Std	818	21	22	37	65	48	721
Median	475.5	14	14.5	22	10	20.5	463
Min	103	0	0	0	0	0	71
Max	4202	100	172	204	442	366	3755

The second dataset was provided by a Chinese company (whose identity we cannot disclose) that is active in the banking and finance domain. Although not popular as the ISBSG dataset, also the Chinese dataset was formerly used in a few studies concerning functional size measurement [19], [20].

Also with this dataset (which is called the "Chinese" dataset throughout the paper) we removed the data concerning projects smaller than 100 UFP. As a result, we obtained a dataset containing 424 project data: some descriptive statistics for the dataset are given in Table II. It can be noticed that the Chinese dataset includes data form much larger projects than the ISBSG dataset.

TABLE II
Descriptive statistics for the Chinese dataset (after removing small projects).

	UFP	*#ILF*	*#EIF*	*#EI*	*#EO*	*#EQ*	NESMA
Mean	3819	90	47	303	122	246	3670
Std	5877	180	129	516	319	470	5706
Median	1447	31	7	116	29	77	1484
Min	103	0	0	0	0	0	99
Max	35910	2169	1198	3551	4517	4231	37571

B. The accuracy of the NESMA estimated method when applied to the ISBSG dataset

As already observed in previous papers [18], [21], the NESMA estimated method tends to underestimate. Figure 1 shows that more than 75% of the NESMA estimates of ISBSG project size have positive error. Being the error defined as the actual size (i.e., the size measured via the ISBSG standard FPA process) minus the estimate, positive error indicate underestimation.

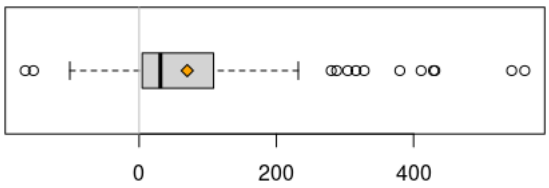


Fig. 1. Boxplot of estimation errors by the NESMA method, when applied to the ISBSG dataset.

In addition, Figure 1 suggests that the distribution of NESMA errors is skewed. The skewedness of NESMA errors is clearly visible in Figure 2, which illustrates the distribution of errors: it is easy to notice that most errors are positive.

For our purposes, the fact that the distribution of NESMA errors is skewed and not centered on zero means that we cannot evaluate confidence errors as is usually done. Specifically, given a confidence level C, we cannot select two error levels e_L and e_H that are symmetric with respect to the mean error $\bar{e}$ (i.e., $|e_H - \bar{e}| = |\bar{e} - e_L|$) such that the proportion of errors such that $e_H \geq error \geq e_L$ is C.

Since it makes hardly sense to provide confidence intervals for a method that underestimates systematically, we first

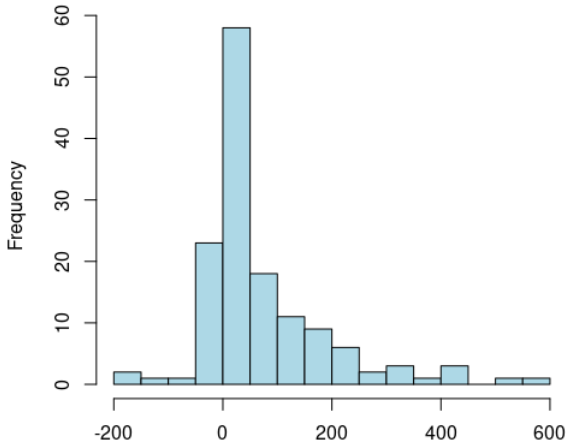


Fig. 2. Histogram of estimation errors by the NESMA method, when applied to the ISBSG dataset.

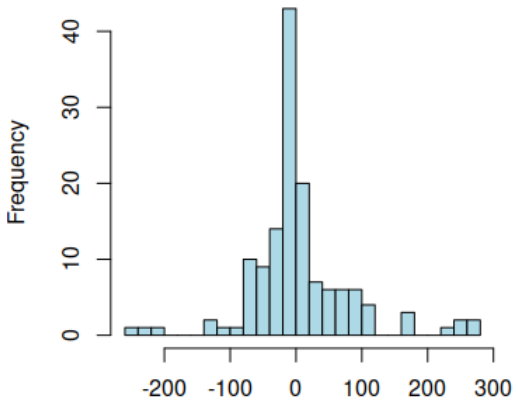


Fig. 4. Boxplot of estimation errors by the corrected NESMA method, when applied to the ISBSG dataset.

"correct" the NESMA estimated method. The mean actual size is 801 UFP, while the mean size estimated via the NESMA method is 730 UFP. The ratio between these two means is approximately 1.09. Accordingly, we need to correct NESMA estimates, multiplying them by 1.09, to make the means of the two distributions equal (in [1] the correction factor was 1.08; subsequent more accurate evaluations led to set the correction factor to 1.09). In this way, we obtain estimates that have a better error distribution (less skewed and centered around zero) and a smaller mean absolute error (49.7 UFP instead of 83.8 UFP).

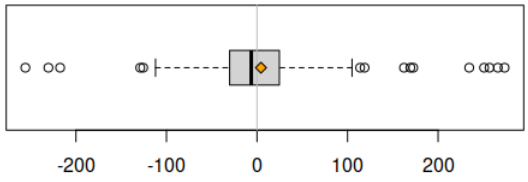


Fig. 3. Boxplot of estimation errors by the corrected NESMA method, when applied to the ISBSG dataset.

The boxplot of estimation errors obtained with the corrected NESMA method is shown in Figure 3: it can be noticed that the mean error is just above zero, while the median error is just below zero.

The error distribution is shown in Figure 4: it can be noticed that the distribution is much less skewed than in Figure 2.

Since the practical objective of this work is to provide project managers with reliable predictions of functional size, in what follows we consider only estimates provided by the original NESMA method and corrected as described above. In other words, we consider the following estimates:

$$EstSize_{UFP} = 1.09\ (7\ \#ILF + 5\ \#EIF + 4\ \#EI + 5\ \#EO + 4\ \#EQ)$$

We make reference to this estimation as the "Corrected NESMA" method.

C. Method used

In essence, given a confidence level C we aim at finding two values k_L and k_H such that a proportion C of the actual size measures (i.e., measures obtained via the official IFPUG FPA process) is in the range $[k_L \cdot EstSize_{UFP}, k_H \cdot EstSize_{UFP}]$, where $EstSize_{UFP}$ is the size estimates computed via the Corrected NESMA method.

Finding k_L and k_H would be straightforward if the estimation errors obtained via the Corrected NESMA method were normally distributed. Instead, it is not so, as shown by the Shapiro-Wilk test.

Therefore, we proceeded as follows:

1) We computed the ratio $\frac{ActualSize}{EstSize_{UFP}}$ for all projects in the dataset, obtaining a set of ratios; this set was then sorted and stored in vector *vRatios*.
2) We computed the quantiles from 0 to 1, with 0.01 steps, of *vRatios*, obtaining an ordered vector *vQuant*.
3) We looked for two indexes i_L and i_H in *vQuant* such that $i_H - i_L + 1 = C \cdot n$, where n is the number of projects in the dataset.
4) k_L and k_H are the values in *vRatios* having index i_L and i_H, respectively, i.e., $vRatios[i_L]$ and $vRatios[i_H]$.

In this way, we obtain a size estimate interval that contains a proportion C of all estimates, such that all estimation errors outside the interval are greater than those within the interval.

D. Results obtained for the ISBSG dataset

We applied the procedure described in Section III-C for various confidence levels. The results obtained are given in Table III. Note that these results depend on the dataset being used, in our case, the ISBSG dataset. In other contexts, a given confidence level could correspond to different confidence intervals. For instance, in the ISBSG dataset, the minimum and maximum ratios $\frac{ActualSize}{EstSize_{UFP}}$ are 0.758 and 1.343, respectively; in another dataset, a smaller minimum and a larger maximum ratios are clearly possible, as shown in Section III-F.

TABLE III
CONFIDENCE INTERVALS FOR VARIOUS CONFIDENCE LEVELS, FOR THE ISBSG DATASET.

conf. level	k_L	k_H
0.10	0.991	1.011
0.20	0.980	1.019
0.30	0.968	1.030
0.40	0.954	1.043
0.50	0.943	1.057
0.60	0.929	1.077
0.70	0.910	1.095
0.80	0.872	1.137
0.90	0.843	1.208
0.95	0.818	1.208
1.00	0.751	1.331

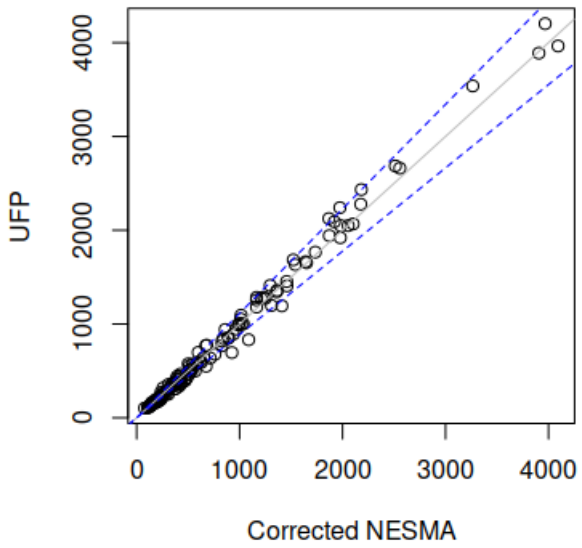


Fig. 5. Corrected NESMA estimates vs. actual size in UFP, with confidence $C = 0.75$, for the ISBSG dataset.

For illustration purposes, Figure 5 plots the ISBSG project data in the plan defined by actual size (the y axis) and the size estimated via the Corrected NESMA method (the x axis). In the plot, the dashed blue lines represent the $y = k_L\ x$ and $y = k_H\ x$ lines.

E. The accuracy of the NESMA estimated method when applied to the Chinese dataset

As observed in Section III-B, the NESMA estimated method tends to underestimate. Figure 6 shows that the majority of the NESMA estimates of the Chinese dataset project size have positive error (positive errors indicate underestimation).

As for the ISBSG dataset, the distribution of NESMA errors is skewed (although less evidently than for the ISBSG dataset), as shown in Figure 7. It is easy to notice that most errors are positive.

Therefore, we corrected NESMA estimation, as we did for the ISBSG dataset. As discussed above, we cannot just apply the same multiplier found for the ISBSG dataset. For the Chinese dataset, the mean actual size is 3819 UFP, while the mean size estimated via the NESMA method is 3670 UFP.

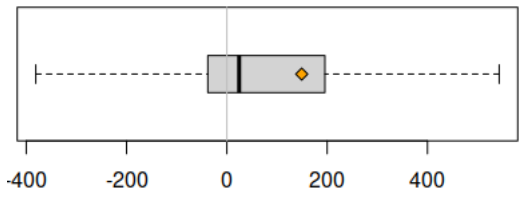


Fig. 6. Boxplot of estimation errors by the NESMA method, when applied to the Chinese dataset (outliers not shown).

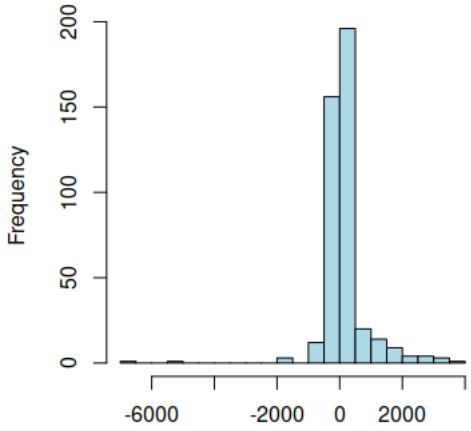


Fig. 7. Histogram of estimation errors by the NESMA method, when applied to the Chinese dataset.

The ratio between these two means is approximately 1.04. Accordingly, we correct NESMA estimates, multiplying them by 1.04, to make the mean value of the estimates sizes equal to the mean value of the actual sizes. In this way, we obtain estimates that have a better error distribution (less skewed and centered around zero) and a smaller mean absolute error (319 UFP instead of 340 UFP).

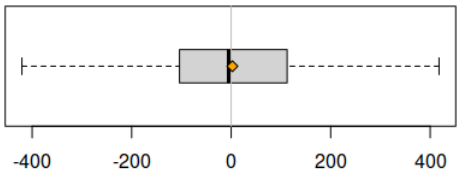


Fig. 8. Boxplot of estimation errors by the corrected NESMA method, when applied to the Chinese dataset (outliers not shown).

The boxplot of estimation errors obtained with the corrected NESMA method is shown in Figure 8: it can be noticed that the mean error is just above zero, while the median error is just below zero.

The error distribution is shown in Figure 9: it can be noticed that the distribution is less skewed than in Figure 7.

In what follows we consider the "Corrected NESMA" estimates, obtained as follows:

$$EstSize_{UFP} = 1.04\ (7\ \#ILF + 5\ \#EIF + 4\ \#EI + 5\ \#EO + 4\ \#EQ)$$

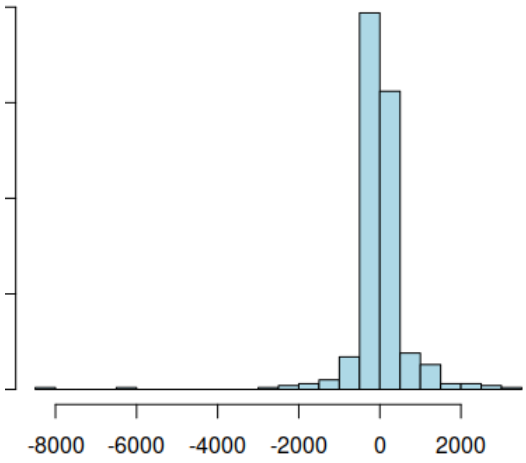


Fig. 9. Boxplot of estimation errors by the corrected NESMA method, when applied to the Chinese dataset.

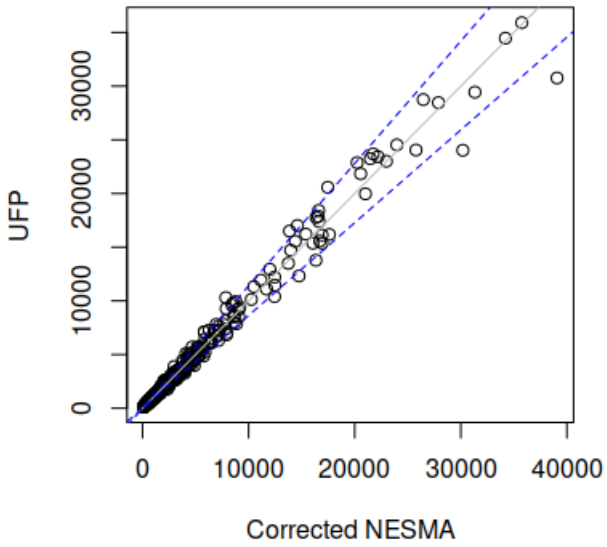


Fig. 10. Corrected NESMA estimates vs. actual size in UFP, with confidence $C = 0.75$, for the Chinese dataset.

F. Results obtained for the Chinese dataset

We applied the procedure described in Section III-C for various confidence levels. The results obtained are given in Table IV. As expected, the results obtained for the Chinese datasets are different from those derived from the ISBSG dataset: for a a given confidence level, we obtained different confidence intervals. Specifically, the confidence intervals are larger for the Chinese dataset than for the ISBSG dataset. For instance, the minimum and maximum ratios $\frac{ActualSize}{EstSize_{UFP}}$ are 0.741 and 1.597, respectively, while they were 0.758 and 1.343, respectively, for the ISBSG dataset.

TABLE IV
CONFIDENCE INTERVALS FOR VARIOUS CONFIDENCE LEVELS, FOR THE CHINESE DATASET.

conf. level	k_L	k_H
0.10	0.984	1.018
0.20	0.967	1.033
0.30	0.950	1.051
0.40	0.933	1.066
0.50	0.921	1.082
0.60	0.905	1.102
0.70	0.882	1.119
0.80	0.843	1.165
0.90	0.791	1.214
0.95	0.741	1.248
1.00	0.741	1.597

For illustration purposes, Figure 10 plots the ISBSG project data in the plan defined by actual size (the y axis) and the size estimated via the Corrected NESMA method (the x axis). In the plot, the dashed blue lines represent the $y = k_L\ x$ and $y = k_H\ x$ lines.

IV. DISCUSSION OF RESULTS

In the previous sections, we exploited two datasets that collect measures from real-life projects to determine i) a correction of the estimates provides by the NESMA method, and ii) confidence intervals for the corrected estimates.

The results of the study show that organizations that own historical data like those we used can apply the procedure illustrated in Sections III-B and III-C to derive the correction constant and the confidence intervals that suite best their development process.

Unfortunately, organizations that do not own historical data like those we used cannot derive the correction constant nor confidence intervals. However, they can adopt some rule of thumb to improve the performances of NESMA estimates. Specifically, the correction constant can be set to a value between 1.04 and 1.09, based on our findings. Similarly, confidence intervals can be defined, based on Tables III and IV. However, these organizations should be aware that the data we used might not match their situations, hence both the correction constant and the confidence intervals might not be perfectly suited for their case.

The confidence interval can be used to perform risk analysis. For instance, Table III shows that, given an estimate already corrected with respect to the NESMA original prediction, there are 30% probabilities that the actual size is more than 10% different (greater or smaller) than estimated. Most likely, half of these 30% probabilities concern underestimation: as a result, a project manager should consider that the probability of underestimating functional size of 10% or more is around 15%. The risk concerning the underestimation of cost can be then computed, if the relationship between size and cost is known.

Finally, being the estimates obtained via the Corrected NESMA method proportional to the estimates obtained via the original NESMA method, the confidence intervals for the Corrected NESMA method can be easily converted into confidence intervals for the original NESMA method.

V. Practical Usage of Size Estimate Intervals

In this section, the practical utility of the proposed method for computing confidence intervals for size is illustrated via an example, concerning the most typical usage of functional size metrics, i.e., effort estimation.

Suppose that Jane, a software project manager, has to estimate the effort required for developing a new application.

For effort estimation, she is using a model, shown in Figure 11, which estimates effort based on the size of the software to be developed. Note that the model shown in Figure 11 includes confidence intervals, which must not be confused with the confidence intervals discussed above: these are the confidence intervals embedded in the *effort* estimation model. That is, assuming that the provided size measure represents exactly the amount of software to be developed, the effort estimation model shows that the required development effort can vary because of many reasons, not connected with size: e.g., the characteristics of non functional requirements, the adopted process, the characteristics of developers, etc. Specifically, minimum and maximum effort values are provided, corresponding to some confidence level.

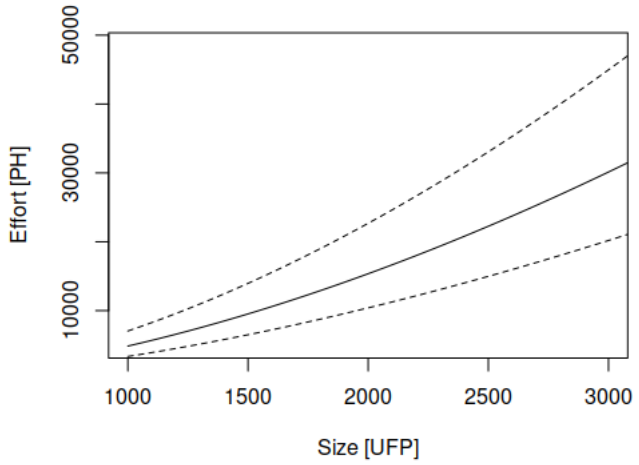


Fig. 11. An effort model with confidence intervals.

Jane, who read this paper, estimated the size of the application to be developed using the corrected NESMA method, and obtained that the estimated size is 2000 UFP. According to the model, developing an application of that size will likely take 15350 PH. With the confidence level being used, the effort will be in the [10393, 22669] PH range, as shown in Figure 12.

Now, Jane computes the confidence interval for the estimated size, using the procedure described in Section III-C. Since the size of the application is around 2000 UFP, Jane decides to use her company's data concerning projects in the [1000, 4000] UFP range to compute the confidence interval (this example uses ISBSG data; that is, for illustration purposes, we assume that Jane's company data are identical to ISBSG data). In this way, she finds that at a 0.75 confidence level the size of the application to be developed is in the [1760, 2248] UFP range.

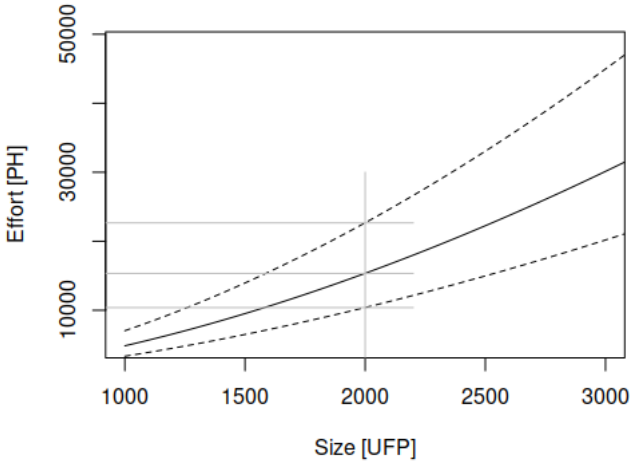


Fig. 12. Estimated effort based on the model and on the NESMA estimation of size.

With these values, Jane can now recompute the estimated effort. In the optimal case, i.e., when the effort model is given by the lower line in Figure 11 and the size of the application is 1760 UFP, the estimated effort is 8243 PH. In the worst case, i.e., when the effort model is given by the upper line in Figure 11 and the size of the application is 2248 UFP, the estimated effort is 27595 PH. The computations are illustrated graphically in Figure 13.

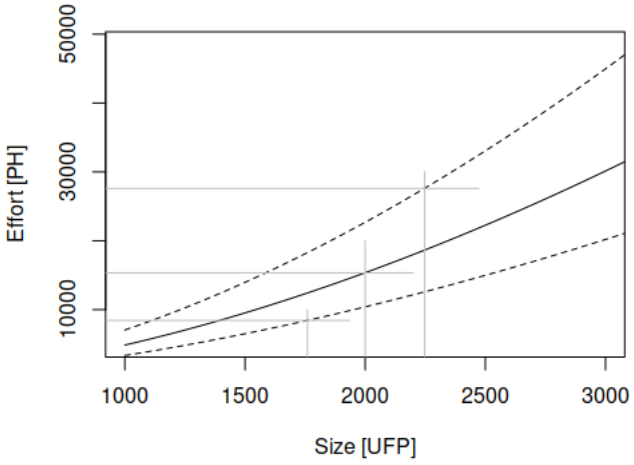


Fig. 13. Estimated effort considering both the confidence intervals of the model and those of the size estimate.

In conclusion, Jane finds out that a project that, according to 'one shot' estimation (using the likely effort model and the likely size estimate) requires 15350 PH, could instead require 27595 PH, i.e., 180% the one shot effort estimate (!) or 8243 PH, i.e., only 54% of the one shot estimate. This knowledge will allow Jane to devise a proper risk management strategy.

VI. Threats to validity

The proposed approach is empirical. In fact, the context itself suggests that a strong theoretical basis is not very

relevant. The definition of the NESMA estimated method itself has no theoretically strong basis: the method is based on the simple hypothesis that, on average, data have low complexity (in FPA terms) and transactions have mid complexity. So, we looked for reasonable confidence intervals, although these intervals are not statistically linked to confidence levels in a rigorous way.

Another typical concern in this kind of studies is the generalizability of results outside the scope and context of the analyzed dataset. We replicated the study with two datasets that are quite representative of real-life projects (the ISBSG dataset is deemed the standard benchmark among the community, and it includes data from several application domains, while the Chinese dataset collects data from a large set of banking and financial software projects). Therefore, our results may be representative of a fairly comprehensive situation. The general result we got is that the amount by which the NESMA method underestimates depends on the considered dataset; similarly, the confidence interval depends on the dataset. At any rate, it is worth underlying that while the numeric results we obtained are not applicable to datasets from different organizations, the proposed method is generally applicable as-is, in any context, provided that representative data are available.

VII. Related Work

Measures for early software estimation were conceived since the last decades [22]–[24]. The present study aims to advance this field by providing statistical foundations to some of these measures, by using confidence intervals where approaches not based on probability distributions were adopted. For example, the "Early & Quick Function Point" (EQFP) method [25] estimates an error of $\pm 10\%$ of the real size of software, for most of the times, but fails to indicate a more robust indicator of this estimate, such as a confidence interval. Several other early estimation methods were proposed: Table V lists the most popular ones.

TABLE V
Early estimation methods: definitions and evaluations

Method name	Definition	Used functions	Weight	Evaluation
NESMA indicative	[26] [27]	data	fixed	[4] [21], [28]–[31] [11]
NESMA estimated	[26] [27]	all functions	fixed	[4] [21], [28]–[31] [11]
Early & Quick FP	[24] [32] [25]	all functions	statistics	[11] [33]
simplified FP (sFP)	[34]	all functions	fixed	[11]
ISBSG average weights	[35]	all functions	statistics	[11]
SiFP	[36]	data and trans.	statistics	[13] [15]

Recently, comparisons based on the accuracy of the NESMA estimated (alias HLFPA) method and statistical modelling methods were carried out in order to assess whether standard measures fail in underestimating or overestimating software size [18].

A survey [37] reports how machine learning techniques were used for software development effort estimation, reporting accuracy as a comparison criterion for all the methods analysed. To the best of our knowledge, confidence intervals are overlooked as robust indicators of the estimates done in software size. In this respect, this study aims to emphasize the importance of providing robust indicators for a more reliable comparison and precision of reporting.

VIII. Conclusion

The "NESMA estimated" method was proposed to estimate the functional size of software (expressed in IFPUG Function Points). The NESMA method assigns fixed weights to base functional components (i.e., ILF, EIF, EI, EO and EQ), so that it is not necessary to analyze in depth every logic data file or transaction. This makes the method both easier and faster, and applicable when the details needed to characterize and weight base functional components are not yet available.

Previous studies showed that the NESMA method is sufficiently accurate to be used in practice. However, it has two possibly relevant limitations: 1) it tends to underestimate the "real" (i.e., as obtained via the IFPUG FPA process) size of software, and 2) it yields a single estimate, with no confidence intervals. Both these characteristics can be be problematic for software project managers. In fact, planning a project based on underestimated size and, consequently, on underestimated effort estimates usually leads to unrealistic plans. Besides, getting a confidence interval for size estimates allows for evaluating the risks connected with imprecise size estimates.

In this paper, we have proposed a correction for the estimates yielded by the NESMA method, to avoid underestimation, and a procedure to compute the confidence interval. Both these contributions are expected to make project managers' life easier.

It is important to remark that both the amount by which the NESMA method underestimates and the confidence intervals depend on the considered dataset. Hence, it is quite advisable that organizations that want to use the proposed techniques do so with their own data, which are expected to represent well the organization's projects.

Acknowledgment

The work reported here was partly supported by Fondo per la Ricerca di Ateneo, Università degli Studi dell'Insubria.

References

[1] L. Lavazza, A. Locoro, and R. Meli, "Estimating functional size of software with confidence intervals," in Proceedings of SOFTENG 2023: The Ninth International Conference on Advances and Trends in Software Engineering, 2023, pp. 14–19.

[2] A. J. Albrecht, "Measuring application development productivity," in Proceedings of the joint SHARE/GUIDE/IBM application development symposium, vol. 10, 1979, pp. 83–92.

[3] International Function Point Users Group (IFPUG), "Function point counting practices manual, release 4.3.1," 2010.

[4] H. van Heeringen, E. van Gorp, and T. Prins, "Functional size measurement-accuracy versus costs–is it really worth it?" in Software Measurement European Forum (SMEF), 2009.

[5] nesma, "nesma site," https://nesma.org/ [retrieved: March, 2023].

[6] A. Timp, "uTip – Early Function Point Analysis and Consistent Cost Estimating," 2015, uTip # 03 – (version # 1.0 2015/07/01).

[7] L. Lavazza, "On the effort required by function point measurement phases," International Journal on Advances in Software, vol. 10, no. 1 & 2, 2017, pp. 108–120.

[8] International Standardization Organization (ISO), "ISO/IEC 20926: 2003, Software engineering – IFPUG 4.1 Unadjusted functional size measurement method – Counting Practices Manual," 2003.

[9] IFPUG, "Simple Function Point (SFP) Counting Practices Manual Release 2.1," 2021.

[10] nesma, "Early Function Point Analysis," https://nesma.org/themes/sizing/function-point-analysis/early-function-point-counting/ [retrieved: March, 2023].

[11] L. Lavazza and G. Liu, "An empirical evaluation of simplified function point measurement processes," Journal on Advances in Software, vol. 6, no. 1& 2, 2013, pp. 1–13.

[12] International Software Benchmarking Standards Group, ""Worldwide Software Development: The Benchmark, release 11," ISBSG, 2009.

[13] L. Lavazza and R. Meli, "An evaluation of simple function point as a replacement of IFPUG function point," in IWSM–MENSURA 2014. IEEE, 2014, pp. 196–206.

[14] L. Lavazza, S. Morasca, and D. Tosi, "An empirical study on the effect of programming languages on productivity," in Proceedings of the 31st Annual ACM Symposium on Applied Computing, 2016, pp. 1434–1439.

[15] F. Ferrucci, C. Gravino, and L. Lavazza, "Simple function points for effort estimation: a further assessment," in 31st Annual ACM Symposium on Applied Computing. ACM, 2016, pp. 1428–1433.

[16] L. Lavazza, S. Morasca, and D. Tosi, "An empirical study on the factors affecting software development productivity," E-Informatica Software Engineering Journal, vol. 12, no. 1, 2018, pp. 27–49.

[17] L. Lavazza, G. Liu, and R. Meli, "Productivity of software enhancement projects: an empirical study." in IWSM-Mensura, 2020, pp. 1–15.

[18] G. Liu and L. Lavazza, "Early and quick function points analysis: Evaluations and proposals," Journal of Systems and Software, vol. 174, 2021, p. 110888.

[19] L. Lavazza, A. Locoro, G. Liu, and R. Meli, "Using locally weighted regression to estimate the functional size of software: an empirical study," International Journal on Advances in Software, vol. 15, no. 3-4, 2022, pp. 211–223.

[20] ——, "Estimating software functional size via machine learning," ACM Transactions on Software Engineering and Methodology, 2023.

[21] L. Lavazza and G. Liu, "An Empirical Evaluation of the Accuracy of NESMA Function Points Estimates," in ICSEA, 2019, pp. 24–29.

[22] D. B. Bock and R. Klepper, "FP-S: a simplified function point counting method," Journal of Systems and Software, vol. 18, no. 3, 1992, pp. 245–254.

[23] G. Horgan, S. Khaddaj, and P. Forte, "Construction of an FPA-type metric for early lifecycle estimation," Information and Software Technology, vol. 40, no. 8, 1998, pp. 409–415.

[24] L. Santillo, M. Conte, and R. Meli, "Early & Quick Function Point: sizing more with less," in 11th IEEE International Software Metrics Symposium (METRICS'05). IEEE, 2005, pp. 41–41.

[25] DPO, "Early & Quick Function Points Reference Manual - IFPUG version," DPO, Roma, Italy, Tech. Rep. EQ&FP-IFPUG-31-RM-11-EN-P, April 2012.

[26] NESMA–the Netherlands Software Metrics Association, "Definitions and counting guidelines for the application of function point analysis. NESMA Functional Size Measurement method compliant to ISO/IEC 24570 version 2.1," 2004.

[27] International Standards Organisation, "ISO/IEC 24570:2005 – Software Engineering – NESMA functional size measurement method version 2.1 – definitions and counting guidelines for the application of Function Point Analysis," 2005.

[28] F. G. Wilkie, I. R. McChesney, P. Morrow, C. Tuxworth, and N. Lester, "The value of software sizing," Information and Software Technology, vol. 53, no. 11, 2011, pp. 1236–1249.

[29] J. Popović and D. Bojić, "A comparative evaluation of effort estimation methods in the software life cycle," Computer Science and Information Systems, vol. 9, no. 1, 2012, pp. 455–484.

[30] P. Morrow, F. G. Wilkie, and I. McChesney, "Function point analysis using nesma: simplifying the sizing without simplifying the size," Software Quality Journal, vol. 22, no. 4, 2014, pp. 611–660.

[31] S. Di Martino, F. Ferrucci, C. Gravino, and F. Sarro, "Assessing the effectiveness of approximate functional sizing approaches for effort estimation," Information and Software Technology, vol. 123, July 2020.

[32] T. Iorio, R. Meli, and F. Perna, "Early&quick function points® v3. 0: enhancements for a publicly available method," in SMEF, 2007, pp. 179–198.

[33] R. Meli, "Early & quick function point method-an empirical validation experiment," in Int. Conf. on Advances and Trends in Software Engineering, Barcelona, Spain, 2015, pp. 14–22.

[34] L. Bernstein and C. M. Yuhas, Trustworthy systems through quantitative software engineering. John Wiley & Sons, 2005, vol. 1.

[35] R. Meli and L. Santillo, "Function point estimation methods: A comparative overview," in FESMA, vol. 99. Citeseer, 1999, pp. 6–8.

[36] R. Meli, "Simple function point: a new functional size measurement method fully compliant with IFPUG 4.x," in Software Measurement European Forum, 2011, pp. 145–152.

[37] M. N. Mahdi, M. H. Mohamed Zabil, A. R. Ahmad, R. Ismail, Y. Yusoff, L. K. Cheng, M. S. B. M. Azmi, H. Natiq, and H. Happala Naidu, "Software project management using machine learning technique—a review," Applied Sciences, vol. 11, no. 11, 2021, p. 5183.

Implementing the Typed Graph Data Model Using Relational Database Technology

Malcolm Crowe
Emeritus Professor, Computing Science
University of the West of Scotland
Paisley, United Kingdom
Email: Malcolm.Crowe@uws.ac.uk

Fritz Laux
Emeritus Professor, Business Computing
Reutlingen University
Reutlingen, Germany
Email: Fritz.Laux@reutlingen-university.de

***Abstract*—Recent standardization work for database languages has reflected the growing use of typed graph models (TGM) in application development. Such data models are frequently only used early in the design process, and not reflected directly in underlying physical database. In previous work, we have added support to a relational database management system (RDBMS) with role-based structures to ensure that relevant data models are not separately declared in each application but are an important part of the database implementation. In this work, we implement this approach for the TGM: the resulting database implementation is novel in retaining the best features of the graph-based and relational database technologies.**

Keywords—typed graph model; graph schema; relational database; implementation; information integration.

I. INTRODUCTION

The work in this paper was signaled in a conference presentation [1] in early 2023 and reflects ongoing work in the standardization community to create standards for graph databases. This has already led to the adoption of a new chapter in the International Standards Organization (ISO) standard 9075 [2] for property graph queries, and a draft international standard (DIS) on Graph Query Language (GQL) is now expected in early 2024.

Many data models assist in the development of software, such as the Unified Modeling Language (UML) [3][4], entity frameworks, and persistence architectures. During such early conceptual model building, incremental and interactive exploration can be helpful [5] as fully automated integration tools may combine things in an inappropriate way, and the use of data types [6] can help to ensure that semantic information is included not merely in the model, but also in the final database. In this short paper we report on such an implementation of the Typed Graph Model (TGM), using metadata in a relational database management system (RDBMS) [7], partly inspired by recent developments in the PostgreSQL community [8]. Some recent database management systems (DBMS) have included metadata in the relational model to form a bridge with the physical database, so that the data model can be enforced across all applications for a single database. In this work, we provide a mechanism for integrating the graphical data model in the physical RDBMS.

As with the original relational model, the TGM has a rigorous mathematical foundation as an instance of a Graph Schema.

The plan of this paper is to review the TGM in Section II, and discuss the implementation details in Section III. Section IV presents an illustrative example, and Section V provides some conclusions.

II. THE TYPED GRAPH MODEL AND INFORMATION INTEGRATION

We will construct a TGM for a database by declaring instances of nodes and edges as an alternative to specifying tables of nodes and edges.

A. Typed Graphs Formalism

In this section we review the informal definition of the TGM from [2], using small letters for elements (nodes, edges, data types, etc.) and capital letters for sets of elements. Sets of sets are printed as bold capital letters. A typical example would be $n \in N \in \mathbf{N} \subseteq \wp(N)$, where N is any set and $\wp(N)$ is the power-set of N.

Let T denote a set of simple or structured (complex) data types. A data type $t:=(l,d) \in T$ has a name l and a definition d. Examples of simple (predefined) types are (int,$\mathbb{Z}$), ($char$,$ASCII$), (%,[0..100]), etc. It is also possible to define complex data types like an order line (*OrderLine*, (*posNo*, *partNo*, *partDescription*, *quantity*)). The components need to be identified in T, e. g., (*posNo*,$int>0$). Recursion is allowed as long as the defined structure has a finite number of components.

The UML-notation was chosen as graphical representation for nodes and include the properties as attributes including their data types. Labels are written in the top compartment of the UML-class. Edges of the TGS are represented by UML associations. For the label and properties of an edge we use the UML-association class, which has the same rendering as an ordinary class, but its existence depends on an association (edge), which is indicated by a dotted line from the association class to the edge. This not only allows to label an edge but to define user defined edge types. The correspondence between the UML notation and the TGS definition is shown in Table I.

Definition 1 (Typed Graph Schema, TGS) *A typed graph schema is a tuple* $TGS=(N_S,E_S,\varrho,T,\tau,C)$

where:

- *N_S is the set of named (labeled) objects (nodes) n with properties of data type t:=(l,d)∈T, where l is the label and d the data type definition.*
- *E_S is the set of named (labeled) edges e with a structured property p:=(l,d)∈T, where l is the label and d the data type definition.*
- *ϱ is a function that associates each edge e to a pair of object sets (O,A), i. e., $\varrho(e):=(O_e,A_e)$ with $O_e,A_e \in \wp(N_S)$. O_e is called the* tail *and A_e is called the* head *of an edge e.*
- *τ is a function that assigns for each node n of an edge e a pair of positive integers (i_n,k_n), i. e., $\tau_e(n):=(i_n,k_n)$ with $i_n \in N_0$ and $k_n \in N$. The function τ defines the min-max multiplicity of an edge connection. If the min-value i_n is 0 then the connection is optional.*
- *C is a set of integrity constraints, which the graph database must obey.*

The notation for defining data types T, which are used for node types N_S and edge types E_S, can be freely chosen: and in this implementation SQL will be used for identifiers and expressions, together with a strongly typed relational database engine. The integrity constraints *C* restrict the model beyond the structural limitations of the multiplicity τ of edge connections. Typical constraints in *C* are semantic restrictions of the content of an instance graph. For instance, in an order processing graph-database a constraint should require that an "order"-node o should have at least one "order-detail" node ol connected by an edge labelled "belongs_to" (see example order GDB in Table II.)

Definition 2 (Typed Graph Model) *A typed graph Model is a tuple TGM=(N,E,TGS,φ) where:*

- *N is the set of named (labeled) nodes n with data types from N_S of schema TGS.*
- *E is the set of named (labeled) edges e with properties of types from E_S of schema TGS.*
- *TGS is a typed graph schema as defined above..*
- *φ is a homomorphism that maps each node n and edge e of TGM to the corresponding type element of TGS, formally:*

$$\varphi: TGM \rightarrow TGS$$
$$n \mapsto \varphi(n) := n_S (\in N_S)$$
$$e \mapsto \varphi(e) := e_S (\in E_S)$$

The fact that φ maps each element (node or edge) to exactly one data type implies that each element of the graph model has a well-defined data type. The homomorphism is structure preserving. This means that the cardinality of the edge types is enforced, too. In our Pyrrho implementation, the declaration of nodes and edge of the TGM develops the associated TGS incrementally including the development of the implied type system *T*. Data type and constraint checking is applied for all nodes and edges before any insert, update, or delete action can be committed.

B. The Data Integration Process

The full benefit of information integration requires the integration of source data with their full semantics. We believe a key success factor is to model the sources and target information as accurately as possible. The expressive power and flexibility of the TGM allows precise description of the meta-data of the sources and target in the same model, which simplifies the matching and mapping of the sources to the target. The tasks of the data integration process are:

1) model sources as TGS S_i (i = 1, 2, ..., n)
2) model target schema *T* as TGS *G*
3) match and map sources S_i with TGS *G*
4) check and improve quality
5) convert TGS G back to T again

Steps 3 and 4 can occur together in an interactive process once the basic model has been outlined. Such a process is crucial for Enterprise Information Integration (EII) and other data integration projects, which demand highly accurate information quality, which can be further improved with the use of different mappings.

To start the process, it may be necessary to collect structure and type information from a data expert or from additional information. Where sources are databases, the rigid structures provide a good starting point. Otherwise, the relevant data must first be identified together with its meta-data if available. This includes coding and names for the data items. The measure units and other meta-data provided by the data owner are used to adjust all measures to the same scale. The paper of Laux [6] gives some examples how to transform relational, object oriented, and XML-schemata into a TGS.

If the source is unstructured or semi-structured, e.g., documents or XML/HTML data, concepts and mechanisms from Information Retrieval (IR) and statistical analysis may help to identify some implicit structure or identify outliers and other susceptible data. If the data are self-describing (JSON, key-value pairs, or XML) linguistic matching can be applied with additional help from a thesaurus or ontology. Nevertheless, it is advisable to validate the matching with instance data or an information expert.

The use of hyper-nodes $n \in N_S$ and hyper-edges $e \in E_S$ instead of simple nodes resp. edges allow to group nodes and edges to higher abstracted complex model aggregates. This is particularly useful to keep large models clearly represented and manageable. Each sub-graph can be rendered as a hyper-node. If the division is disjoint these hyper-nodes are connected via hyper-edges forming a higher abstraction level schema.

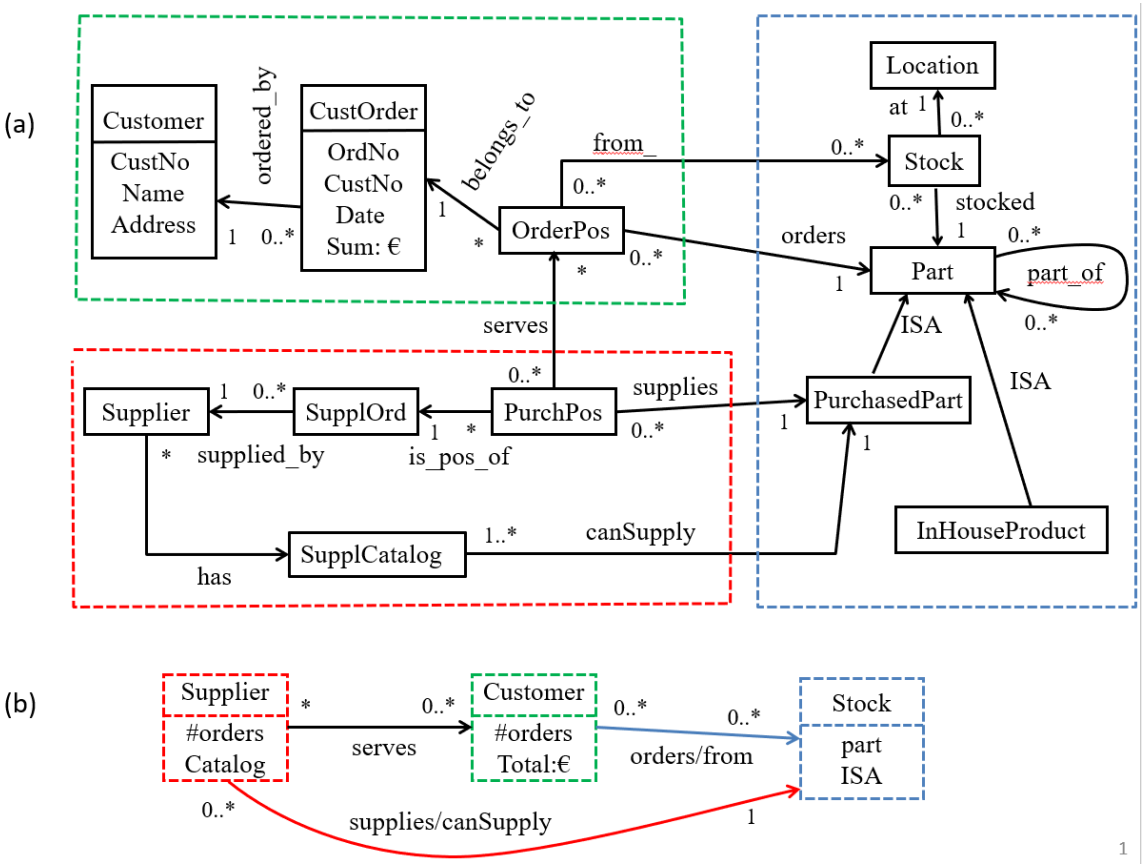


Figure 1. Example TGM of a commercial enterprise showing two levels of detail

We present two possible TGS abstraction levels for a single enterprise in UML notation in Figure 1. The dashed green line in part (a) encompasses the Customer data, comprising Customer master data, customers' orders, and order positions. In part (b) this information is concentrated in one node type named "Customer". The Supplier data side (red dashed line) is modelled in the same manner. The stock management data show in the detailed part (a) the bill of material (BOM) which is modelled as a recursive edge "Part of" on the parts node. This is no longer explicitly visible in the aggregated part (b). This hidden information should be part of the now complex property "part" of the Stock node.

This little example demonstrates already the flexibility of the model in terms of detail and abstraction. We discuss this example in some detail in Section IV.

III. Implementation in the Relational Database Schema

The implementation of a typed graph modelling system can build on the user-defined type mechanism of an RDBMS. Node and edge types should have special columns: for node types, there is an automatic primary key with default name ID, and edge types also automatic foreign keys for their source and destination nodes, that are referred to here by their default names LEAVING and ARRIVING, and these should have automated support from the RDBMS. It should be possible to convert between standard types and node/edge types and rearrange subtype relationships. These tables can be equipped with indexes, constraints, and triggers in the normal ways.

Then, if every node type or edge type corresponds to a single base table containing the instances of that type, one way to build a graph is to insert rows in these tables. But a satisfactory implementation needs to simplify the tasks of graph definition and searching. Most implementations add CREATE and MATCH statements, which we describe next, and indicate how they can be implemented in the RDBMS.

A. Graph-oriented Syntax Added to SQL

The typical syntax for CREATE sketches nodes and edges using additional arrow-like tokens, for example:

```
[CREATE (:Person {name:'Fred Smith'})<-
[:Child]-(a:Person {name:'Peter Smith'}),
(a)-[:Child]->(b:Person {name:'Mary Smith'})
-[:Child]->(:Person {name:'Lee Smith'}),
(b)-[:Child]->(:Person {name:'Bill Smith'})]
```

Without any further declarations, this builds a graph with nodes for Person and edges for Child, as in Figure 2.

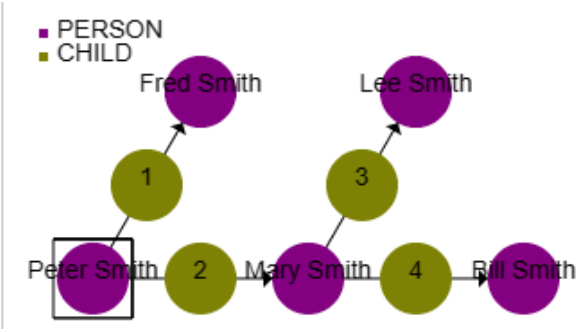


Figure 2. Browser output for web address http://localhost:8180/ps/PS/PERSON/NAME='Peter Smith'?NODE

There is already a standard abstract syntax [9][10], that can be represented as:

CREATE Graph {','Graph} [THEN Statement].
Graph = Node Path {',' Node Path } .
Path = { Edge Node } .
Node = '(' GraphItem ')' .
Edge = '-[' GraphItem ']->' | '<-[' GraphItem ']-'.
GraphItem = [id | *Node*_Value] [GraphLabel] [Document] .
GraphLabel = ':' (id | *Label*_Value) [GraphLabel] .

In this syntax, the strings enclosed in single quotes are tokens, including several new token types for the TGM. In corresponding source input, unquoted strings are used for case-insensitive identifiers and double quoted strings for case-sensitive identifiers, possibly containing other Unicode characters. As usual in SQL, string constants in input will be single quoted, and doc is a JSON-like structure providing a set of properties and value expressions, possibly including metadata definitions for ranges and multiplicity.

Nodes and edges and new node types and edge types can be introduced with this syntax. The database engine constructs a base table for each distinct label, with columns sufficient to represent the associated properties. These database base tables for node types (or edge types) contain a single row for each node (resp. edge) including node references. They can be equipped with indexes, constraints, and triggers in the normal ways.

To the normal SQL DML, we add the syntax for the MATCH query, which has a similar syntax, except that it may contain unbound identifiers for nodes and edges, their labels and/or their properties.

MatchStatement = MATCH Match {',' Match} [WhereClause] [Statement] [THEN Statements END].
Match = (MatchMode [id '='] MatchNode) {'|' Match}.

The first part of the MATCH clause has an optional MatchMode (see below) and one or more graph expressions, which in simple cases appear to have the same form as in the CREATE statement.
MatchNode = '(' MatchItem ')' {(MatchEdge|MatchPath) MatchNode}.
MatchEdge = '-[' MatchItem '->' | '<-' MatchItem ']-' .
MatchItem = [id | *Node*_Value] [GraphLabel] [Document | WhereClause] .

In all cases, the execution of the MATCH proceeds directly on the tables, without needing auxiliary SQL statements. The MATCH algorithm proceeds along the node expressions, matching more and more of its nodes and edges with those in the database by assigning values to the unbound identifiers. If we cannot progress to the next part of the MATCH clause, we backtrack by undoing the last binding and taking an alternative value. If the processing reaches the end of the MATCH statement, the set of bindings contributes a row in the default result, subject to the optional WHERE condition.

In this way, the MATCH statement can be used (a) as in Prolog, to verify that a particular graph fragment exists in the database, (b) to display the bindings resulting from the process of matching a set of fragments with the database, (c) to display a set of values computed from such a list of bindings, or (d) to perform a sequence of actions for each binding found. In case (d) no results are displayed, as the MATCH statement has been employed for its side effects. These could include further CREATE, MATCH or other SQL statements, or assignment statements updating fields referenced in the current bindings.

Following the forthcoming GQL standard, repeating patterns are supported by the MATCH statement (see [9]):

MatchPath = '[' Match ']' MatchQuantifier .
MatchQuantifier = '?' | '*' | '+' | '{' int , [int] '}' .
MatchMode = [TRAIL|ACYCLIC| SIMPLE] [SHORTEST |ALL|ANY] .

The MatchMode controls how repetitions of path patterns are managed in the graph matching mechanism. A MatchPath creates lists of values of bound identifiers in its Match. By default, binding rows that have already occurred in the match are ignored, and paths that have already been listed in a quantified graph are not followed. The MatchMode modifies this default behaviour: TRAIL omits paths where an edge occurs more than once, ACYCLIC omits paths where a node occurs more than once, SIMPLE looks for a simple cycle. The last three options apply to MatchStatements that do not use the comma operator, and select the shortest match, all matches or an arbitrary match.

The implementation of the matching algorithm uses continuations to control the backtracking behavior. Continuations are constructed as the match proceeds and represent the rest of the matching expression.

The MATCH statement can be used in two ways. The first is make the dependent Statement a RETURN statement that contributes a row to a result set for each successful binding of the unbound identifiers in the MATCH, for example,

```
E:\PyrrhoDB70\Pyrrho>pyrrhocmd ps
SQL> [CREATE (a:Person {name:'Fred Smith'})<-[:Child]-(b:Person {name:'Peter Smith'}),
> (a)-[:Child]->(c:Person {name:'Mary Smith'})
> -[:Child]->(:Person {name:'Lee Smith'}),
> (c)-[:Child]->(:Person {name:'Bill Smith'})]
SQL> MATCH ({name:'Peter Smith'}) [()-[:Child]->()]+ (x) RETURN x.name
|----------|
|NAME      |
|----------|
|Lee Smith |
|Bill Smith|
|Mary Smith|
|Fred Smith|
|----------|
SQL> MATCH ({name:'Peter Smith'}) [(p)-[:Child]->()]+ ({name:x})
|----------------------------------------------------------------------------------------------------|----------|
|P                                                                                                   |X         |
|----------------------------------------------------------------------------------------------------|----------|
|ARRAY[PERSON(ID=2,NAME=Peter Smith),PERSON(ID=1,NAME=Fred Smith),PERSON(ID=3,NAME=Mary Smith)]|Lee Smith |
|ARRAY[PERSON(ID=2,NAME=Peter Smith),PERSON(ID=1,NAME=Fred Smith),PERSON(ID=3,NAME=Mary Smith)]|Bill Smith|
|ARRAY[PERSON(ID=2,NAME=Peter Smith),PERSON(ID=1,NAME=Fred Smith)]                                    |Mary Smith|
|ARRAY[PERSON(ID=2,NAME=Peter Smith)]                                                                |Fred Smith|
|----------------------------------------------------------------------------------------------------|----------|
SQL> alter table person add primary key(name)
SQL> alter table person drop id
SQL> create role ps
SQL> grant ps to "MALCOLM1\Malcolm"
SQL>
```

Figure 3. This shows the commands needed in our implementation to create a new database containing the example graph data, some simple graph-oriented queries, and some steps to develop the model and make it available to the network

```
MATCH ({name:'Peter Smith'}) [()-[:Child]->()]+
(x) RETURN x.name
```

will yield a list of the descendants of Peter Smith.

Without using RETURN or any dependent statements, the result of a MATCH statement is the list of bindings. The above example has two columns, one for each of the unbound identifiers p and x, but p will be an array with an element for each iteration of the pattern.

The results are shown in Figure 3, which also shows all of the statements needed in our implementation to build and display this small example, including two lines for replacing the default primary key ID. A feature of the implementation described in this paper is the lack of structural clutter.

In sections B and C, we continue this small example with two further steps, to display the contents as a graph, and to show how the relational database directly supports object-oriented application programming for such graphical data.

B. Graph versus Relation

The nodes and edges contained in the database combine to form a set of disjoint graphs that is initially empty. Adding a node to the database adds a new entry to this set. When an edge is added, either the two endpoints are in the same graph, or else the edge will connect two previously disjoint graphs. If each graph in the set is identified by a representative node (such as the one with the lowest uid) and maintains a list of the nodes and edges it contains, it is easy to manage the set of graphs as data is added to the database.

If an edge is removed, the graph containing it might now be in at most two pieces: the simplest algorithm removes it from the set and adds its nodes and edges back in.

It is helpful if the RDBMS is extended to provide a graphical display as in Figure 2 above. In our work the RDBMS provides a simple HTTP service, so that once the database has given appropriate authorization an ordinary web access will display the graph in a browser. Selection of a node with the mouse displays its properties.

The database with its added graph information can be used directly in ordinary database application processing, with the advantage of being able to perform graph-oriented querying and graph-oriented stored procedures. The normal processing of the database engine naturally enforces the type requirements of the model, and also enforces any constraints specified in graph-oriented metadata. The nodes and edges are rows in ordinary tables that can be accessed and refined using normal SQL statements. In particular, using the usual dotted syntax, properties can be SET and updated, and can be removed by being set to NULL.

C. Database Design by Example

From the above description of the CREATE statement, we can see that this mechanism allows first versions of types and instances to be developed together, with minimal schema indications. The MATCH statement allows extension of the design by retrieving instances and creating related nodes and edges.

If example nodes and edges are created, the DBMS creates suitable node and edge types, modifying these if additional properties receive values in later examples.

Since transactions are supported, tentative examples can be explored and rolled back or committed. Alter statements can change names, enhance property types and modify subtype relationships, and the SQL Cast function can be used to parse the string representation of a structure value. The usual restrict/cascade actions are available, and node and edge types can have additional constraints, triggers, and methods. As each node and edge type has an associated base table in the database, the result of this process is a relational database that is immediately usable.

As the TGM is developed and merged with other graphical data, conflicts will be detected and diagnostics will help to identify any obstacles to integrating a new part of the model, so that the model as developed to that point can be refined. The SQL ALTER TABLE and ALTER TYPE statements, together with a metadata syntax, allow major changes to the model to be performed automatically, e.g., to enforce expectations on the data.

It is important that all such major changes, indeed all cascades and trigger side effects, are validated as part of the transaction commit process, so that the database is not left in an inconsistent state as a result of a mistake or security exception. An example of such a cascade occurs where a graph has been created using the server's autokey mechanism for primary keys, and the analyst has identified a more suitable numeric or string-valued key. A single ALTER TABLE statement can install this as the new primary key and the change automatically propagates to the edge types that attach to the node type in question. The previous primary key remains as a unique key but can later be dropped without losing any information. Figure 3 shows this process, and its consequences are visible in Figures 2 and 4.

Other restructuring of node types can be performed with the help of the CAST function, which can be used to parse complex types from strings, array and set constructors, and UNNEST. Node and edge manipulations can also be performed by triggers and stored procedures.

The points covered in the above section already go a long way towards an integrated DBMS product that supports the TGM. The resulting TGM implementation inherits aspects such as transacted behavior, constraints, triggers, and stored procedures from the relational mechanisms, since Match and Create statements are implemented as Procedure Statements. The security model in the underlying RDBMS, with its users, roles, and grants of privileges also applies to the base tables and hence to the graphs. Node and edge types emerge as a special kind of structured type. It is thus a relatively simple matter to support view-mediated remote access and object-oriented entity management. Nodes and edges are entities and the same access and Multiple Version Concurrency Control (MVCC) models in our previous work [11] transfer with little trouble into the new features.

As the TGM is developed and merged with other graphical data, conflicts will be detected and diagnostics will help to identify any obstacles to integrating a new part of the model, so that the model as developed to that point can be refined.

It is natural to expect a user interface that displays a graphical version of the property graph. Figure 2 was generated by sending a link (see caption of Figure 2) to our implementation's HTTP service to draw a picture of a portion of a graph starting at a given node. Selection of a node or edge displays the properties of that node and links to redraw the graph starting at another node.

Our database server implementation has for years generated classes for C#, Python or Java applications corresponding to versioned database objects. Here this leads to object-oriented application programming, where node and edge types correspond to classes whose instances are nodes and edges. The Match and Create statements can be used (a) for SQL clients in commands and prepared statements, (b) in the generated C#, Java or Python and the widely used database connection methods ExecuteReader and ExecuteNonQuery, or (c) in JavaScript posted to the web service interface of the database server. In Figure 4 we show a portion of a C# application program to display the descendants of Peter Smith in the little example graph database discussed above.

The normal processing of the database engine naturally enforces the type requirements of the model, and also enforces a range of constraints specified in graph-oriented metadata. The nodes and edges are rows in ordinary tables that can be accessed and refined using normal SQL statements. In particular, using the usual dotted syntax, properties can be SET and updated, and can be removed by being set to NULL.

```
/// <summary>
/// EdgeType CHILDOF from Database ps, Role PS
/// PrimaryKey(ID)
/// ForeignKey, CascadeUpdate(PARENT) PERSON
/// ForeignKey, CascadeUpdate(CHILD) PERSON
/// </summary>
[EdgeType(164, 533)]
5 references
public class CHILDOF : Versioned
{
    [Identity]
    [Field(PyrrhoDbType.Integer)]
    [AutoKey]
    public Int64? ID;
    [Leaving]
    [Field(PyrrhoDbType.String)]
    public String? PARENT;
    [Arriving]
    [Field(PyrrhoDbType.String)]
    public String? CHILD;
    0 references
    public PERSON? PARENTis => conn?.FindOne<PERSON>(("NAME", PARENT));
    1 reference
    public PERSON? CHILDis => conn?.FindOne<PERSON>(("NAME", CHILD));
}
0 references
public class Demo
{
    static PyrrhoConnect? conn = null;
    2 references
    static List<PERSON> Descendants(PERSON p)
    {
        var ds = new List<PERSON>();
        if (p.ofPARENTs is CHILDOF[] ca)
            foreach (var c in ca)
                if (c.CHILDis is PERSON d)
                {
                    ds.Add(d);
                    ds.AddRange(Descendants(d));
                }
        return ds;
    }
    0 references
    static void Main()
    {
        conn = new PyrrhoConnect("Files=ps;Role=PS");
        conn.Open();
        try
        {
            // Get a list of all descendants of Pete Smith
            var pa = conn.FindWith<PERSON>(("NAME","Peter Smith"));
            if (pa.Length == 1)
                foreach (var c in Descendants(pa[0]))
                    Console.WriteLine(c.NAME);
        }
        catch (Exception ex)
```

Figure 4. A portion of a C# application program to find the descendants of Peter Smith in the example database above

IV. AN EXAMPLE

Examples for a graph structure usually choose social networks. We want to show that the TGM is equally suitable for Enterprise Resource Planning (ERP) and other business systems. As a non-trivial example, we have chosen a commercial enterprise which buys parts and products, resells the purchased products or assembles products from purchased parts and sells these value-added products. It does not develop and construct products from raw material but add some value to parts or assembles some products to form systems.

The data model shown above in Figure 1 is suitable for a customer-supplier ordering system and comprises 3 company divisions or departments: sales (green), stock (blue), and procurement (red). These are framed in Figure 1(a) with a green dashed line for sales data, with blue for

TABLE I. TGS CORRESPONDENCE WITH UML NOTATION

TGS	UML
$n \in N_S$	class
$e \in E_S$	association
$t = (l, d) \in T$	l = name of n resp. e; d = type of n resp. e
$\varrho\ (e)$	all ends of e
$\tau_e(n)$	(min,max)-cardinality of e at n
C	constraints in [ ] or { }

stock data, and red for procurement or purchase. The graph schema is visualized using UML notation which allows specifying the cardinality of the edges. The correspondence between the Typed Graph Schema (TGS) elements and the UML is shown in Table I.

The sales division needs to manage customer data and process the customer's orders. It consists of Customer nodes with properties CustNo, Name and Address. The Name and Address might as well be structured data types for first- and last name resp. street, ZIP code, and city. The CustOrder node mainly comprises OrdNo, the (redundant) CustNo, order date Date and the order total Sum in Euros. The CustOrder contains 1 to many order detail lines of OrderPos which consist at least of the order quantity as property. The order quantity itself is suppressed in the UML diagram to avoid overloading the picture. According to the semantics of the TGM the edge arrows signify the reading direction of the edge type. In the case of "belongs_to" the reading direction is from OrderPos to CustOrder.

All other necessary properties for an order line (e. g. partNo, PartName, UnitProce) could be determined by following the edges of the model to the Part, Stock, and CustOrder node. In Figure 1 (a) only the nodes Customer and CustOrder are showing exemplified properties. More properties are maintained in a real situation, e. g. planned delivery, shipping date, etc for a customer order. The same applies to all other nodes, e. g. unit and quantity discount for parts.

The procurement division is responsible for maintaining the supplier data and ordering of parts and products from them. It mirrors the sales model structurally and comprises supplier, the purchases (SupplOrd, PurchPos) and the supplier catalogue. Purchase- and Sales division have connections to the stock management.

Finally, the stock division comprises master data management and stock management. Master data management includes structural information about the parts in the form a Bill Of Materials (BOM). Stock management deals with adding parts to the stock and releaseing them from stock, The central node of the stock model is the Part node who distinguishes between purchased parts (PurchasedParts) and in-house products (InHouseProduct) modelled as subtypes of Part. We have a BOM structurally represented as a recursive edge "part_of" on the part nodes. The BOM forms a tree structure with the product at the top. The product is made up recursively of components (composed parts) and finally of single parts. The stock itself is represented as a node with properties like number of parts, reservations, and commissions. A stock node is linked to a part and a storage location. This allows knowing exactly which part is located at a certain location in the warehouse.

Figure 1 (b) gives a high level view on the scenario. Such kinds of abstractions are important for complex graphs in order to keep the model manageable. CASE tools that support zoom-in and zoom-out functions would be beneficial to assist the graph modelling.

The syntax of the above presented example ERP model will be presented in the following subsection. Multiline statements are enclosed in square brackets.

A. *Syntax of the ERP example*

First we start with the sales graph (green schema), followed by the supplier (red schema) and stock division (blue schema), and finally the three divisions are linked by the edge types "serves", "supplies", "canSupply", "orders", and "from".

The green schema is illustrated in Figure 5 below, and the declarations:

```
// sales division
[CREATE
(a:Customer {CustNo:1001, Name:'Adam', Address:'122,
Nutley Terrace, London, ST 7UR, GB'} ), // Customer
(b:Customer {CustNo:1002, Name:'Brian', Address:'45,
Belsize Square, London, ST 7UR, GB'} ),
 // …
(f:Customer {CustNo:1006, Name:'Eddy', Address:'72,
Ibrox Street, Glasgow, G51 1AA, UK'} ), // customer
without order
 (o1:CustOrder {OrdNo:2001, CustNo:1001,
Datum:DATE'2023-03-22', SummE:211.00} ), //
CustOrder
(o2:CustOrder {OrdNo:2002, CustNo:1002,
Datum:DATE'2023-03-22', SummE:24.00} ),
// …
(o8:CustOrder {OrdNo:2008, CustNo:1002,
Datum:DATE'2023-04-24', SummE:808.00} ),
 (op1:OrderPos {Quantity:4, Unit:'piece'} ), //
OrdPos
(op2:OrderPos {Quantity:4, Unit:'litre'} ),
// …
(op18:OrderPos {Quantity:10, Unit:'piece'} ),
(a)<-[:ORDERED_BY]-(o1), // each order was ordered
by exactly 1 customer
(a)<-[:ORDERED_BY]-(o6),
(a)<-[:ORDERED_BY]-(o7),
(b)<-[:ORDERED_BY]-(o2),
//…
(o1)<-[:BELONGS_TO]-(op1), // each orderPos belongs
to exactly 1 order
(o2)<-[:BELONGS_TO]-(op2),

// …
(o8)<-[:BELONGS_TO]-(op9), // and an order has at
least 1 orderPos
(o8)<-[:BELONGS_TO]-(op10),
(o1)<-[:BELONGS_TO]-(op11),
// …
   (o8)<-[:BELONGS_TO]-(op18)]
```

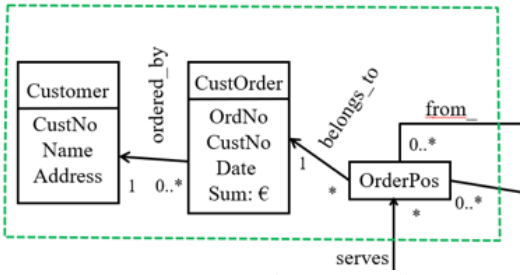


Figure 5. The customer section of the database (from Figure 1)

We continue with the supplier division, illustrated in Figure 6 below. Here are sample declarations for this section:

```
// supplier division
[ CREATE
(a:Supplier {SupplNo:101, Name:'Rawside Furniture',
Address:'58 City Rd, London , EC1Y 2AL, UK'} ),
(b:Supplier {SupplNo:102, Name:'Andreas Stihl Ltd',
Address:'Stihl House Stanhope Road, GU 15 3 YT,
Camberley Surrey, GB'} ),
// …
// SupplOrd
(o1:SupplOrd {OrdNo:2001, SupplNo:101,
Datum:DATE'2023-01-11', "Sum€":260.00} ),
(o2:SupplOrd {OrdNo:2002, SupplNo:102,
Datum:DATE'2023-02-22', "Sum€":2405.00} ),
// …
// OrdPos purchase details
(op1:PurchOrd {PosNo:1, Quantity:4, Unit:'piece'} ),
(op2:PurchOrd {PosNo:1, Quantity:4, Unit:'litre'} ),
// …
// (Supplier)<-[:SUPPLIED_BY]-(SupplOrd)
(a)<-[:SUPPLIED_BY]-(o1), // each order was ordered
by exactly 1 Supplier
(a)<-[:SUPPLIED_BY]-(o4),
// …
// (SupplOrd)<-[:IS_POS_OF]-(OrdPos)
(o1)<-[:IS_POS_OF]-(op1), // each PurchPos belongs
to exactly 1 order
(o2)<-[:IS_POS_OF]-(op2),
// …
(o1)<-[:IS_POS_OF]-(op7), // and an order has at
least 1 PurchPos
(o1)<-[:IS_POS_OF]-(op8),
(o1)<-[:IS_POS_OF]-(op9),
//…
// SupplCatalog
(sc11:SupplCatalog {SupplNo:101, SPartNo:'sp1',
description:'Hammer handle, Wood (ash), Weight:100
g', unit:'piece', unitPrice:2.00}), //P15
(sc12:SupplCatalog {SupplNo:101,SPartNo:'sp2',
description:'Tabletop, Wood (oak), Color:brown,
Size:80w x120l cm', unit:'piece', unitPrice:40.00}),
//P16
// …
(sc46:SupplCatalog {SupplNo:104, SPartNo:'sp6',
description:'Shelf spruce, Color: white, Weight:6 kg,
Size:60w x180h cm', unit:'piece', unitPrice:20}),
// (Supplier)-[:HAS]->(SupplCatalog)
(a)-[:HAS]->(sc11), (a)-[:HAS]->(sc12), (a)-[:HAS]-
>(sc13), (a)-[:HAS]->(sc14), (a)-[:HAS]->(sc15), (a)-
[:HAS]->(sc16),
   (b)-[:HAS]->(sc21), (b)-[:HAS]->(sc22), (b)-
[:HAS]->(sc23), (b)-[:HAS]->(sc24), (b)-[:HAS]-
>(sc25)]
```

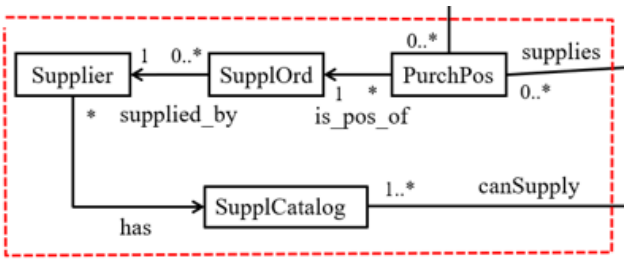


Figure 6. The Supplier part of the example database from Figure 1

Next, the Stock part, shown in Figure 7 below. Here are sample declarations:

```
// stock division
// create Part types
create type Part as (PartID char ,Designation char,
Color char, Weight char, Size char) nodetype
// PurchasedPart
create type PurchasedPart under Part as
(PreferredSupplNo int, sumOrderedThisYear currency,
discountPrice currency)
// InHouseProduct
create type InHouseProduct under Part as
(ProductionPlan char, producedThisYear int,
manufacturingCosts currency)
[CREATE
(a1:Location {LocationNo:10011, Aisle:1, Shelf:'left
A', Rack: 'A1'} ), // Location
(a2:Location {LocationNo:10012, Aisle:1, Shelf:'left
A', Rack: 'A2'} ),
// …
(l:Location {LocationNo:10111, Aisle:2, Shelf:'left
A', Rack: 'A1'} ), // Location without parts
//Part will be filled implicitly
// PurchasedPart
(p1:PurchasedPart {PartID:'P01'
,Designation:'Wallplug',Material:'Fiber',
Color:'grey', Weight:'6 g', Size:'12 cm',
PreferredSupplNo:103, sumOrderedThisYear:2000,
discountPrice:'0.04 €' }), //p1 Wallplug
```

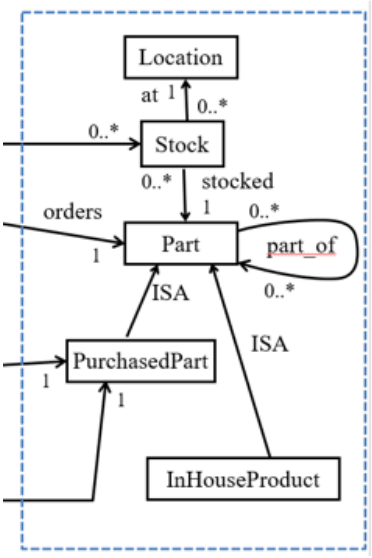


Figure 7. The Stock part of the example database from Figure 1

```
(p5:PurchasedPart  {PartID:'P05'  ,Designation:'Metal
nail', Material:'Metal', Color:'grey', Weight:'2 g',
Size:'A 50 x2.2 mm',
PreferredSupplNo:102,        sumOrderedThisYear:10000,
discountPrice:'0.005 €'}),  //p5 Metal nail
// …
(p30:PurchasedPart                          {PartID:'P30'
,Designation:'Degreasing liquid', Material:'benzine',
Color:'clear', Weight:'100 g', Size:'100 ml bottle' ,
       PreferredSupplNo:101,  sumOrderedThisYear:150,
discountPrice:'1.80 €'}), //p30 Degreasing liquid
// InHouseProduct
(p2:InHouseProduct  {PartID:'P02'  ,Designation:'Power
plug', Color:'white', Weight:'30 g', Size:'dia 5 cm
',
ProductionPlan:'P02 Power plug',
producedThisYear:1000, manufacturingCosts:'2.50 €'}),
(p3:InHouseProduct                          {PartID:'P03'
,Designation:'Hammer',                Material:'Compound
material',Color:'blue', Weight:'1,1 kg', Size:'35 cm
long',
ProductionPlan:'P03   Hammer',   producedThisYear:100,
manufacturingCosts:'2.50 €'}),
// …
(p28:InHouseProduct                         {PartID:'P28'
,Designation:'Tableleg',
Material:'Metal',Color:'Silver',              Weight:'1
kg',Size:'80w x120l cm',
ProductionPlan:'P28  Tableleg',  producedThisYear:160,
manufacturingCosts:'7.00 €'}),
```

```
// Stock
(s1:Stock       {PartID:'P02',       LocationNo:10011,
available:55,                         commissioned:20,
reserved_until:DATE'2023-09-22'} ),
(s2:Stock       {PartID:'P11',       LocationNo:10012,
available:500,                        commissioned:100,
reserved_until:DATE'2023-10-12'} ),
// …
(s34:Stock      {PartID:'P30',       LocationNo:10101,
available:30,                           commissioned:5,
reserved_until:DATE'2024-09-21'} ),
 //BOM
(p2)<-[:IS_Part_OF {no_of_components:1}]-(p11),
(p2)<-[:IS_Part_OF       {no_of_components:2}]-(p12)<-
[:IS_Part_OF {no_of_components:1}]-(p13),
(p3)<-[:IS_Part_OF {no_of_components:1}]-(p14),
// …
(p26)<-[:IS_Part_OF {no_of_components:1}]-(p23),
// Links: Parts<-Stock->Location
(p1)<-[:stocked]-(s33)-[:at]->(i3),
(p2)<-[:stocked]-(s1)-[:at]->(a1),
// …
(p30)<-[:stocked]-(s34)-[:at]->(k)]
```

Table II summarizes the schema objects (node and edge types) of the ERP graph schema and Figure 8 shows part of the resulting graph view of the database.

TABLE II. NODE AND EDGE TYPES IN AN EXAMPLE DATABASE (RELATIONAL DESCRIPTION)

Type name	Informal Description	SuperType
Customer	(CustNo, Name, Address)	
CustOrder	(CustNo, Datum, OrdNo, Summ€)	
OrderPos	(Id, Quantity, Unit)	
Location	(LocationNo, Reihe, Shelf, Rack)	
PurchasePart	(PartID, Designation, Material, Color, Weight, Size)	Part
InHouseProduct	(PartID, Designation, Material, Color, Weight, Size)	Part
Stock	(PartID, LocationNo, Available, Commissioned, Reserved_Until)	
Supplier	(SupplNo, Name, Address)	
SupplOrd	(OrdNo, SupplNo, Datum, Sum€)	
PurchPos	(PosNo, Quantity, Unit)	
SupplCatalog	(SupplNo, SPartNo, Desription, Weight, Unit, unitPrice)	

Type name	Leaving	Arriving	Other properties
Ordered_by	CustOrder	Customer	
Belongs_to	OrderPos	CustOrder	
Is_Part_Of	Part	Part	No_of_components
Stocked	Stocked	Part	
At	Part	Location	
Supplied_by	SupplOrd	Supplier	
Is_Pos_of	PurchPos	SupplOrd	
Has	Sypplier	SupplCatalog	
Orders	OrderPos	Part	
From_	OrderPos	Stock	
Supplied	PurchPos	ParchasePart	
Can_Spply	SupplCatalog	PurchasePart	
Serves	PurchPos	OrderPos	

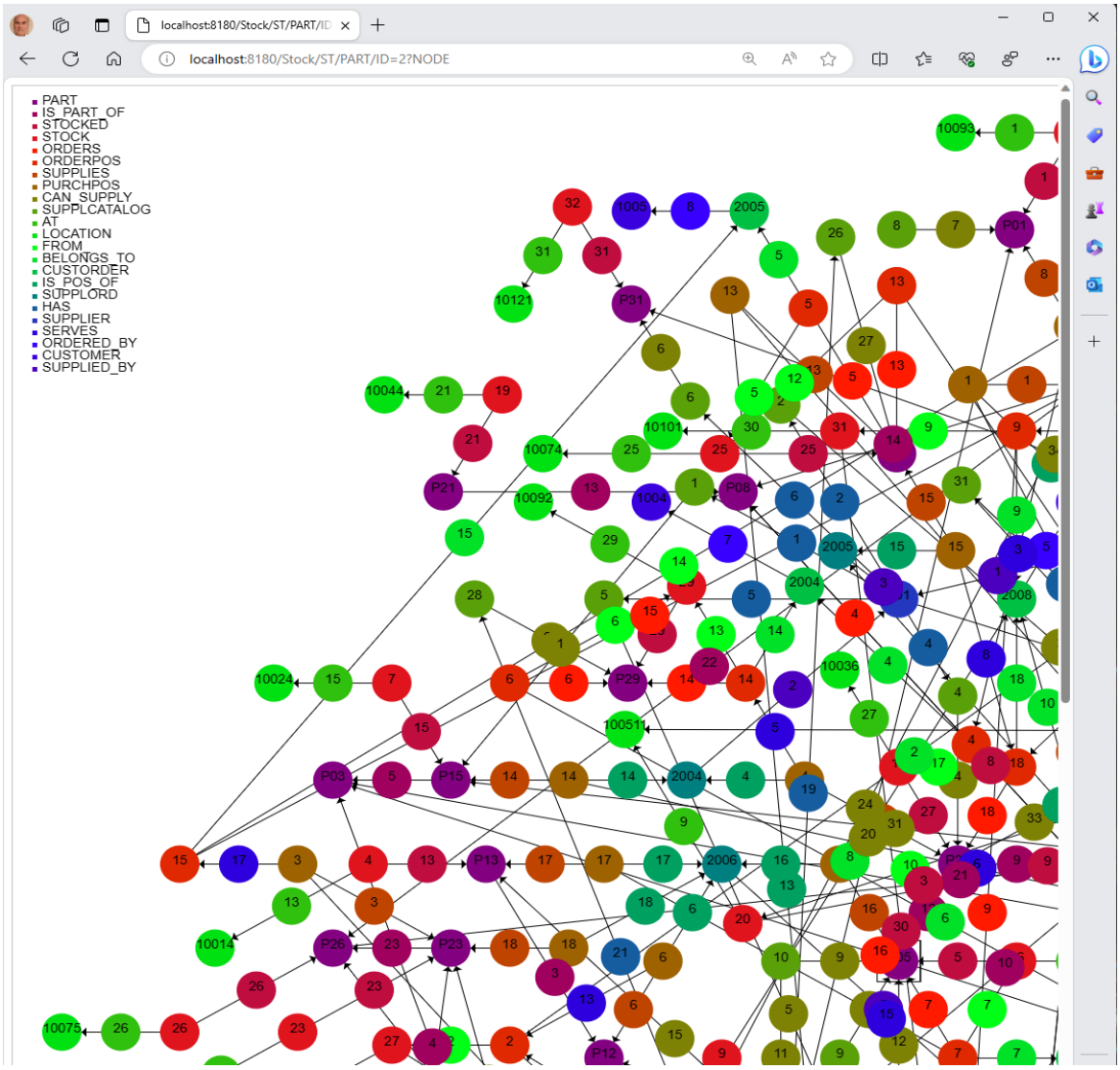


Figure 8. A part of the ERP example graph, after changes to primary keys similar to Figure 2 and 3 (e.g., PART now has key PartID).

V. CONCLUSIONS

The purpose of this paper was to report some progress in our Typed Graph Modeling workstream. The work is available on Github [11] for free download and use and is not covered by any patent or other restrictions. The main challenges, as expected, were related to the implementation of the MATCH algorithm for repeating patterns, and the solution found is an elegant one involving continuations and documented in the Pyrrho blog [12] and in [11]. We plan to add further facilities for altering the types of graph properties, and to track development of the forthcoming GQL standard.

Unsurprisingly, the performance of our implementation is modest for complex statements when the database becomes large. Simple CREATE and MATCH statements like those found in benchmarks are processed at over 2500 per second. The implementation will no double benefit from a review of this aspect.

The current “alpha” state of the software implements all of the above ideas. The test suite includes simple cases that demonstrate the integration of the relational and typed graph model concepts in Pyrrho DBMS. The implementation is backward compatible with previous versions of Pyrrho DBMS, so legacy databases can immediately use these new capabilities. Pyrrho DBMS is free standing and works directly with the operating system (Windows, Linux, or MacOS), and clients interact with the server using TCP/IP or HTTP.

It is our hope that other DBMS developers will also adopt GQL in new versions of their DBMS.

REFERENCES

[1] F. Laux and M. Crowe, "Typed Graph Models and Relational Database Technology", DBKDA 2023: The Fifteenth International Conference on Advances in Databases, Knowledge, and Data Applications, IARIA, March 2023, pp. 33-37, ISSN: 2308-4332, ISBN: 978-1-68558-056-8

[2] ISO 9075-16 Property Graph Queries (SQL/PGQ), International Standards Organisation (2023).

[3] F. Laux and M. Crowe, "Information Integration using the Typed Graph Model", DBKDA 2021: The Thirteenth International Conference on Advances in Databases, Knowledge, and Data Applications, IARIA, May 2021, pp. 7-14, ISSN: 2308-4332, ISBN: 978-1-61208-857-0

[4] E. J. Naiburg and R. A. Maksimschuk, UML for database design. Addison-Wesley Professional, 2001

[5] R. De Virgilio, A. Maccioni, R. Torloner, "Model-Driven Design of Graph Databases", in Yue, E. et al (eds) Conceptual Modeling, 33rd International Conference (ER 2014), Springer, Oct 2014, pp. 172-185, ISSN: 0302-9743 ISBN: 978-3-319-12205-2

[6] F. Laux, "The Typed Graph Model", DBKDA 2020 : The Twelfth International Conference on Advances in Databases, Knowledge, and Data Applications, IARIA, Sept 2020, pp. 13-19, ISSN: 2308-4332, ISBN: 978-1-61208-790-0

[7] M. Crowe and F. Laux, "Database Technology Evolution", IARIA International Journal on Advanced is Software, vol. 15, numbers 3 and 4, 2022, pp. 224-234, ISSN: 1942-2628

[8] S. Shah et al., The PostgreSQL Data Computing Platform (PgDCP) (Online), Available from: https://github.com/netspective-studios/PgDCP [retrieved: Aug 2023]

[9] N. Francis, A. Gheerbrant, P. Guagliardo, L. Leonid, V. Marsault, et al., A Researcher's Digest of GQL. 26th International Conference on Database Theory (ICDT 2023), Mar 2023, Ioannina, Greece. doi:10.4230/LIPIcs.ICDT.2023.1. https://hal.science/hal-04094449 [retrieved: Aug 2023]

[10] M. Laiho and F. Laux, An Introduction to Neo4j Graph Database, DBTechNet.org preprint [retrieved: Aug 2023].

[11] M. Crowe, PyrrhoV7alpha, https://github.com/MalcolmCrowe/ShareableDataStructures [retrieved: Nov, 2023]

[12] M. Crowe, PyrrhoDBMS http://pyrrhodb.com [retrieved Nov, 2023]

Governance and Legitimacy of Artificial Intelligence

Olga Gil
Instituto Complutense de Ciencias de la Administración
Departamento de Historia, Teorías y Geografía Políticas
Facultad de Ciencias Políticas y Sociología
Universidad Complutense de Madrid
Madrid
olgagil@ucm.es

Abstract - **Amids challenges posed to humanity by artificial intelligence disruptive developments, this work is set to engage discussants from different perspectives -encompassing scientists in different fields, governments, firms and other social actors- on the topics of artificial intelligence, governance and legitimacy. The main aim and output of the paper is to present a dashboard for the analysis of governance and legitimacy of artificial intelligence. This Dashboard resolves disputes within the literature on political theory over classical approaches to study governance and legitimacy. The Dashboard has also the capacity to allow for comparisons in AI governance and legitimacy in democratic and non democratic regimes, at different government levels, both in Western Countries and in the Global South. An additional output is the application of the framework to the case of China as a case study. This analysis is carried out by applying the framework to take a fresh look at existing data in the Chinese case and showing its value as a methodological and analytical tool.**

Keywords - Artificial intelligence; democracy; ethics; political theory; governance.

I. Introduction

This work is an extended version of "AI Philosophy: Sources of Legitimacy to Analyze Artificial Intelligence," a paper presented to the IARIA Annual Congress on Frontiers in Science, Technology, Services, and Applications in 2023 [1]. This extended version aims to present a general framework to analyze artificial intelligence (AI), and to discuss legitimacy and governance from political theory as a stream of philosophy. As such, the work addresses questions related to governance and legitimacy that are at the basis of political and social power and of command and control. At the end of 2020, Pfizer-BioNTech vaccine based on mRNA molecules made a breakthrough, allowing for a first treatment against the COVID-19 plague. The vaccine used modified RNA molecules and ferried them for the first time as a drug into cells. This vaccine was made possible with complexity, using algorithms departing from a strict digital approach to more complex algorithms incorporating layers, in the same ways that neurons branch out in the human brain. This fascinating breakthrough furthers -even more- the appetite for competition among the big actors: Google, Elon Musk, the Chinese government, among others, all wanting to reach a trillion operations in machine learning [2]. The example of the COVID-19 vaccine brings us the evidence of the importance of the use of algorithms for the good of humanity. From the point of view of physics and biology, Contera stresses that reality is not digital, it is analog, and therefore complex: Current artificial intelligence seeks to achieve this complexity by including new parameters and new interactions [2]. But there are limitations to this pattern of development towards complexity: the first is based on energy reasons - the cost of computations and the blockchain is very high. The second limitation is based on geopolitical reasons. Taiwan is currently the only country capable of producing a computing chip below the size of five nanometers [2]. This makes the United States, Western economies and the global south heavily dependent on a single company, TSMC, for the supply of leading edge technology chips. Only TSMC in Taiwan and Samsung in South Korea can make the most advanced semiconductors, and this, for the case of Taiwan exposure to China, is interpreted by the United States as putting at risk the ability "to supply current and future [US] national security and critical infrastructure needs" according to María Ryan [3][4]. This is evident when the United States Department of Commerce's Bureau of Industry and Security announces the implementation of export controls to restrict China's ability to both purchase and manufacture certain high-end chips used in military applications on October 7th, 2022. This actually means restricting China´s ability to obtain advanced computing chips, develop and maintain supercomputers, and manufacture advanced semiconductors. Similar risks -and opportunities- are perceived by China. Both risks and opportunities underlie the Chinese Party Constitutional amendment made in October 2022, looking forward to making sustained and steady progress with the One Country, Two Systems policy, advancing national reunification with Taiwan [5].

Thus, this work applies the political theory framework to China, bearing in mind the context and very interesting issues at hand: competitive interests and domestic preferences, economic development, national security and social control. There are challenges in trying to tame the beast of reality -as the big actors are seeking to do

with algorithms- and with the resources and human talent that are assigned for the task. These challenges make the study of current changes in artificial intelligence (AI) from the perspective of social sciences, and in particular from the perspective of legitimacy and democracy -or the lack thereof- interesting and acute. Other questions related to political theory also motivate this study: what can we learn about the complex reality of AI related to command and control in China? What may we learn about the future society and the polity against AI development in China? And, are there any particular cultural values enshrined in the country´s AI development?

In the following sections, the methodology is introduced and a general theoretical framework is proposed, the case study of sources of legitimacy and control in China follows, and finally a discussion with conclusion and further work is presented, followed by acknowledgements.

II. Methodology

The methodology of this research seeks to bring basic questions linked to legitimacy - a basis of governance- into the study of artificial intelligence. The purpose is to reflect upon how artificial intelligence is going to affect democratic and non democratic regimes.

The study departs from classical publications in political theory by Max Weber and Craig Mathesson [6][7], whose approaches combined allow to draft a table with an eight dimensional view of sources of legitimacy. These approaches taken combined, however, are unable to capture new features linked to legitimacy when artificial intelligence is taken into account. Searching for what is missing in these classical inquiries when AI is taken into account a new theoretical framework is developed. This new theoretical framework allows for a comparison of national cases, and eventually, supranational and subnational cases. The selection of studies started by a search in scopus with the terms artificial intelligence AND China in 2020, 2021, 2022. This brought about 776 articles. The selection was further refined under the social sciences category, with 170 documents published matching the query. These journal articles were reviewed looking for governance and legitimacy as topics for retrieval and further work, identifying 37 source articles. Once first relevant works were identified, the reference list of these articles became a main source of materials -both those that were included in the scopus database or were not- as detailed knowledge became crucial to build up the study. Google scholar was also utilized, searching for the first 10 publications on artificial intelligence and social sciences, the 10 most cited, and the ten most recent ones. These works were reviewed searching for interesting insights. Proquest database has also been consulted, with the query artificial intelligence in the Financial Times newspaper. Specific articles on the query were of value to identify authors with new ideas on artificial intelligence nowadays and how AI affects governance. As a result, these searches brought about information from comparative reports with general information on the United States [8], the work on Europe [9][10], and on China and China local AI ecosystems [11][12], which is the focus for the purpose of this work.

This research and discussion have been pursued without the aid of artificial intelligences or data bases in the process of ideas. Research and discussion are the result of a human mind. There is no use of any big data software, organic life engineering, or cyborg aid. Thus, at this stage, the results of the work are solely the responsibility of a human author´s mind. At a future stage, it could be explored whether there are interesting possibilities from non natural intelligences to broaden the scope and findings of this research.

III. Theoretical framework of AI Governance: from related work to sources of legitimacy to analyze artificial intelligence

The theoretical framework is based on the sources of legitimacy to analyze artificial intelligence. Here we are bringing to the fore political theory to address a contemporary problem: AI governance. This is what the paper tries to achieve, a better understanding of governance in the context of AI. For this purpose, in this part of the work, Table I, including an eight dimensional view of sources of legitimacy is developed. Table I is based on dimensions, concepts and definitions from classical works by Max Weber and Craig Mathesson [6][7]. With over five million mentions in Google Scholar, Weber´s work is a reference to explain politics, society and economics. Decades later Mathesson includes democracy as a fundamental axis to review Weber's approach on legitimacy. The current work argues that there is room for further improvement, departing from the insights from these two authors. Improvement is pursued in two steps: Step one is developing a framework for analysis based on the theories of these two authors, the eight dimensional view of sources of legitimacy. Step two, in the following section, follows, with the next stage, using fundamental questions from political theory to address the contemporary problem of AI governance. In doing so, method -as source of change- and legitimacy are enshrined in the Gil dashboard making up for an upgraded theoretical framework. The new Dashboard has been developed in a wider context that is not addressed in this article: The wider context aims to compare the AI regulatory framework of China, the European Union and the United States [8]-[12], which is the endeavor the author is currently devoted to in a wider research. The current work focuses on the theoretical dashboard that has been developed to make the comparisons. Using a name for the Dashboard follows the practice of using the name of the author for scales –such as Sherry Arnstein Ladder of Citizen Participation, one of the most influential models in the field of democratic public participation; it also has the purpose of setting a reference for further discussion across disciplines.

The following work tries to unveil a complex reality, 1) where there are new rules attached to command and control derived from the use of AI in political regimes and 2) to bring to light new ways of thinking about AI, governance and legitimacy. A framework for analysis, the Gil dashboard for legitimacy is developed. The dashboard

allows for comparisons of most similar and most different cases. The theoretical framework is in the intersection between values and AI development, and allows to unveil how AI is mediating problems related to coordination and control, what uncertainties about the future society and the polity different countries face against AI development, and what could we say about different cultural values.

We depart from the work on legitimacy from Max Weber -for whom there exist three types of domination, charismatic, traditional and rational or legal [6]. This framework was revised by Matheson [7] in 1987, nearly a century after Weber started writing. Matheson qualifies and opposes Max Weber theory on legitimacy. Later on, and departing from Matheson, the current work develops a theoretical framework to allow for the comparison of AI legitimacy bases in the European Union, the United States and China - and could be valuable for the analysis of developing countries, and countries in the global south.

TABLE I. THE EIGHT DIMENSIONAL VIEW OF SOURCES OF LEGITIMACY, BY OLGA GIL

Dimension	Definition of the dimension
Convention	Norms, rules: legal or customary rules that prescribe forms of behavior
Contract as basis of legitimacy	Mutual rights and obligations. The theory of consent as the basis of obligations
Basis of legitimacy in a conformity with universal principles: natural law	Theories of natural law, aka, the existence of a natural order superior to man-made law
Sacredness of authority	Power-holder or his/her norms considered to be sacred divine right of reigns. For Max Weber it could also be an attribute of an office rather than a person
Legitimacy by expertise	Technical expertise, in the vein defended by Saint-Simon, Taylorian theories, or historic laws
A popular mandate in a constitutional democracy	Popular mandate: a claim to democratic election in accordance with constitutional procedures. Based on constitutionalism, power holders elected in accordance with constitutional procedures. Here we find a distinction between populist democracies, where the will of a majority rules, and constitutional democracies, were there will of the majority is limited by a constitution
Personal relation	Domination, in which there are close ties between power-holders and power-subjects such as personal authority or paternal authority relationships
Personal quality of the power holder	Domination based on the personal quality of the power holder, by virtue of which he/she can claim a right of command

Weber differentiated three types of domination: charismatic, traditional and rational or legal. This differentiation is based on the legitimacy of the power-holder. The work by Matheson nearly a century later includes eight types of domination, including the perspective of both the power holders and the power subjects. The main critique that Matheson introduces to Weber's work is that democracy and its effects along the XX century are not reflected in Max Weber typology. Matheson reaches new layers of granularity for the study of the polity and society with his revised proposal. From Matheson´s critique of Weber Table 1 above is developed: The table explains visually the eighth types of domination. This would be an eight dimensional view of sources of legitimacy.

Having AI in mind and looking at this framework for the analysis of the cases selected, observations about new sources of legitimacy out of the scope of the table above can be drawn. A first one would be coercion as an instrument for legitimacy. A second source of legitimacy would be AI development outside the umbrella of the state, based in ethics codes. For instance, an applied comparison of national AI strategies in nine countries, including China and the United States finds that national AI strategies have an approach towards AI governance that entails cooperation among the public sector, industry and academia and this has been based largely on ethics [8]. Based on ethics, cooperation is achieved with voluntary mechanisms including best practices, codes of conduct, and guidelines. At the core of a general approach to use ethical guidelines as an efficient measure to prevent or reduce harm caused by AI, the general argument is for its higher flexibility, as opposed to hard regulations that could represent an obstacle to economic and technical innovation [8][9], or other means of legitimacy.

IV. METHOD AS A SOURCE OF CHANGE AND LEGITIMACY

A third source of legitimacy would be linked to method. Matheson's approach to sources of legitimacy reviews Max Weber work making important contributions. But a further contribution is missing: the concept of improved democracies through method as source of legitimacy. This type of legitimacy -experimenting with method, in an active process to reach better results- is not included in Matheson analysis. Method points out to new types of democracies that would not be only based on a popular mandate. Method has been the basis to reach new knowledge following the scientific revolution in Europe. Method, in contrast, has not been explored as such to improve democratic governments. The result is that there has not been an appraisal of method as a way to reach better results in democratic regimes. An example of the dangers and limitations of not including method as a source of improved legitimacy is the work comparing national AI strategies in nine countries, including China and the United States [8], stressing the lack of concrete mechanisms for inclusion of civic society and public engagement in AI control.

These new sources of legitimacy -coercion, ethics, improved method, and legitimation based on algorithms- will be incorporated in the previous table in order to develop a new table, the Gil Dashboard, allowing us to analyze artificial intelligence in case studies, in multilevel analysis and from a comparative perspectives. The sources of

legitimacy are incorporated close to the category that is more akin to the concept, if any. The additions are included in bold text.

Table II. The Gil Dashboard: Thirteen sources of legitimacy to analyze AI

Dimension	Definition of the dimension
Convention	Norms, rules: legal or customary rules that prescribe forms of behavior
Contract as basis of legitimacy	Mutual rights and obligations. The theory of consent as the basis of obligations
Basis of legitimacy in a conformity with universal principles: natural law	Theories of natural law, aka, the existence of a natural order superior to man-made law
Sacredness of authority	Power-holder or his/her norms considered to be sacred divine right of reigns. For Max Weber it could also be an attribute of an office rather than a person
Legitimation by **human** expertise	Technical expertise, in the vein defended by Saint-Simon, Taylorian theories, or historic laws
Legitimation based on an algorithm	**Legitimation based on macrodata –hindering the idea of individual liberty and decisions taken by means of human conversation and persuasion**
A popular mandate in a constitutional democracy	Popular mandate: a claim to democratic election in accordance with constitutional procedures. Based on constitutionalism, power holders elected in accordance with constitutional procedures. Here we find a distinction between populist democracies, where the will of a majority rules, and constitutional democracies, were there will of a majority is limited by a constitution
Improved democracies experimenting with method	**A type of legitimacy based not only in a popular mandate but also on experimenting with method and in a continuous process, in order to reach better results, including accountability**
Regimes -non democracies- developed through method	**A type of legitimacy based on experimenting with method and a continuous process to justify objectives and reached results**
Personal relation	Domination, in which there are close ties between power-holders and power-subjects such as personal authority or paternal authority relationships
Personal quality of the power holder	Domination based on the personal quality of the power holder, by virtue of which he/she can claim a right of command
Coercion	**The use of power to influence someone to do something they do not want to do, from exerting fear to nudging as positive reinforcement**
Societal cooperation, excluding the polity	**Development of mechanisms of cooperation among the public sector, industry and academia: cooperation is achieved with voluntary mechanisms including best practices, ethical codes of conduct, and guidelines**

V. Applying the Dashboard to study governance, legitimacy and control: Artificial intelligence in China

In this section we analyze sources of legitimacy using the author´s Dashboard in the Chinese case. The work proceeds first of all with a brief introduction on the economic governance of AI in China, followed by Table III, with a quantitative analysis (where 1 is existence and 0 absence), and a qualitative analysis follows. At this stage of the research using a binary code has the sole purpose to state existence or absence of a given dimension. The subsections explain those features that have proved existing. To refine limitations derived from using binary code, this coding could be complemented with normalized scales –i.e. Likert scale- other metrics showing further comparative scalability for each of the dimensions in the Dashboard, coupled with in depth dimension studies.

The baseline of the economic governance of AI in China has laid on the increase of total fiscal expenditures on science and technology rising from 48 per cent in 2007-2011 to 59 per cent in 2015-2016 [13]. Provinces and local governments have significant autonomy in the implementation of these funds, and from different approaches [14]. There are local unbalances in AI development, with three cities being home to 70 per cent of AI firms: Beijing being home to 43 of Chinese firms, Shanghai at 15 per cent and Shenzhen at 12 per cent [15]. Following with expertise, the mode of economic governance has not been based on cutting edge technologies in China. The mode of AI economic governance, instead, has been based in rapid deployment and scaling of existing AI technologies [15]. The results have been fusion and speed over breakthrough technologies, and ensuring the adoption of existing technologies. Adoption and scale have been the formulae for AI implementation, both in the private and the public sector. This is very much in contrast with the case of European countries, where deployment of AI technologies at the local level remains very low [9]. An additional key in economic governance has been the attraction of global and supra-local linkages by ambitious policy makers searching for increased access to capital and other AI ecosystems: Linking to cities such Amsterdam, Barcelona, Stockholm, and clusters forming around Cambridge, Oxford and Manchester –with the AI ecosystem around Manchester university and the United Kingdom government communications headquarter. Another component in economic governance has to do with the objective to reduce policy fragmentation in China. In order to do so, local governments are incentivized to develop plans that can be later used to assess progress and to induce competition between different regions and localities.

TABLE III. THE GIL DASHBOARD: THE GIL DASHBOARD ON SOURCES OF LEGITIMACY AND CONTROL: AN APPLICATION TO CHINA, CONCEPTUAL, QUANTITATIVE ANALYSIS.

Dimension	Quantitative analysis	Definition of the dimension
Convention	1	Norms, rules: legal or customary rules that prescribe forms of behavior
Contract as basis of legitimacy	0	Mutual rights and obligations. The theory of consent as the basis of obligations
Basis of legitimacy in a conformity with universal principles: natural law	1	Theories of natural law, aka, the existence of a natural order superior to man-made law
Sacredness of authority	0	Power-holder or his/her norms considered to be sacred divine right of reigns. For Max Weber it could also be an attribute of an office rather than a person
Legitimation by **human** expertise	1	Technical expertise, in the vein defended by Saint-Simon, Taylorian theories, or historic laws
Legitimation based on an algorithm	**0**	**Legitimation based on macrodata –hindering the idea of individual liberty, or decisions taken by consensus**
A popular mandate in a constitutional democracy	0	Popular mandate: a claim to democratic election in accordance with constitutional procedures. Based on constitutionalism, power holders elected in accordance with constitutional procedures. Here we find a distinction between populist democracies, where the will of a majority rules, and constitutional democracies, were there will of a majority is limited by a constitution
Improved democracies experimenting with method	**0**	**A type of legitimacy based not only in a popular mandate but also on experimenting with method and in a continuous process, in order to reach better results, including accountability**
Regimes -non democracies- developed through method	**1**	**A type of legitimacy based on experimenting with method and a continuous process to justify objectives and reached results**
Personal relation	0	Domination, in which there are close ties between power-holders and power-subjects such as personal authority or paternal authority relationships
Personal quality of the power holder	0	Domination based on the personal quality of the power holder, by virtue of which he/she can claim a right of command
Coercion	**1**	**The use of power to influence someone to do something they do not want to do, from exerting fear to nudging as positive reinforcement**
Societal cooperation, excluding the polity	**1**	**Development of mechanisms of cooperation among the public sector, industry and academia: cooperation is achieved with voluntary mechanisms including best practices, ethical codes of conduct, and guidelines**

A. Convention

The first source of legitimacy and control that we can draw from this table and apply to the Chinese case is convention. It could be argued that in China there are general changes in convention as a source of legitimacy, understood as norms, rules –legal or customary rules- that prescribe forms of behavior.

The mode of social governance has implications in China's choice of adoption of AI technologies. As Ding states [11], the State Council's AI plan sees AI playing an irreplaceable role in maintaining social stability. In practice, this is reflected in local-level integrations of AI across a broad range of public services, including judicial services, medical care, and public security. Specially affecting the mode of social governance are two areas, the first one, concerning privacy, and the second concerning private companies' participation in social credit systems [16][17]. AI is proved as a good tool to improve efficiency and reach services, however it is a less desirable tool for complex areas where context, emotional judgment, flexibility and moral judgements are crucial.

In the case of the social credit system, Lewis defines it as an initiative based on a cluster of experiments harnessing public data with the aim to improve governance [18]. This improvement seeks to boost trust among government, firms and individuals, and includes larger national efforts - the Blacklist-Redlist Joint Sanctions and Rewards regimes- as well as smaller efforts being implemented in some cities. Lewis defines it as:

"an overarching policy initiative consisting of multiple sub-systems (...) with different policy goals and rules, rather than one distinct system. Ambitiously, it takes aim at nearly all of China's development ills – from environmental protection to IP and financial fraud to academic plagiarism" all of which the Chinese government believes stems from firms and individuals not following laws and regulations" [18].

The intent, according to the Chinese government, would be to enshrine trust in order to develop a market economy [18]. An important loophole, however, is that individuals or firms have little knowledge about the data collected. Lewis recalls that the black list regime has been reinforced and had real implications for business and individuals, but it is difficult to be conclusive about whether policy is truly achieving the general goal of business and individuals behaving in a more trustworthy manner [18], and more generally, whether the system improves trust in Chinese institutions.

General changes in convention as a source of legitimacy, understood as norms, rules –legal or customary rules- that prescribe forms of behavior are the aim of AI scoring systems assigning a credit to the population with the aim, according to the government, to improve societal trust. Xiamen and Fuzhou are two examples of cities that have implemented score systems for their population since 2018. Xiamen has over 85,000 users exchanging their scores to avail services. Fuzhou has over 1,19 million residents doing the same. Scores look at the behavior of residents, and the individual participation such as keeping promises as measure of responsibility and trust, while a breach of contracts would be contemplated as unwillingness to obey the law. A system of credit repair has been invented, with the possibility to gain credit back through active participation in social service, public interest events and welfare activities. These are mechanisms to change traditional convention. Other mechanisms are local scores looking at hard working, observation of ethics and morals, as defined by the government, delayed payment, the follow up of administrative regulations and legal duties. Danit Gal [19] argues that mechanisms of credit scoring exist in other countries such as the United States, however, the level of development and deployment in China makes it unique in scale.

It could be argued that changes in convention and social cooperation affect innovation ecosystems as well. Ding remarks how the central government's important guiding role in China is targeted by other public and private actors, pursuing their own objectives in AI, including academic labs, bureaucratic agencies, private companies and subnational governments [11]. Many actors involved have resulted in rapid innovation in many fields, based in local innovation ecosystems. By the end of 2018, 20 provinces had issued 30 specific AI policies “many forward thinking local governments implemented AI-related policies that preceded national government action” [12]. The pragmatic approach to innovation has resulted in important developments in the fields of healthcare, medical image processing and pharmaceutical research. Kim describes the ecosystem of actors as a hybridized industrial ecosystem including firms, networks of small and medium enterprises and research institutes specially adapted to the local conditions [20]. Ding emphasizes the importance of specializing in AI subdomains, and he actually stresses the importance of specialization in AI subdomains or parts of the value chain as clues to success [11]. Ding also stresses that in a new vein, transparent budget disclosures show allocation to companies in subdomains ranging from predictive analytics of smart city data to sign language translation [12].

An example of a hybridized industrial ecosystem has been the Hangzhou AI Town opening for business in July 2017 –inspired by visits from local leaders to Silicon Valley and searching for similar spillovers. The mission of this local ecosystem has been to link the e-commerce company Alibaba and subsidiaries -with more than 90 per cent of the projects in some categories-, with Zhejiang university, graduates studying overseas, and local businesses together in a cluster. The creation of the AI park is housed in the Hangzhou Future Sci-Tech City, connected to a larger infrastructure of science and technology parks [11]. This local industrial ecosystem has been designed with international linkages in mind, and thus Silicon Valley Bay area council has an office helping Californian companies to register enterprises, and Hangzhou AI Town, in turn, has offered 3 million RMB for settlement expenses, and 15 million RMB in subsidized office space costs [11]. An additional aim for Hangzhou AI Town managers has been attracting talent, such as returning Chinese graduates from international universities [21], but not exclusively Chinese: recent measures restricting the support of development, production, and semiconductor fabrication by United States nationals in China show that global talent attraction was also a key in this local development model [22]. In Hangzhou AI Town, Alibaba functioned as anchor tenant, a necessary condition for AI development success that is also found in other Chinese local ecosystems [11]. Jeffrey Ding also speaks of elite universities, such as Zhejiang University as a glue to hold the ecosystem together, and the existence of large technology firms such as Alibaba as a requirement to enhance productivity and local innovation [11]. The involvement of private actors, however, brings in the risk of inequality [32]. In order to avoid inequalities and the marginalization of social groups, the adoption of AI educational tools has been defended as a need, as well as a source for better comprehension about how innovation may prevent the marginalization of less favored social groups.

B. Contract

We find that the appeal to contractualism is absent as an instrument of legitimacy in China -as the search for related keywords yielded invalid or no significant results.

C. Basic of legitimacy in conformity with a natural law: Ethics as a set of laws

Legitimacy based in a natural law is the following category existing in our quantitative analysis. Legitimacy to set up AI in public services in China has been driven according to Rogier Creemers by the ideological view that social order is governed by an objective and a determined set of laws where AI can solve social problems and help to understand those laws [19]. In this context, AI is generally designed to improve existing institutions, not to replace or reform them, and thus policies integrating AI play an important role. Policies and public-private partnerships are at the center of this short of approach, in which national, local and company levels concur often. Policies focus on speeding up technology development, data collection and implementing pilots. Issues such as accountability, data privacy, and risk management appear to be secondary to crucial developments. We thus find legitimacy based on universal principles, ethics in the case of China is linked to the development of AI outside the umbrella of the state. This result is consistent with the findings of Gianni et al., in an applied comparison of national AI strategies in nine countries, including China and the United States [33] –at least until the summer of 2023, when the Chinese Minister

of Science and Technology shifts policies on generative AI towards regulation. The Chinese case reflects that the source of legitimacy for AI governance entails cooperation among the public sector, industry and academia. This is AI development outside the umbrella of the state, based on ethics codes [31], up to the scope of time covered by this research. In this particular conception of ethics, cooperation is achieved with voluntary mechanisms including best practices, codes of conduct, and guidelines. In general terms, at the core of a general approach to use ethical guidelines as an efficient measure to prevent or reduce harm caused by AI the general argument is for its higher flexibility, as opposed to hard regulations that could represent an obstacle to economic and technical innovation [32][33], or other means of legitimacy.

In China's approach to ethics there is a basis of legitimacy in a conformity with a call to universal principles, where harmony, as principle in Chinese philosophy for all life forms [31] would be relevant in the contexts of human-machine interactions [19]. The call to universal principles making a reference to harmony is furthered in a new document addressing human-machine harmony, and more specifically stating in article n. 1:
"AI development should begin from the objective of enhancing the common well-being of humanity; it should conform to human values, ethics, and morality, promote human-machine harmony, and serve the progress of human civilization; it should be based on the premise of safeguarding societal security and respecting human rights, avoid misuse, and prohibit abuse and malicious application [32, their translation]."

Multi-stakeholder committees have been settled outlining AI ethic principles, many of them according to global standards [33]. An additional challenge is to bring a number of relevant stakeholders into key conversations on AI ethics, both internationally [19] and at the national level. There have been expert groups, including several companies, business associations and expert groups releasing principles, and the New AI Governance Expert Committee, created by the Ministry of Science and Technology, stating that AI should conform to safeguard social security and respecting human rights, according to Creemers [22]. Interpreting this statement would make us close to Chinese Communist Party ideology, which in the aftermath of the 20th National Congress of the Chinese Communist Party closing in 22th October 2022 is driven by a top down hierarchy, with General Secretary Xi Jinping on top, and 90 million Communist Party members: As Xinhua relates, "Xi Jinping Thought on Socialism with Chinese Characteristics for a New Era (...) should be incorporated into the Party Constitution (...) with Comrade Xi Jinping at its core to advancing the Party's theoretical, practical, and institutional innovations." [5][16]

D. Sacredness of authority and improvement of democracy thorough method

Sacredness of authority is absent, as well as improving democracy through method and personal relation -as the search for related keywords yielded invalid or no significant results.

E. Method as a source of legitimacy

Whereas method is the basis to reach new knowledge following the scientific revolution in Europe, method, in contrast, has not been institutionally embedded as a basic feature to improve democratic -or undemocratic- governments. The result is that there has not been an appraisal of method as a way to reach better social results in democracies [23]. The sources of legitimacy linked to method deserve further elaboration. Matheson's approach to sources of legitimacy reviews Max Weber work making important contributions. However, the search of improved democracies through method as a source of legitimacy is not included in Matheson analysis. This type of legitimacy is based on the active involvement of citizens -or residents- in promoting public values. In this active involvement, there is a need for a process of social construction. This social construction would entail employee participation, citizen involvement, empowerment and consultation at center stage: not just as outcome, but in the dialectical process of construing public institutions and a theory where the emphasis is on the public in the administrative process [24]. This conception of social design includes the general public, the government, and the public administration. Social design would be understood as evolutionary, as an integrative process to build shared realities that could lead to a process of invention, evolution and self-governance [25]. This is what the Dashboard refers to as experimenting with methods and in an active process, not only based on a popular mandate, to reach better results. The work by Gianni et al. comparing national AI strategies in nine countries, including China and the United States stresses the lack of concrete mechanisms for inclusion of civic society and public engagement in AI control [27]. This could be understood as lacking the experimentation with method as a formula to build better shared realities.

In the case of education we could argue AI development in China is experimenting with method as a continuous process, and justifying the reach of better results. However, as Liu argues, high-quality education involves creativity, collaboration and critical thinking, and for those aims, the role of just AI technologies for the next generation of students is limited [42].

F. Basis of legitimacy in conformity with expertise

Pointing at sources of legitimacy and control in China, expertise is the following category in our analysis. Legitimation by expertise is on the basis of economic governance in China. Legitimation by expertise is heavily ingrained in the AI strategic plan designed by the China´s State Council in 2017 [35]. This is also the case in the plan when calling for the development of a whole range of AI related healthcare technologies to put cognitive computing at the service of learning, recalling and appling vast amounts of text works for medical professionals. Expertise as a basis for legitimacy is reported by Karen Hao at The Wall Street Journal:

"Chinese leader Xi Jinping has packed the top ranks of the Communist Party with a new generation of leaders who have experience in aerospace, artificial intelligence and other strategically important areas (…) Chinese officials with technical expertise occupy 81 seats, nearly 40% of the total, in the new Central Committee—the elite body that decides major national policies—according to data compiled by the Washington-based Brookings Institution think tank and shared exclusively with The Wall Street Journal. That compares with less than 18% in the previous Central Committee." [34]

Legitimation by expertise has also been applied to the judiciary: It is the baseline for System 2016, which is Shanghai High People's Court Intelligence assistive case-handling system for criminal cases. The purpose of the system is to improve the quality and reduce false, unjust or wrong changes and sentences [32]. Eugeniu Han explains that the system has two components, a cross reference system using speech recognition to compare different types of evidence and alert the judge about contradictions in the judge patterns [32]. The second component is a sentencing reference tool based on machine learning combining the defendants basic information and a large database of past court records to make sentencing recommendations. The system can also be used to judge the judges and prosecutors by pinpointing the outliers [32], moreover, applications within System 2016 could skew a prosecutor or judge to the detriment of the defendant. Han stresses that defendants and their defenders may lack the technical knowledge, resources and access to challenge AI processes for generating a sentencing reference and assess its potential biases [32]. Gal suggests another loophole since Alibaba is usually the defendant in many cases while is also the co-creator of the Smart Court System 2016: conflicts of interests are clearly at stake, "exacerbating legal accountability for decisions made by using these systems" [19]. Gal pinpoints that the use of AI to support the court system occurs in other countries, what is unique to China is a "smart court and an AI judge handling claims against a corporate actor, while also being developed by the same corporate actor [19].

An additional tool contributing to legitimation by expertise is City Brain, a system first launched for traffic management in Hangzhou in 2016 with the aim of tackling traffic congestion. City Brain was developed by 13 companies together with the city government and based on Alibaba cloud platform service: The firm optimizing traffic has developed into a data coordination center consolidating data from over 700 IT government agencies. This data coordination center offers services for parking, traffic management -including ambulances and firefighters- waste collection and even health monitoring of the city's aging population [32]. With different modifications City Brain has been implemented in more than 10 cities in Asia, sometimes under the umbrella of the Belt and Road initiative of the Digital Silk Road.

Following expertise as a base for legitimacy, in 2018 the Guangzhou Women and Children Medical Centre developed an AI prototype using NLP and deep learning to work with relevant information from 1,4 million patients to help frontline patient care, for instance triaging patients to decide degrees of urgency. Some other examples include AI deep learning to recognize visual symptoms: here researchers have been using AI to scan and diagnose congenital cataracts, where an estimated 200.000 children are bilaterally blind from cataracts annually [33]. In some of the AI developments blockchain technology is used to ensure trust in data stored in the system [33]. Andy Chun explains that Alibaba and Tencent are investing to interpret scans and to detect early signs of cancer [33]. In July 2019 the Chinese startup JF Healthcare -specialized in providing remote diagnosis services for rural town hospitals where radiologists are not available often- was the first to beat Stanford University radiologists. This approach to AI development is based on experimenting and innovating first and it seeks to achieve time to market results in fields as important as medical care [33] Here AI is seen as a possible solution to doctor shortage -China has two practicing doctors for 1000 inhabitants- to scarce medical services in rural areas, and to highly strained services in rural areas due to large patient volumes [33].

G. Coercion

In our Dashboard on sources of legitimacy and control, coercion is the following category existing in quantitative analysis. For the purpose of our model, albeit with a difference, we draw a similarity between coercion and nudging. There is nudging attached to wearable technology, with over 52 per cent of inhabitants in China using this technology able to monitor their health. Insurance companies such as Ping An Health have integrated wearables into their offers to facilitate discounts and rewards to customers sharing data and living healthier lifestyles [33].

China's privacy is at risk by the lack of rights and guarantees [41][35]. This draws a fine line with coercion [43][37][38][39], even though some data privacy efforts have been addressed in laws and regulations [22][40][41]. AI education systems are an example. Chinese AI educational systems have been collecting, storing and analyzing students' facial expressions without regulation [19]. Facial information has been also collected through boards subways, enforced recycling and the obtention of toilet paper in public toilets, raising many public concerns and establishing a culture of pervasive individuals monitoring [19]. There is no limit in government access to and use of private data. At the same time, without transparency of knowledge about how the variables to calculate scores work, the possible divide between low and high scores may increase.

H. Personal relations and personal quality of the power holder

Both traits, personal relations and personal quality of the power holder, are part of the model developed to explain legitimacy and governance of AI. However, for the case of China the author finds that both features lack power to explain AI development, as the search for related keywords yielded invalid or no significant results. Thus,

both features rank as absent in Table III, defining sources of legitimacy and control applied to China.

VI. Conclusion and Future Work

The work presented in this article allows us to unveil a complex reality from the perspective of philosophy, political theory and sociology, where AI brings new rules attached to command, control and governance in general. One of these new rules is human pace in decision making, in contrast to decisions being made quickly, as they are generally in AI frameworks [43]. The article presents the Gil Dashboard to show how AI is mediating problems related to governance and legitimacy. The Dashboard brings to light new ways of thinking in methodological terms and in comparative perspective about artificial intelligence in different political and social settings. The article argues that the theoretical Dashboard is useful to apply in case studies, multilevel analysis and for comparative perspectives; in countries in Asia, western countries and countries in the global south.

Once the Dashboard is presented, it has been applied empirically to the case of China. Firstly, using a binary code with the sole purpose to state existence or absence of the dimension studied. Additionally, with the references and works covered by the author analyzing 170 documents matching the query, the following conclusions are highlighted for this particular case: Researching on convention, a focus on local-level integrations of artificial intelligence across a broad range of public services has been founded, including judicial services, medical care, and public security. The use of AI for these services affect the mode of social governance on privacy, limiting it. The use of AI for services also affects private companies' participation in social credit systems. Artificial intelligence scoring systems are being used to assign credit to the population with the aim, according to the government, to improve societal trust albeit with limited usefulness, and with associated pitfalls linked to privacy. Contractualism has been found absent as an instrument of legitimacy in China. Legitimacy in conformity with a natural law has been found linked to ethics as a set of laws -an appeal that is shared by private companies in western countries as a main resource towards a self legitimation of artificial intelligence use. A broad development of artificial intelligence is found outside the umbrella of the Chinese state, based on ethics codes. In China's approach to ethics, legitimacy is attached to harmony as a principle in Chinese philosophy for all life forms. At the economic level, multi-stakeholder committees have outlined artificial intelligence ethical principles, many of them according to global standards. Both sacredness of authority and improving democracy through method have been found absent as legitimacy resources. In contrast, legitimation by expertise is deeply ingrained on the basis of economic governance in China. Legitimation by expertise is heavily linked to the artificial intelligence strategic plan designed by the China's State Council in 2017, and to subsequent developments: Chinese officials with technical expertise occupy 81 seats, nearly 40% of the total, in the new Central Committee elected in 2023—the elite body that decides major national policies, up from 18% in the previous Central Committee. Legitimation by expertise is also on the basis of the judiciary, with the creation of System 2016, the contentious Shanghai High People's Court Intelligence assistive case-handling system for criminal cases. Finally, artificial intelligence in the form of NLP and deep learning have also been used extensively to work with relevant information from 1,4 million patients to help frontline patient care, including artificial intelligence deep learning to recognize visual symptoms.

Future works may refine limitations derived from using binary code when the Dashboard proposed is applied to particular cases. In order to avoid this limitation, this coding could be complemented with normalized scales –i.e. Likert scale- and other metrics showing further comparative scalability for each of the dimensions in the Dashboard. Future works may also consider the shift of the Chinese government towards regulation of generative AI, with the new policies of the Chinese Minister of Science and Technology in the summer of 2023. This is an important departure from previous hands off policies towards AI regulation. For future works and research, the Gil Dashboard presented may further help to ask relevant questions on challenges in current societies; from uncertainties that countries face against AI development to challenges based on cultural values including those related to democratic realms, and challenges due to the intersection between local values and AI development.

Acknowledgments

The author wants to thank María Llanos Robledano for the revision of the final manuscript. The author acknowledges the comments of five anonymous reviewers, which helped to improve the final version when the paper was accepted for presentation at IARIA Annual Congress on Frontiers in Science, Technology, Services, and Applications, Venice, Italy April 24 - 28. The author also wants to acknowledge the comments of three anonymous reviewers, which helped to improve the final manuscript for the journal. The author also wants to acknowledge the inspiration and lessons learned from Prof. Joaquín Abellán, teaching together at the Master on Political Theory and Democratic Culture at UCM in the 2000-2022 editions, and to thank Prof. Carmelo Moreno´s comments to earlier work at the 2022 AECPA Congress in Girona (Spain).

References

[1] Gil, Olga. AI Philosophy: Sources of Legitimacy to Analyze Artificial Intelligence. The Sixth International Conference on Advances in Computer-Human Interactions (ICDS - ACHI 2013) IARIA, April 24-28, 2023 - Venice, Italy. 2023, ISSN: 2308-4138, ISBN: 978-1-61208-250-9, doi: 10.31219/osf.io/39njx

[2] Contera, Sonia. (2022, February 22) La nanotecnología llega a la vida [video]. https://youtu.be/vAYAnI6kqrY. Last retrieved November 14th, 2023.

[3] US Department of Commerce. Commerce Implements New Export Controls on Advanced Computing and Semiconductor Manufacturing Items to the People's Republic of China (PRC). October 7, 2022.

[4] Ryan, Maria. How Taiwan's tiny chips are quietly shaping US geopolitics, The Conversation. 22 August, 7 https://interestingengineering.com/culture/taiwan-tiny-chips-shape-us-geopolitics. Last retrieved November 14th, 2023.

[5] Xinhua, (CPC Congress) Full text of resolution on Party Constitution amendment. Oct., 22th., 2022. https://english.news.cn/20221022/fea670f419d7426ab564a795d5737b52/c.html. Last retrieved November 14th, 2023.

[6] Abellán, Joaquín. El político y el científico: Weber. Madrid, Alianza Editorial. 2021.

[7] Matheson, Craig. Weber and the Classification of Forms of Legitimacy. British Journal of Sociology, pp. 199-215, 1987.

[8] World Bank Group. Harnessing artificial intelligence for development in the post-covid-19 era. A Review of National AI Strategies and Policies. May, 2021.

[9] Eichler, William. Shockingly small number of councils embrace automation. LocalGov. 10 May 2019. https://www.localgov.co.uk/Shockingly-small-number-of-councils-embrace-automation-study-reveals/47387 Last retrieved November 14th, 2023.

[10] Justo-Hanani, Ronit. The politics of Artificial Intelligence regulation and governance reform in the European Union. Policy Sciences, vol. 55, no 1, pp. 137-159, 2022.

[11] Ding, Jeffrey. Promoting nationally, acting locally: China´s next generation AI approach. In NESTA. The AI Powered State. China´s Approach to public innovation, pp. 11-17, 2020.

[12] Ding, Jeffrey. Deciphering China's AI Dream: The Context, Components, Capabilities, and Consequences of China's Strategy to Lead the World in AI. Future of Humanity Institute, Oxford University. 2018

[13] Hillman, Noel L. The Use of Artificial Intelligence in Gauging the Risk of Recidivism, The Judges, Journal 58, no. 1, pp. 36-39, 2019.

[14] Naughton, Barry. Chinese Industrial Policy and the Digital Silk Road: The Case of Alibaba in Malaysia, Asia Policy 27, no. 1, pp. 23-39, 2020.

[15] Brandt, Loren, and Rawski, Thomas G. Policy, Regulation, and Innovation in China's Electricity and Telecom Industries, in Policy, Regulation and Innovation in China's Electricity and Telecom Industries. Cambridge: Cambridge University Press, 13, pp. 1-51, 2019.

[16] Creemers, Rogier. The ideology behind China´s AI strategy. NESTA. The AI Powered State. China´s approach to public innovation. 2020.

[17] Creemers, Rogier, Triolo, Paul and Webster, Graham. 'Translation: Cybersecurity Law of the People's Republic of China [Effective June 1, 2017]', DigiChina for New America, updated 29 June 2018, https://www.newamerica.org/cybersecurity-initiative/digichina/blog/translation-cybersecurity-law-peoples-republic-china. Last retrieved November 14th, 2023.

[18] Lewis, Dev. Separating Myth from Reality: How China's Social Credit System uses public data for social governance. In NESTA. The AI Powered State. China´s Approach to public innovation. 2020.

[19] Gal, Danit. Perspectives and Approaches in AI Ethics: East Asia, in Oxford Handbook of Ethics of Artificial Intelligence, eds. M. Dubber, F. Pasquale, and S. Das, Oxford: Oxford University Press, 2020.

[20] Kim, Sung-Young. Hybridized Industrial Ecosystems and the Makings of a New Developmental Infrastructure in East Asia's Green Energy Sector, Review of International Political Economy 26, no. 1, pp. 158-182, 2019.

[21] ECNS, Hangzhou Leads Nation in Attracting Overseas Returnees, ECNS Wire, 28 June 2018, http://www.ecns.cn/news/cns-wire/2018-06-28/detail-ifyvrptq6363759.shtml Last retrieved November 14th, 2023.

[22] Creemers, Rogier. The Ideology Behind China's AI Strategy. In NESTA. The AI Powered State. China's Approach to Public innovation. 2020.

[23] Gil, Olga. Philosophy and Public Administration in China and Western Countries. Intel Prop Rights. Vol. 11 Iss. 1 No: 1000216, 2023, pp. 1-11.

[24] Ongaro, Edoardo. Philosophy and Public Administration. An Introduction. Northampton: Edward Elgar Publishing. 2018.

[25] Jun, Jong S. Social Construction of Public Administration, The: Interpretive and Critical Perspectives. SUNY Press, 2012, pp 83.

[26] Stanley, Isaac, Alex Glennie, and Madeleine Gabriel. How inclusive is innovation policy, Insights from an international comparison, London: Nesta, 2018.

[27] Gianni, Robert, Santtu Lehtinen, and Mika Nieminen. "Governance of responsible AI: from ethical guidelines to cooperative policies." Frontiers in Computer Science 4, 2022.

[28] Roberts, Huw, Josh Cowls, Emmie Hine, Jessica Morley, Vincent Wang, Mariarosaria Taddeo, and Luciano Floridi. Governing artificial intelligence in China and the European Union: comparing aims and promoting ethical outcomes. The Information Society, pp. 1-19, 2022.

[29] von Carnap, Kai, Shi-Kupfer, Kristin. Values as a Quality Seal: Germany Should be More Active in Shaping Cooperation with China, MERICS blog: European Voices on China, 4 September 2019, Mercator Institute for China Studies (MERICS), https://www.merics.org/en/blog/values-quality-seal-germany-should-be-more-active-shaping-Cooperation-china Last retrieved November 14th, 2023.

[30] Laskai, Lorand and Webster, Graham. Translation: Chinese Expert Group Offers Governance Principles for Responsible AI, DigiChina for New America, 17 June, 2019,

https://www.newamerica.org/cybersecurity-initiative/digichina/blog/translation-chinese-expert-group-offers-governance-principles-responsible-ai/ Last retrieved November 14th, 2023.

[31] Li, Chenyang. 'The Ideal of Harmony in Ancient Chinese and Greek Philosophy', Dao 7, pp. 81-98, 2008.

[32] Laskai, Lorand and Webster, Graham. Translation: Chinese Expert Group Offers Governance Principles for "Responsible AI, DigiChina for New America, 17 June 2019, https://www.newamerica.org/cybersecurity-initiative/digichina/blog/translation-chinese-expert-group-offers-governance-principles-responsible-ai/. Last retrieved November 14th, 2023.

[33] State Council, New Generation Artificial Intelligence Development Plan, 2017. https://www.newamerica.org/cybersecurity-initiative/blog/chinas-plan-lead-ai-purpose-prospects-and-problems/ (Full English translation by a group of experienced Chinese linguists with deep backgrounds on the subject matter, see this document from the New America foundation). Last retrieved November 14th, 2023.

[34] Hao, Karen. China's Xi Stacks Government With Science and Tech Experts Amid Rivalry With U.S. Wall Street Journal, Nov. 18, 2022, https://www.wsj.com/articles/chinas-xi-stacks-government-with-science-and-tech-experts-amid-rivalry-with-u-s-11668772682?utm_source=substack&utm_medium=email Last retrieved November 14th, 2023.

[32] Han, Eugeniu. From Traffic Management to Smart Courts: China's approach to smart cities, in The AI Powered State. In NESTA. China´s Approach to public innovation, pp. 35-41, 2020.

[33] Chun, Andy. Breaking the iron triangle: AI in China's healthcare system. In NESTA. The AI Powered State. China's Approach to Public innovation. pp. 19-27, 2020.

[34] Hu, Minghe. China Issues Rules to Stop Apps from Abusing User's Personal Information in Latest Data Privacy Effort, South China Morning Post, 31 December, 2019, https://www.scmp.com/tech/apps-social/article/3044051/china-issues-rules-stop-apps-abusing-users-personal-information. Last retrieved November 14th, 2023.

[35] State Council, Planning Outline for the Construction of a Social Credit System (2014-2020), China Copyright and Media, 14 June 2014, updated 25 April, 2015. https://chinacopyrightandmedia.wordpress.com/2014/06/14/planning-outline-for-the-construction-of-a-social-credit-system-2014-2020/ Last retrieved November 14th, 2023.

[36] Chinoy, Sahil. The Racist History Behind Facial Recognition', New York Times, 10 July, 2019, https://www.nytimes.com/2019/07/10/opinion/facial-recognition-race.html. Last retrieved November 14th, 2023.

[37] Elgan, Mike. 'Uh-oh: Silicon Valley is Building a Chinese-Style Social Credit System', Fast Company, 26 August 2019, https://www.fastcompany.com/90394048/uh-oh-silicon-valley-is-building-a-chinese-style-social-credit-system. Last retrieved November 14th, 2023.

[38] Chokshi, Niraj. Facial Recognition's Many Controversies, From Stadium Surveillance to Racist Software, New York Times, 15 May, 2019, https://www.nytimes.com/2019/05/15/business/facial-recognition-software-controversy.html. Last retrieved November 14th, 2023.

[39] White, Geoff. Use of Facial Recognition Tech "Dangerously Irresponsible, BBC News, 13 May, 2019, https://www.bbc.com/news/technology-48222017. Last retrieved November 14th, 2023.

[40] Shi, Mingli, Samm Sacks, Qiheng Chen, and Graham Webster, Translation: China's Personal Information Security Specification, DigiChina for New America, 8 February 2019, https://www.newamerica.org/cybersecurity-initiative/digichina/blog/translation-chinas-personal-information-security-specification/. Last retrieved November 14th, 2023.

[41] Hu, Yue, Changfeng Chen, Hongnan Liu and Kai Zhou, Research on the Impact of AI and Big Data on the Trade Between China and other Countries along the Belt: Take the Application of Medical Economy as an Example, Changchun: SSER, 2019. https://webofproceedings.org/proceedings_series/ESSP/SSER%202019/SSER30212.pdf Last retrieved November 14th, 2023.

[42] Liu, Yi-Ling. The Future of the Classroom? China's experience of AI in education, in The AI Powered State. China´s Approach to Public Innovation, pp. 27-34, 2020.

[43] Gianni, Letizia. Democratic Accountability in Stressful Times: When Decisions Must Be Made Quickly. Penn State Journal of Law & International Affairs, vol. 11, no 1, p. 1, 2023.

How little code is low-code? - Towards productivity measures for the use of low-code development platforms by business user developers

Olga Levina
Department of Business Management
Brandenburg University of Applied Sciences
Brandenburg an der Havel, Germany
email: olga.levina@th-brandenburg.de

Katharina Frosch
Department of Business Management
Brandenburg University of Applied Sciences
Brandenburg an der Havel, Germany
email: frosch@th-brandenburg.de

***Abstract*— Low code development platforms (LCDP) often promise an easy and fast way to include data processing and support into the otherwise non-digital process. This research explores how to measure the productivity of low code development to assess the effort needed for business users to respond to their need for support via these tools. We chose field experiments as a research method to evaluate the feasibility and derive the metrics for software development with LCDP by novices. The paper provides some insights on how these measures can be implemented in practice, how to support business unit developers to efficiently deliver productive results, and how to evaluate LCDP-based development processes.**

Keywords- low code development platforms; software development process; digital novices, productivity; performance indicators

I. INTRODUCTION

Demand for data management solutions in a business context, coupled with the challenge of modernizing legacy systems is fueling the innovation of new software development tools and methods. To create an application or be productive in manual coding, the programmers need to be skilled in specific programming languages. As skilled IT staff is scarce, this development creates a positive environment for the adoption of Low Code Development Platforms (LCDPs). This paper builds on the findings by [1] in the context of Business User Development of business applications using LCDPs. While this previous research explored the suitability of LCDPs to answer the data management needs of business users and the platform's potential to provide them with a satisfactory development tool turning business users into Business User Developers, the expansion of this research focuses on the determination of the productivity of the LCDP use in a specific business software development project.

LCDPs promise an easy and fast possibility to include data processing and support in the otherwise non-digital process [2]. The terms "citizen developer" or "Business Unit Developer" (BUD) [3] are often used in the LCDP context to underline the potential of the software tools to involve programming novices in the development of solutions for their needs [4].

Low-code platforms abstain as far as possible from using textual programming that requires manual coding and offer instead visual or, less often, natural languages [5]. As a result, developing applications using low-code technologies is faster and may result in swifter delivery and higher productivity [6]. Thus, this research addresses the following research questions: How can the effort needed to create an application with an LCDP by BUDs be assessed? As well as, how can the effort for software development using a programming language versus the development of the same requirements using LCDPs be compared?

As LCDPs were shown by [1] to be a usable tool for novices to address their digitalization needs, this research expands this question and enriches the usage and implementation of LCDPs by providing indicators for effort assessment in the context of software development projects.

In particular, we suggest a metric for the evaluation of LCDPs in terms of programming effort – the Low Code Factor (LCF), which is defined as the number of actions taken by the developers on the LCDP per use case. This metric will allow an assessment of LCDPs in terms of their effectiveness in fulfilling the digitization needs of the BUDs. It is based on UCPA (Use Case Points Analysis) [7], the effort assessment method for object-oriented software development projects, which we extend by the user interaction data with the LCDP.

As the research method, we use an experimental setting, where software application requirements are derived and documented by BUDs. Then we let BUDs create applications using an open source LCDP Joget. Based on LCDP activity logs, we evaluate the effort invested by BUDs to develop an application with an LCDP. As a novel contribution, we suggest LCF as a measure of BUD's productivity on the LCDP. A further metric, LCDPfit aims to provide project managers with means for the assessment of the project size and effort needed to complete the project. Also, a cost-benefit calculation of the planned software

realization using the two different approaches (low code vs. classic software development) can now be achieved.

The paper is structured as follows: First, we review the current literature on how LCDPs are currently used in a business context, and what methods are commonly used for productivity assessment in software development. Then we derive productivity measures, in particular the LCF, that we then apply in our experimental setting. The results obtained from the analysis lead to recommendations for implementing LCDPs in a productive environment. We close with a summary and outlook on future research.

II. Related Work

A. Use of LCDP

The use of LCDPs in different business domains has been increasingly the focus of research in the last few years. Sanchis et al. [8] showed that rapidity and cost reduction through intuitive development and management can be attributed to the use of an LCDP in a manufacturing context. Nowak et al. [9] showcase the usage of LCDPs in the context of the internal logistics processes in a company from the E-Commerce industry. This case study is meant to display the use of LCDPs in the context of process improvement as it allows for the direct elimination of found limitations in processes. The authors argue that the implementation of the IT support using LCDPs was effective, i.e., an enhancement in terms of time and costs needed for its realization.

Bies et al. [10] conducted a mixed-method study to identify challenges and promising perspectives for digital innovations in small and medium-sized enterprises (SMEs). The authors found that the application areas of LCDPs are mostly of a supportive nature such as the creation of applications for resource management or the creation of customized digital forms. Nevertheless, the majority of the surveyed SMEs stated LCDPs to be of high to very high relevance. Factors that diminish the relevance of low code in SMEs are according to the authors: limited human resources, as personnel is still necessary to develop and maintain the application, knowledge transfer between the platforms as well as training in dealing with IT structures and detailed knowledge of the platforms.

Lethbridge [11] also explores the development process of the software product as well as the aspects of implementation and maintenance of the LCDP software within the existing enterprise architecture. His findings suggest that LCDPs create “technical debts” that can be overcome by the development of the LCDP towards “scaling, understandability, documentarily, usability, vendor- independence and user experience for the developers”. Hintsch et al. 2021 [12] also identify threats and opportunities in the LCDP development concerning the security and availability of the created applications. Nevertheless, the authors also uncover success factors for LCDP use in a business context by novices.

Kermanchi et al. [13] focus in their research on software development methods and the use of LCDPs. In their experiment, they explored the episodic experience with different LCDPs among software developers with varying levels of programming experience but no experience in the specific LCDP. The findings show that previous programming experience seems to have a significant impact on developers' performance, experiences, and tool preferences, yet most developers continue to have doubts about the scalability and maintainability of applications created with LCDPs. The opinions on the effectiveness of the instruments vary among the participants.

Bernsteiner et al. [14] conduct expert interviews in their research to investigate what skills developers with little or no software development experience, i.e., novices, need to successfully develop software on LCDPs. Several of the interviewed experts mention that successfully developing an LCDP solution requires at least basic programming skills. This is in line with research findings stating that LCDPs still require some prerequisites in software development [15] or in database structures [16], which hampers the adoption of LCDPs by non-programmers without any further training.

Krejci et al. [16] report in a case study how non-IT employees were involved in the process of digital innovation while making efficient use of their IT resources. These citizen developers, i.e., employees who are working outside of the Information Technology (IT) department and are not professional programmers, as users of LCDPs are the focus of the analysis by Lebens et al. [17]. The authors surveyed the use of LCDPs in organizations. The results show that companies both large and small are making use of low- and no-code platforms. Additionally, the majority of the surveyed organizations have employees outside of the IT department who are creating IT solutions.

Bock and Frank [18] provide a critical overview of the LCPD features, architecture, and opportunities while pointing out research directions for information systems research in this domain. They state that although both professional developers and citizen developers use LCDPs, there is a lack of research on how to make LCDPs fit the cognitive capabilities and personal working styles of these two groups [p. 739]. This is in line with other studies pointing out that successfully developing software on LCDPs requires at least basic programming skills.

The use of development templates in the context of software creation is analyzed by Boot et al. [19]. The authors compare instructional software products made by developers with low production experience and high production experience, working with a template-based authoring tool. The analysis showed that the technical and authoring quality was equal for both groups, indicating that templates enable domain specialists to participate successfully in the production process. Research in agile software development shows that projects based on the Scrum methodology profit from having a coach on the team [20]. The same is visible in software engineering education [21].

BUDs and job crafting, i.e., proactive strategies to improve work processes according to one’s own needs and goals, are subjects of the analysis by Li et al. [3]. The authors found that using LCDPs provides positive job

crafting consequences such as meaningfulness, for the employees using these tools [3], [22]. In what follows, we prefer to use the term BUDs instead of citizen developers, stressing that they might make up for the lack of programming skills with their large expertise in the respective business domain. Nevertheless, the research does not focus on the description of how much support was needed for BUDs to finish their application.

In conclusion, in these first attempts to understand the "human side" of LCDPs, research is still scarce concerning acceptance and successful adoption by domain experts outside corporate IT departments. We also lack information on how effective (or productive) BUDs are in using LCDPs to fulfill their own digital business needs.

B. Productivity assessment in software development

How widely LCDPs will be used in enterprise context by BUDs without sound programming expertise might also depend on the productivity they can achieve with the respective tool.

The concept of productivity in software development is not new to the domain and has been studied from various perspectives. However, there is still no consensus within academic and industry circles, as some researchers argue that using a single metric to measure productivity can lead to problematic and misleading [6] assessments.

Hence, among the methods to assess productivity in software development are lines of code [23]; function point analysis [24]; and Use Case Points Analysis (UCPA). UCPA was developed in the context of object-oriented software development by Gustav Karner (see e.g., [7], [25]) and is similar to the function point analysis. We chose UCPA in our study following [6] as it provides a way to estimate the size and complexity of a software development project early on, based solely on requirements [26]. UCPA leverages use cases representing functional requirements as its starting point. The resulting Use Case Points (UCP) metric reflects the complexity of the project across three dimensions - functional, technical, and environmental, i.e., considering the context of the project. Thus, UCPA allows sizing and estimation of the effort required for a software development project.

Hence, to assess productivity in a software development project, we lend the definition from the economics discipline and define software development productivity simply as the ratio of outputs produced to the inputs involved in that production, also following [6], [27], [28] who use this definition in the software development domain. In the context of application development, input is defined as the time and activities invested in the development and the output will be the implemented use cases.

While UCPA presents a good tool for manual programming effort assessment, it does not account for the potential that the LCDPs are providing for the development project.

Given the research activities in the areas of LCDP usage in the business context, especially among BUDs, as well as the nature of finding a digital solution to a business problem being a software development project, the following research questions are identified:

- RQ1: How can the effort needed to create an application with LCDPs by BUDs be assessed?
- RQ2: How can the effort for software development using a programming language versus the development of the same requirements using an LCDP be compared?

III. RESEARCH METHOD

To answer the research questions, experiments were set up with the Master's students of Business Management and Information Systems. The goal of the experiments was to assess the effectiveness of the app development using the LCDP Joget, which is described further in [1]. Therefore, different scenarios requiring digital support were suggested for the students for their implementation in the app, using the LCDP. Based on the application design that was documented in activity diagrams, user stories, and mockups, as well as based on data logs from the experiment, the productivity metric was derived.

A. Data collection based on field experiments

To gain evidence for answering our research questions, we draw upon a field experiment where BUDs with little prerequisites in software development build app prototypes in the business domain of human resource management (HRM) based on an LCDP given a finite time frame of a few weeks. Overall, 13 HR apps have been developed.

The LCDP used for the experiment was Joget [29], an open-source LCDP with the promise to easily build, run, and maintain apps. A visual builder allows drag-and-drop for pages, forms, views, data lists, menus, and a process builder to automate workflows. It also offers user management and role-based authentication. We used the community edition that can be self-hosted at no license cost.

BUDs were Master's students of business management with a specialization in human resources management (HR) and Master's students of information systems management (ISM). All of the ISM students had already taken at least one course in advanced software engineering within their Master's program at the time of the experiment but were far from being experienced software developers. The HR students had no previous expertise in software development. None of the participants in either group was familiar with or had heard of the LCDP selected for the experiment. Figure 1 presents the data collection process and the sequence of the experiments.

The experiment was divided into four self-contained challenges with modified compositions of participants. The challenges are described below.

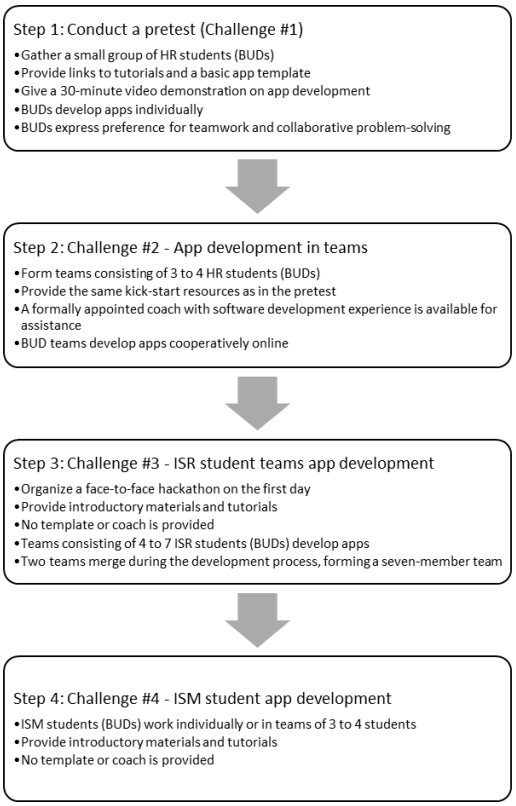


Figure 1. Process of the data collection in the field experiments

For the first two challenges, BUDs are Master's students of business management with a specialization in HR (six and 16 students, respectively). They sketched their app concept and subsequently implemented it. In the third and the fourth challenge master students of information systems management (ISM) were provided with ready-prepared HR app concepts and then asked to implement them (24 and eight ISM students, respectively). The apps were from the HR domain but otherwise differed in their content and scope.

Challenge #1 was run with a few HR students as BUDs (six) only, to have a pretest and check whether they are, at all, able to use the LCDP to develop simple apps. The pretest was run between April 21 and June 6, 2021 (47 days). To kick start app development BUDs were provided with links to tutorials as well as with a basic app template and a 30-minute video showing exemplarily how an app can be built starting from this template. In this context, they were also explicitly pointed to the open-source character of app development in this setting, and about the possibility to share and reuse app elements from other groups. In the pretest, BUDs managed to develop apps but pointed out that they would have enjoyed working in teams to solve problems collaboratively. Furthermore, support from one student who previously had graduated from a Bachelor's program in software engineering and acted as an informal coach for his fellow students has been acknowledged as extremely helpful.

Based on the insights gained in the pretest, we recruited the informal coach from challenge #1 to act as a formally appointed coach in challenge #2 and decided to run development in teams. For challenge #2 BUD teams (with three to four HR students, 16 in total) developed their apps within six weeks between November 1 and December 12, 2021 (42 days). The team members cooperated online, due to the restrictions because of the COVID-19 pandemic. Developers got the same kick start as in the pretest and were also pointed towards the template and the possibility to share and reuse apps. Furthermore, a coach with experience in software development was available to get help with questions on tool usage and minor development questions.

In Challenge #3, 24 BUDs in teams of four to seven ISR students developed their apps between May 20 and June 7, 2022 (19 days). The first day of the development phase (May 20, 2022) was organized as a face-to-face daylong hackathon. The introductory video and tutorials were made available beforehand, but no template or coach was provided for the teams. During the development challenge, two teams joined forces within the development process, resulting in a seven-member team working on the challenge.

Challenge #4 was a replica of challenge #3 with 8 ISM students acting as BUDs, where one worked alone and the others in teams of three or four students between May 24 and June 28, 2022 (36 days).

The effectiveness of using the LCDP to solve the business needs for BUDs was described in [1]. In this research, the focus is on the description of the productivity metric for the project assessment as well as for the assessment of the suitability of the LCDP for solving business-related questions compared to the software development using a programming language.

B. Measuring the coding effort on the LCDP

To evaluate efforts made by BUDs to develop their app, the LCDP activity logs were archived and anonymized. These data were used to calculate indicators to measure the effort invested in app development based on the LCDP. Note that there is no log data available for challenge #1. We use the following indicators related to time spent on the platform and the number of actions:

- *Time on the platform (hours)*: Total time a developer was active on the LCDP during the developing stage. Based on the first and the last action performed for each login identified, we can compute the duration users are active per login. Idle periods of 30 minutes or longer are omitted, assuming that the user then has stopped developing. Summing up yields the total time on the platform in hours.
- *Time investment (hours)*: Total time invested by app is obtained by summing up hours spent on the platform across all members of the developer team of the respective app.

- *Number of actions, by developer*: For this variable, we count actions taken by each developer, such as creating, editing, and deleting code, forms, views, or other assets.
- *Number of actions, by app*: Aggregation of actions undertaken by all members of the development team of the respective app.

As the duration of the development phase and team size vary across challenges, app-based indicators for effort invested are more informative as compared to effort indicators at the level of individual developers.

Using these indicators and relying on the methods for productivity measurement in software engineering described in section II, the LCDP-related productivity factors *LCF* and *LCDPfit* were derived and calculated.

C. The Low Code Factor (LCF)

The calculation of the *LCF* is based on the UCPA method. This method considers users involved in the interaction with the software as well as the interaction patterns, i.e., use cases of these actors. The UCPA method consists of several stages, see e.g., [25]:

First, the actors (roles interacting with the system) and use cases need to be identified. Then, the actors need to be classified into one of three categories based on the complexity of interaction with the system according to [25]:

- Simple actor- e.g., system interface, weight 1
- Average actor- e.g., protocol-driven interface, weight 2
- Complex actor- e.g., GUI, weight 3

Then, each use case needs to be classified based on its integration complexity as simple, average, or complex. Complexity assessment is based on aspects such as transactions, i.e., communication, information exchange, or data access, etc.

- Integrated use cases: already implemented transactions in the LCDP, weight 0
- Simple use cases: 1-3 transactions, <5 classes, weight 5
- Average use case: 4-7 transactions, 6-10 classes weight 10
- Complex use case: >8 transactions, >11 classes, weight 15

As LCDPs already provide some implemented interaction patterns, we suggest a new class of use cases that is specific to the use of LCDPs: the *integrated use case* with the weight 0, as no programming effort is required to implement this use case. Also, since the app development project was based on the LCDP-based development, the UML classes as referred to in UCPA were realized as "data lists" in Joget terms.

After the classification of the use cases, the productivity indicators need to be calculated (see [30] for calculation details):

- Unadjusted use case points (*UUCP*) are calculated as the sum of the unadjusted actor weight (*UAW*) and unadjusted use case weight (*UUCW*):
  - $UUCP = UAW + UUCW$
  - *UUCW* is calculated by multiplying the number of each use case type by a weighting factor according to its classification.

UAW is calculated by multiplying the number of actors by the weighting factor.

Now, UUCP needs to be adjusted using technical complexity factors (*TCF*) and environmental complexity factors (*ECF*) to derive adjusted use case points (*UCP*).

- The combination of the *UUCP* variable with the *TCP* and EF variables results in the actual number of *UCP* of the project:
  - $UCP = UUCP \times TCF \times ECF$
- TCF is one of the factors applied to the estimated size of the software to account for technical considerations of the system. It is determined by assigning a score between 0 (factor is irrelevant) and 5 (factor is essential) to each of the 13 technical factors. This score is then multiplied by the defined weighted value for each factor (*TF*).
  - $TCF = 0.6 + (TF/100)$
- ECF is determined by assigning a score between 0 (no experience) and 5 (expert) to each of the 8 environmental factors. This score is then multiplied by the defined weighted value for each factor (*EF*):
  - $ECF = 1.4 + (-0.03 \times EF)$

This value is multiplied by the productivity factor (*PF*), which represents the number of hours required to develop each *UCP*:

- $Total\ Effort = UCP \times PF$.

Tables II and III provide the calculations for selected apps from challenge #3. In sum, the productivity assessment in our context considers *UCP* as the output measure and PF as the input measure. To assess the productivity of the LCDP-based app development, we introduce the *LCF* and *LCDPfit* metrics that are based on the platform log data that was generated per app.

The Low Code Factor (*LCF*) assesses the effort submitted versus the functional complexity required for the realization of the business solution that is calculated using UCPA. To calculate the *LCF* we derive the number of actions performed per app (see Table I) and divide them per weighted use cases *UCP*. Thus, it provides the measurement of the platform interaction needed to realize the use cases. *LCDPfit* is calculated as the quotient of the number of lines of code needed to realize the app despite using an LCDP and the UCP. Thus, the *LCDPfit* provides an assessment of the programming effort required despite using the LCDP,

while *LCF* assesses the effort of the platform interaction for the realization of the app.

Calculation and interpretation of *LCF* and *LCDPfit* for productivity assessment across the presented challenges are described in the following section.

IV. RESULTS

The experiment has shown that in all challenges, BUDs were able to create a software application using an LCDP in a given amount of time without any (challenges #1 and #2) or at least no extensive professional training (challenges #3 and #4) in software development, see also [1]. All apps created during the challenges have been successfully developed and implemented. "Successfully" means that they met the requirements depicted in the conceptual papers and that 13 apps worked when tested. The technology readiness of the prototypes corresponds to level 3 (experimental proof of concept) according to the European Union Technology Readiness Levels [31].

Overall, our data comprises 568 logins, resulting in 10,395 actions taken, respectively. The distribution of time spent on the platform is right-skewed, with most developers investing not more than 10 hours in development. Moreover, we observe two outliers with more than 60 (challenge #2) and more than 30 (challenge #3) hours, respectively. When analyzing effort at the level of developers, comparing means may lead to misleading results whereas modal values provide a more robust measure for typical development effort.

To gain more insights into what effort is needed to develop a business app using LCDP and analyze time spent on the platform and the number of actions taken by the app for each of the 13 apps that have been created across challenges #2 to #4 (Table 1).

TABLE I. EFFORT PER APP

App	Challenge	Total time	No. of actions
1	#2	20.28	929
2	#2	23.72	608
3	#2	81.41	2454
4	#2	25.13	807
5	#2	19.91	417
6	#3	30.91	951
7	#3	43.92	1344
8	#3	19.18	373
9	#3	30.03	946
10	#3	17.5	686
11	#4	10.92	363
12	#4	12.91	299
13	#4	14.51	207

Table 1 shows that the number of actions taken per app and time investment for development by app varies considerably. However, effort invested by the app does not necessarily seem to depend on previous programming expertise, as on average, the completely unexperienced BUDs in challenge #2 show a medium effort level concerning both, time and number of actions as compared to the somewhat experienced BUDs in challenges #3 (higher effort levels) and #4 (lower effort levels).

In the next step, we undertake productivity assessments for each of the 13 apps developed across challenges #2 to #4 using the suggested metric, the low code factor (*LCF*). This measurement will allow us to assess the effort submitted versus the functional complexity required for the realization of the business solution that is calculated using *UCPA*.

To assess the development productivity, *LCF* and *LCDPfit* are calculated. Table II shows the use cases and weights of the apps 6 –8 as well as their Technical Complexity Factor (*TCF*), Environmental Complexity Factor (*ECF*) as well as the productivity factor that is calculated as the quotient of the total effort (time spent on the app) and the weighted *UCP*.

TABLE II. UCPA CALCULATION OF THE APPS

App	No. of actors	Use Case	Weight
6	3	user login	2
		solve quiz	5
		view score	10
		view detailed score	10
		view feedback	10
		see score per applicant	10
		generate user	5
		manage questions	5
		manage evaluation guides	10
	UUCP	67	
7	3	login	5
		upload doc	5
		solve task	0
		view results	10
		view doc	5
		provide task	10
		check results	10
		send feedback	10
		CRUD results	15
		CRUD users;	15
		creates tasks	10
		solves tasks	5
	UUCP	100	
8	4	solve quiz	5
		view score	10
		view score per applicant	10
		generate evaluation	5
		manager users	10
	UUCP	40	

Table III shows the TCF and ECF of some of the apps as well the UCP according to the calculation of UUCP and adjusting it with the TCF and ECF:

- *UUCP = UAW + UUCP*
- *UCP = UUCP×TCF×ECF*

The productivity factor was calculated using the time spent per app from Table I and the *UCP* value.

TABLE III. METRIC OF THE APPS 6-8

App	*TCF*	*ECF*	*UCP*	*PF*	*LCF*	*LCDPfit*
6	1.02	1.1	84.85	0.36	11.21	9.09
7	1.02	1.1	123.36	0.36	10.90	3.55
8	1.02	1.1	58.06	0.33	6.42	5.34

Furthermore, Table III shows the *LCF* and the *LCDP*fit metrics for the selected apps. Assessment and data for all 13 apps are provided in the dataset at Zenodo [32].

The selected apps were designed by three different BUD teams according to the general requirements to build a mini assessment center for an HR responsible. Besides this general description, each team was supported by a "customer", i.e., an HR Master student who derived the requirements for the app and was supervising their implementation. All three teams did not have any previous knowledge of the LCDP in question, i.e., Joget, encountered similar values of the *TCF* and *ECF* in the UCP calculation. Despite similar basic conditions, the teams fulfilled their task with different functional extenuations. While app 7 realized twelve of the required use cases, team 8 realized five and team 6 nine use cases. Nevertheless, the teams showed similar productivity factors (see Table III). The efficiency of the LCDP use as indicated by the *LCF* and *LCDPfit* also varied between the teams, with team 6 engaging in the highest programming and LCDP engagement effort as shown by *LCDPfit* and *LCF* metrics respectively, and team 8 showed an efficient use of the platform and its given functionalities as shown by the *LCF*.

V. SUMMARY AND OUTLOOK

Using the results of the described experiment, we can draw the conclusion that BUDs can create their software applications in their business domain using an LCDP, and that time and effort invested in development are not significantly different between BUDs with no and BUDs with some programming knowledge. One interpretation of this result is that the LCDP used is really low code, as it does not seem to make a difference whether developers have no or some prerequisites in software development. Differences in the average effort displayed may for example result from individual performance preferences in the developer teams. Another possible explanation is that the complexity of the apps varied between challenges and also between apps within a challenge.

Besides the suitability of LCDP to support the realization of digitalization by BUDs this paper explored the possibility to measure the productivity of a software developer using LCDP as well as to provide an estimate for the effort needed to compose a business app using a LCDP. Therefore, an experiment with three different challenges was conducted. All the solutions for the challenges led to an app that was ready to be implemented in the business context. Although the quality of the created artifacts was not measured, and the size of the developer groups varied, the research offers valuable insights into the development process using LCDP by both non-IT and IT-trained users.

In addition, this paper presented two indicators to measure LCDP performance within the software development process: Low Code Factor (*LCF*), which measures the software development effort needed for the app creation using an LCDP, and the *LCDPfit*, a metric that can assess the suitability of an LCDP to realize the intended use cases. These metrics and results can be used by managers and practitioners to support an effective and successful LCDP implementation. The applied research method can be expanded by HR and ISM researchers to support their conceptual artifacts in a low-code development context with data. Also, the suggested indicators can be used to assess the process performance of the software development with LCDP.

In our future work, the focus will be on understanding the intensity of the programming activity and how it might reflect a behavioral pattern. This will involve quantifying the motivation of the developer team by using the activity/action profiles of the app development process. Additionally, we envision exploring, how LCDP empowers BUDs within their working environment. Another future research direction will focus on the job-crafting effects of LCDP-based development for BUD and experts.

REFERENCES

[1] K. Frosch and O. Levina, „Taking the Matter in their own Hands – Can Business Unit Developers Fullfill their Digital Demands with Low-Code Development Platforms?," Design and application of socially-aware IT (DASAIT) IARIA, Apr. 2023, no. 18001, ISSN: 2308-3956, ISBN: 978-1-68558-077-3.

[2] S. Rafi, M. A. Akbar, M. Sánchez-Gordón, and R. Colomo-Palacios, „DevOps Practitioners' Perceptions of the Low-code Trend," International Symposium on Empirical Software Engineering and Measurement, pp. 301–306, Sep. 2022, doi: 10.1145/3544902.3546635.

[3] M. M. Li, C. Peters, M. Poser, K. Eilers, and E. Elshan, „ICT-enabled job crafting: How Business Unit Developers use Low-code Development Platforms to craft jobs," International Conference on Information Systems (ICIS), Dec. 2022, ISBN 978-1-958200-04-9.

[4] K. Talesra, „Low-Code Platform for Application Development," International Journal of Applied Engineering Research, vol. 16, pp. 346–351, doi: 10.37622/IJAER/16.5.2021.346-351, 2021.

[5] M. Hirzel, „Low-code programming models," Communications of the ACM, vol. 66, pp. 76–85, 2023.

[6] A. Trigo, J. Varajao, and M. Almeida, „Low-Code Versus Code-Based Software Development: Which Wins the Productivity Game?," IT Professional, vol. 24, pp. 61–68, 2022, doi: 10.1109/MITP.2022.3189880.

[7] J. Smith, „The Estimation of Effort Based on Use Cases," Rational Software, White Paper. [Online]. Available from: https://www.inf.ufpr.br/andrey/ci221/docs/finalTP171.pdf (last accessed: Dec 1, 2023).

[8] R. Sanchis, Ó. García-Perales, F. Fraile, and R. Poler, „Low-code as enabler of digital transformation in manufacturing industry," Applied Sciences, vol. 10, 12, Dec. 2020, doi: 10.3390/app10010012.

[9] F. Nowak, J. Krzywy, and W. Statkiewicz, „Study on the Impact of the Use of No-code Application on Internal Logistics Processes in a Company from the E-Commerce Industry - Process Analysis," European Research Studies Journal, vol. 25, pp. 59–71, Aug. 2022, doi: 10.35808/ERSJ/2936.

[10] L. Bies, M. Weber, T. Greff, and D. Werth, „A Mixed-Methods Study of Low-Code Development Platforms: Drivers of Digital Innovation in SMEs," International Conference on Electrical, Computer, Communications and Mechatronics Engineering (ICECCME), 2022, doi: 10.1109/ICECCME55909.2022.9987920.

[11] T. C. Lethbridge, „Low-Code Is Often High-Code, So We Must Design Low-Code Platforms to Enable Proper Software Engineering," Lecture Notes in Computer Science, vol. 13036, pp. 202–212, 2021, doi: 10.1007/978-3-030-89159-6_14/COVER.

[12] J. Hintsch, D. Staegemann, M. Volk, and K. Turowski, „Low-code Development Platform Usage: Towards Bringing Citizen Development and Enterprise IT into Harmony," ACIS 2021 Proceedings, vol. 10, Jan. 2021. [Online]. Available from: https://aisel.aisnet.org/acis2021/11.

[13] A. Kermanchi, „Developer Experience in Low-Code Versus Traditional Development Platforms - A Comparative Experiment," Aalto University, Dec. 2022 [Online]. Available from: https://aaltodoc.aalto.fi/server/api/core/bitstreams/18f9453c-5930-4a94-a545-4f9dc51be7aa/content (last accessed: Dec 1, 2023).

[14] R. Bernsteiner, S. Schlögl, C. Ploder, T. Dilger, and F. Brecher, „Citizen vs. Professional developers: Differences and Similarities of Skills and Training Requirements for Low Code Development Platform", in ICERI2022 Proceedings, 2022, pp. 4257–4264.

[15] A. Sahay, A. Indamutsa, D. Di Ruscio, and A. Pierantonio, „Supporting the understanding and comparison of low-code development platforms," Euromicro Conference on Software Engineering and Advanced Applications (SEAA), 2020, pp. 171–178.

[16] D. Krejci, L. Küng, and S. Missonier, „A Case Study of Enterprise-wide Digital Innovation: Involving Non-IT Employees", ECIS 2022 Research Papers, June 2022. [Online]. Available from: https://aisel.aisnet.org/ecis2022_rp/55 (last accessed: Dec 1, 2023).

[17] M. Lebens, R. J. Finnegan, S. C. Sorsen, and J. Shah, „Rise of the Citizen Developer," Muma Business Review, Vol. 5, pp. 101–111, 2021, doi: 10.28945/4885.

[18] A. C. Bock and U. Frank, „Low-Code Platform," Business and Information Systems Engineering, vol. 63, pp. 733–740, Dec. 2021, doi: 10.1007/S12599-021-00726-8/FIGURES/1.

[19] E. W. Boot, J. J. G. Van Merriënboer, and A. L. Veerman, „Novice and experienced instructional software developers: Effects on materials created with instructional software templates," Educational Technology Research and Development, vol. 55, pp. 647–666, 2007, doi: 10.1007/s11423-006-9002-9.

[20] C. Bunse, I. Grützner, C. Peper, S. Steinbach-Nordmann, and G. Vollmers, „Coaching professional software developers an experience report," Software Engineering Education Conference, pp. 123–130, 2006, doi: 10.1109/CSEET.2006.11.

[21] H. I. Akyüz and M. Kurt, „Effect of teacher's coaching in online discussion forums on students' perceived self-efficacy for the educational software development," Procedia - Social and Behavioral Sciences, vol. 9, pp. 633–637, Jan. 2010, doi: 10.1016/J.SBSPRO.2010.12.209.

[22] E. Elshan, E. Dickhaut, and P. Ebel, „An Investigation of Why Low Code Platforms Provide Answers and New Challenges," 56th Hawaii International Conference on System Sciences, 2023, ISBN: 978-0-9981331-6-4.

[23] R. Pressman, *Software Quality Engineering: A Practitioner's Approach*, New York, NY, USA: McGraw-Hill Education, 2009.

[24] C. J. Lokan, „Function points," Adv. Comput, vol. 5, pp. 297–347, 2005.

[25] M. K. Chemuturi, *Software Estimation Best Practices, Tools & Techniques*. Fort Lauderdale, J. Ross Publishing, 2009.

[26] R. K. Clemmons, „Project estimation with use case points," J. Defense Softw. Eng., vol. 19, pp. 18–22, 2006.

[27] K. Petersen, „Measuring and predicting software productivity: A systematic map and review," Inf. Softw. Technol., vol. 53, pp. 317–343, 2011.

[28] R. Premraj, B. Kitchenham, M. Shepperd, and P. Forselius, „An empirical analysis of software productivity over time," 11th IEEE International Software Metrics Symposium (METRICS), Sept. 2005, https://doi.org/10.1109/METRICS.2005.8.

[29] Joget. [Online]. Available from: www.joget.org (last accessed: Dec 1, 2023).

[30] M. Ochodek, J. Nawrocki, and K. Kwarciak, „Simplifying effort estimation based on Use Case Points," Information and Software Technology, vol. 53, pp. 200–213, 2011, doi: 10.1016/j.infsof.2010.10.005.

[31] European Commission, Technology readiness levels (TRL), 2014. [Online]. Available from: https://ec.europa.eu/research/participants/data/ref/h2020/

wp/2014_2015/annexes/h2020-wp1415-annex-g-trl_en.pdf (last accessed: Dec 1, 2023).

[32] O. Levina, UCPA for Low Code Factor Calculation. [Online]. Available from: https://zenodo.org/records/8308333, 2023 (last accessed: Dec 1, 2023).

Deep Learning Decision Making for Autonomous Drone Landing in 3D Urban Environment

Oren Gal[1,2] and Yerach Doytsher[3]

[1]Department of Marine Technologies
University of Haifa
Haifa, Israel

[2]Kinneret Academic College
Kinneret
Israel

[3]Mapping and Geo-information Engineering
Technion - Israel Institute of Technology
Haifa, Israel

e-mails: {orengal@alumni.technion.ac.il, doytsher@technion.ac.il}

***Abstract*— Quadcopters are four rotor Vertical Take-Off and Landing (VTOL) Unmanned Aerial Vehicle (UAV) with agile manoeuvring ability, small form factor and light weight – which makes it possible to carry on small platforms. Quadcopters are also used in urban environment for similar reasons – especially the ability to carry on small payloads, instead of using helicopters on larger vehicle which are not possible in these dense places. In this paper, we present a new approach for autonomous landing a quadcopter in 3D urban environment, where the first stage is based on free obstacle environment and maximal visibility for the drone in the palled landing spot. Our approach is based on computer-vision algorithms using markers identification as input for the decision by Stochastic Gradient Descent (SGD) classifier with Neural Network decision making module with greedy motion planner avoiding static and dynamic obstacles in the environment. We use OpenCV with its built-in ArUco module to analyse the camera images and recognize platform/markers, then we use Sci-Kit Learn implementation of SGD classifier to predict landing optimum angle and compare results to manually decide by simple calculations. Our research includes real-time experiments using Parrot Bebop2 quadcopter and the Parrot Sphinx Simulator.**

Keywords - Swarm; Visibility; 3D; Urban environment; autonomous landing.

I. INTRODUCTION AND RELATED WORK

A Quadcopter is a specific type of a UAV, with four rotors and Vertical takeoff and Landing (VTOL) capability, its agility, light weight and size makes it a perfect companion to smaller boats from sail-boats to even kayak, rather than classic helicopters that accompany bigger ships or fixed-wings airplanes on extremely large aircraft carriers.

The efficient computation of visible surfaces and volumes in 3D environments is not a trivial task. The visibility problem has been extensively studied over the last twenty years, due to the importance of visibility in GIS and Geomatics, computer graphics and computer vision, and robotics. Accurate visibility computation in 3D environments is a very complicated task demanding a high computational effort, which could hardly have been done in a very short time using traditional well-known visibility methods [1].

The exact visibility methods are highly complex, and cannot be used for fast applications due to their long computation time. Previous research in visibility computation has been devoted to open environments using DEM models, representing raster data in 2.5D (Polyhedral model), and do not address, or suggest solutions for, dense built-up areas.

Most of these works have focused on approximate visibility computation, enabling fast results using interpolations of visibility values between points, calculating point visibility with the Line of Sight (LOS) method. Lately, fast and accurate visibility analysis computation in 3D environments.

A vast number of algorithms have been suggested for speeding up the process and reducing computation time. Franklin evaluates and approximates visibility for each cell in a DEM model based on greedy algorithms. Wang et al. introduced a Grid-based DEM method using viewshed horizon, saving computation time based on relations between surfaces and the line of sight (LOS method). Later on, an extended method for viewshed computation was presented, using reference planes rather than sightlines.

One of the most efficient methods for DEM visibility computation is based on shadow-casting routine. The routine cast shadowed volumes in the DEM, like a light bubble. Extensive research treated Digital Terrain Models (DTM) in open terrains, mainly Triangulated Irregular Network (TIN) and Regular Square Grid (RSG) structures. Visibility analysis in terrain was classified into point, line and region visibility, and several algorithms were introduced, based on horizon computation describing visibility boundary.

In the many uses of UAV (Unmanned Aerial Vehicle) a pilot uses real-time telemetry to take-off, fly and land the craft with continuous communication between ground station and the UAV on-board computer. Making these tasks autonomous, will allow UAVs to perform missions without continuous communication, and thus prevent hijack or damage by hackers, be more stealth for surveillance and have unlimited distance from ground station (bound to energy limitation).

Autonomous landing of a UAV is a problem on the focus of many studies [6][7][8] and landing on marine vessel makes this problem even more complex due to sea level motion that also occur when target platform is at stand-still.

The object of this research is to produce a safe landing mechanism for a quadcopter in 3D urban environment, in order to allow it to perform fully autonomous missions carried out at sea. Also, this mechanism could be used in pilot guided missions, as guideline suggestions to the pilot with how/when it is safe to land.

We assume the target position is known and Ground Station sets "home" position in the drone to be target's GPS position. Then the Bebop2 built-in "Return Home" function will bring it to the target, with up to a few meters off.

The proposed mechanism will perform the following tasks to achieve a "safe landing" decision: First, we need to visually search for and recognize the platform target and find the docking area. Once the target is found, the drone should set course and fly to target to be exactly above. Then, we detect and analyze the position of the landing surface and its plane angle relative to the camera. And finally, we will send the data to each of two implementations of the decision algorithms: 1. Using a supervised machine-learning classifier (pre-loaded with data), The machine input requires a quick pre-processing to set the data into a fixed structure vector, to resemble fitted data in the classifier. 2. Calculating directly from the data returned from the ArUco detection functions. The drone will then land safely on the boat, by sending a "land" command on time.

The problem of autonomous landing an UAV was on the focus of many studies as the survey review state-of-the art methods of vision-based autonomous landing, for a wide range of UAV classes from fixed-wing to multi-rotors and from large-scale aircrafts to miniatures. The main motivation for dealing with autonomous landing is the difficulty in performing a successful landing even with a pilot controlling the UAV. As it seems by statistics showed in [5], most of the accidents related to Remotely Piloted Aircraft Systems (RPAS) occur when the pilot tries to land the UAV.

Extensive research has been done on the subject to explore the various situations, technologies and methods to engage this problem. The work performed on previous studies, reviewed later in this section, is a great starting point for this project, as it is purely academic and relays on series of already existent technologies and tools, such as OpenCV [4], Sci-kit learn and the Parrot Ground SDK [2].

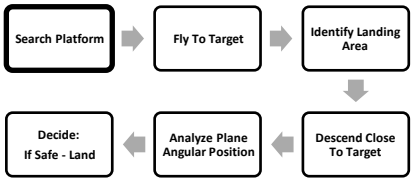


Figure 1. Proposed autonomous landing mechanism

In the following sections, we first introduce an overview of 3D models and extended the 3D visible volumes analysis. In the next section, we present the autonomous navigation process based on our fast visibility analysis with training data and classifier as can be seen in Figure 1. Later, we present the simulation based on our 3D visible volumes analysis.

II. AUTONOMOUS NAVIGATION PROCESS

The basic step starting this process related to obstacle avoidance and visible area described in the next sections. Following that, we divide the autonomous navigation mission into two separate problems. The first part deals with navigating UAV from an arbitrary position far from target, as far field. The second part is related to navigating to the target in the near field where the target is visible.

In the first scenario, which is when the mission objectives are reached and the drone needs to get to the target vessel for landing, we can use the built-in functionality of the drone to "Return Home" by setting it "Home" position to the target's known GPS position.

Bebop2 "Return Home" function works in a way that it will lift the drone to 20m above ground relative to take-off position, then fly directly to GPS position of "Home" and descend to 2m. Notice that if the drone is starting at height of more than 20m it will not descend to 20m, but rather keep its height until final descend near "Home".

The "Return Home" accuracy brings the drone to "Home" sometimes with offset of a few meters. This is good enough to get us to the second problem of navigation with visual distance to the target, until the drone will be directly above target and ready for landing.

Once the drone is at "Home" position, it will rotate and with each full rotation the tilt angle will increase to look further below, and if after rotating and tilting to the maximum of -90 degrees to the horizon, i.e., directly down, it will try again at higher altitude (1m up) to maybe see further away.

After getting a visual identification the drone will set course, keeping the target in the middle of the screen, and moving forward to it, tilting the camera during the movements until the landing pad is directly below. According to that, landing pad located in the middle of the image and camera tilt is maximum.

Then the drone will lower altitude to ~50cm while keeping the landing pad centered underneath, and in that height the data from the AR tags will be converted to a vector of predefined structure to feed a classifier trained to detect optimum landing angle/position. Once the classifier gives "Safe" signal – a "Land" command will issue to the drone to perform immediately.

III. FAST AND APPROXIMATED VISIBILITY ANALYSIS

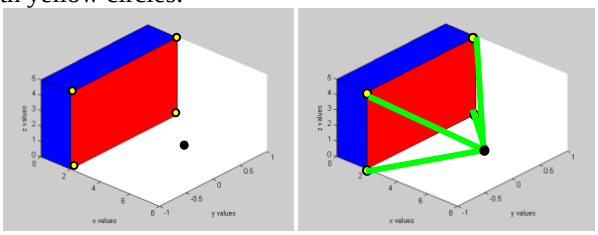
In this section, we present an analytic analysis of the visibility boundaries of planes, cylinders and spheres for the predicted scene presented in the previous sub-section, which leads to an approximated visibility.

A. Analytic 3D Visible Volumes Analysis

In this section, we present fast 3D visible volumes analysis in urban environments, based on an analytic solution which plays a major role in our proposed method of estimating the number of clusters. We present an efficient solution for visible volumes analysis in 3D.

We analyze each building, computing visible surfaces and defining visible pyramids using analytic computation for visibility boundaries. For each object we define Visible Boundary Points (VBP) and Visible Pyramid (VP).

A simple case demonstrating analytic solution from a visibility point to a building can be seen in Figure 2(a). The visibility point is marked in black, the visible parts colored in red, and the invisible parts colored in blue where VBP marked with yellow circles.



(a) (b)

Figure 2. (a) Visibility Volume Computed with the Analytic Solution. (b) Visible Pyramid from a Viewpoint (marked as a Black Dot) to VBP of a Specific Surface

In this section, we introduce our concept for visible volumes inside bounding volume by decreasing visible pyramids and projected pyramids to the bounding volume boundary. First, we define the relevant pyramids and volumes.

The Visible Pyramid (VP): we define $VP_i^{j=1..Nsurf}(x_0, y_0, z_0)$ of the object *i* as a 3D pyramid generated by connecting VBP of specific surface *j* to a viewpoint $V(x_0, y_0, z_0)$.

In the case of a box, the maximum number of N_{surf} for a single object is three. VP boundary, colored with green arrows, can be seen in Figure 2(b).

For each VP, we calculate Projected Visible Pyramid (PVP), projecting VBP to the boundaries of the bounding volume S.

Projected Visible Pyramid (PVP) - we define $PVP_i^{j..N_{surf}}(x_0, y_0, z_0)$ of the object *i* as 3D projected points to the bounding volume *S*, VBP of specific surface *j* trough viewpoint $V(x_0, y_0, z_0)$. VVP boundary, colored with purple arrows, can be seen in Figure 3.

The 3D Visible Volumes inside bounding volume *S*, VV_S, computed as the total bounding volume *S*, V_S, minus the Invisible Volumes IV_S. In a case of no overlap between buildings, IV_S is computed by decreasing the visible volume from the projected visible volume, $\sum_{i=1}^{N_{obj}} \sum_{j=1}^{N_{surf}} (V(PVP_i^j) - V(VP_i^j))$.

By decreasing the invisible volumes from the total bounding volume, only the visible volumes are computed, as seen in Figure 4. Volumes of VPV and VP can be simply computed based on a simple pyramid volume geometric formula.

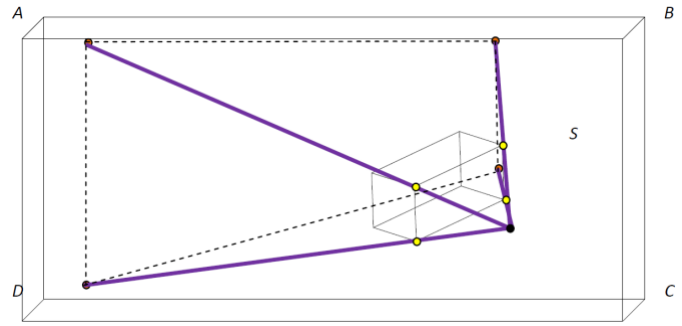


Figure 3. Invisible Projected Visible Pyramid Boundaries colored with purple arrows from a Viewpoint (marked as a Black Dot) to the boundary surface ABCD of Bounding Volume *S*

In a case of two buildings without overlapping, IV_S computed for each building, as presented above, as can be seen in Figure 5.

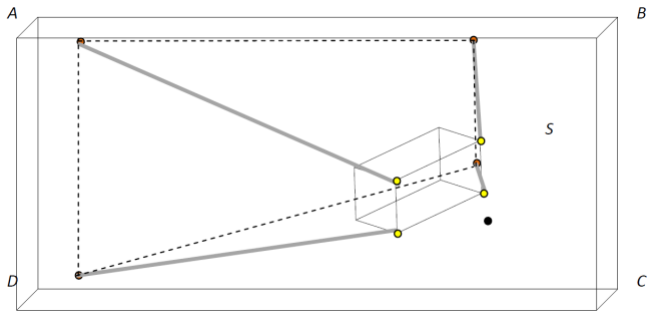


Figure 4. Invisible Volume $V(PVP_i^j) - V(VP_i^j)$ Colored in Gray Arrows. Decreasing Projected Visible Pyramid boundary surface ABCD of Bounding Volume S from Visible Pyramid

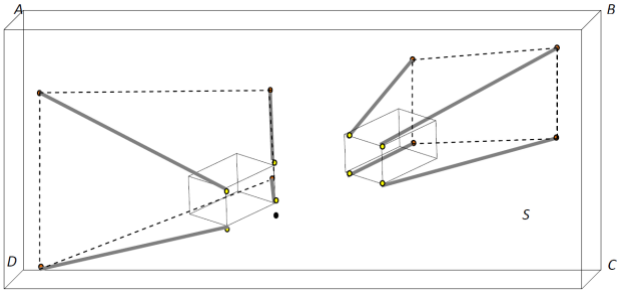


Figure 5. Invisible Volume $V(PVP_i^j) - V(VP_i^j)$ Colored in Gray Arrows. Decreasing Projected Visible Pyramid boundary surface ABCD of Bounding Volume S from Visible Pyramid

Considering two buildings with overlap between object's Visible Pyramids, as seen in Figure 6(a). In Figure 6(b), VP_1^1 boundary is colored by green lines, VP_2^1 boundary is colored by purple lines and the hidden and Invisible Surface between visible pyramids $IS_{VP_1^i}^{VP_2^i}$ is colored in white.

Invisible Hidden Volume (IHV) - We define Invisible Hidden Volume (*IHV*), as the *Invisible Surface (IS)* between visible pyramids projected to bounding box *S*.

For example, IHV in Figure 6(c) is the projection of the invisible surface between visible pyramids colored in white, projected to the boundary plane of bounding box *S*.

In the case of overlapping buildings, by computing invisible volumes IV_S, we decrease IHV twice between the overlapped objects, as can be seen in Figure 6(c), *IHV* boundary points denoted as $\{A_{11},..,A_{18}\}$. The same scene is presented in Figure 7, where Invisible Volume $V(PVP_i^j) - V(VP_i^j)$ is colored in purple and green arrows for each building.

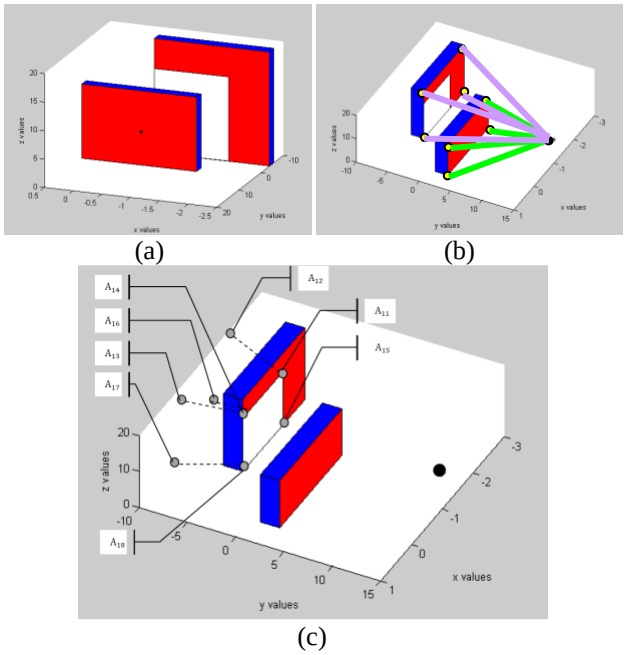


Figure 6. (a) Computing Hidden Surfaces between Buildings , VP_2^1 Base Plane, $IS_{VP_1^i}^{VP_2^i}$ (b) The Two Buildings - VP_1^1 in green and VP_2^1 in Purple (from the Viewpoint) and $IS_{VP_1^i}^{VP_2^i}$ in White (c) IHV boundary points colored with gray circles denoted

The *PVP* of the object close to the viewpoint is marked in black, colored with pink circles denoted as boundary set points $\{B_{11},..,B_{18}\}$ and the far object's *PVP* is colored with orange circles, denoted as boundary set points $\{C_{11},..,C_{18}\}$. It can be seen that *IHV* is included in each of these invisible volumes, where $\{A_{11},..,A_{18}\} \in \{B_{11},..,B_{18}\}$ and $\{A_{11},..,A_{18}\} \in \{C_{11},..,C_{18}\}$.

Therefore, we add *IHV* between each overlapping pair of objects to the total visible volume.

The same analysis holds true for multiple overlapping objects, adding the IHV between each two consecutive objects.

In Figure 8, we demonstrate the case of three buildings with overlapping. The invisible surfaces are bounded with dotted lines, while the projected visible surfaces to the overlapped building are colored in gray. In order to calculate the visible volumes from a viewpoint, *IHV* between each two buildings must be added as a visible volume, since it is already omitted at the previous step as an invisible volume.

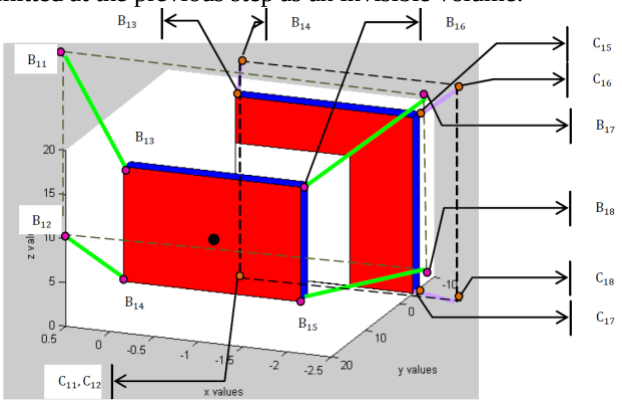


Figure 7. Invisible Volume $V(PVP_i^j) - V(VP_i^j)$ colored in purple and green arrows for each building. PVP of the object close to viewpoint colored in black, colored with pink circles and the far object PVP colored with orange circle

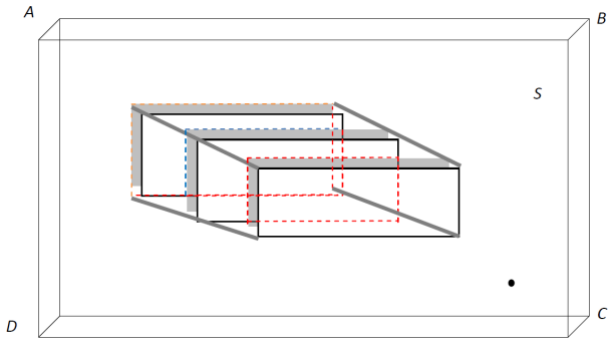


Figure 8. Three overlapping buildings. Invisible surfaces bounded with dotted lines, projected visible surfaces of the overlap building colored in gray

In this part, we extend the previous visibility analysis concept and include cylinders as continuous curves parameterization $C_{cln\,d}(x,y,z)$.

Cylinder parameterization can be described in (1):

$$C_{Cln\,d}(x,y,z) = \begin{pmatrix} r\sin(\theta) \\ r\cos(\theta) \\ c \end{pmatrix}_{r=const}, \quad \begin{matrix} 0 \le \theta \le 2\pi \\ c = c+1 \\ 0 \le c \le h_{peds\_\max} \end{matrix} \qquad (1)$$

We define the visibility problem in a 3D environment for more complex objects as:

$$C'(x,y)_{z_{const}} \times (C(x,y)_{z_{const}} - V(x_0,y_0,z_0)) = 0 \qquad (2)$$

where 3D model parameterization is $C(x,y)_{z=const}$, and the viewpoint is given as $V(x_0,y_0,z_0)$. Extending the 3D cubic parameterization, we also consider the case of the cylinder. Integrating (1) to (2) yields:

$$\begin{pmatrix} r\cos\theta \\ -r\sin\theta \\ 0 \end{pmatrix} \times \begin{pmatrix} r\sin\theta - V_x \\ r\cos\theta - V_y \\ c - V_z \end{pmatrix} = 0 \tag{3}$$

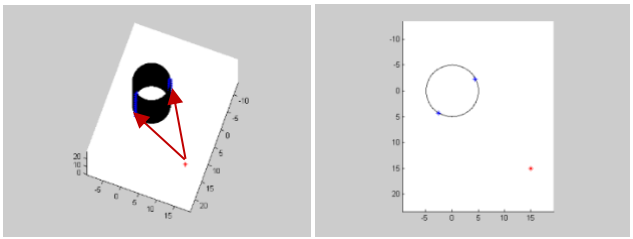
$$\theta = \arctan\left(-\frac{-r - \frac{\left(-vy\,r + \sqrt{vx^4 - vx^2\,r^2 + vy^2\,vx^2}\right) vy}{vx^2 + vy^2}}{vx}, -\frac{-vy\,r + \sqrt{vx^4 - vx^2\,r^2 + vy^2\,vx^2}}{vx^2 + vy^2} \right) \tag{4}$$

As can be noted, these equations are not related to Z axis, and the visibility boundary points are the same for each *x-y* cylinder profile, as seen in (3), (4).

The visibility statement leads to complex equation, which does not appear to be a simple computational task. This equation can be efficiently solved by finding where the equation changes its sign and crosses zero value; we used analytic solution to speed up computation time and to avoid numeric approximations. We generate two values of θ generating two silhouette points in a very short time computation. Based on an analytic solution to the cylinder case, a fast and exact analytic solution can be found for the visibility problem from a viewpoint.

We define the solution presented in (4) as x-y-z coordinates values for the cylinder case as Cylinder Boundary Points (CBP). CBP, defined in (5), are the set of visible silhouette points for a 3D cylinder, as presented in Figure 9:

$$CBP_{i=1..N_{PBP\_bound}=2}(x_0,y_0,z_0) = \begin{bmatrix} x_1, y_1, z_1 \\ x_{N_{PBP\_bound}}, y_{N_{PBP\_bound}}, z_{N_{PBP\_bound}} \end{bmatrix} \tag{5}$$



(a) (b)

Figure 9. Cylinder Boundary Points (CBP) using Analytic Solution marked as blue points, Viewpoint Marked in Red: (a) 3D View (Visible Boundaries Marked with Red Arrows); (b) Topside View.

In the same way, sphere parameterization can be described as formulated in (6):

$$C_{Sphere}(x,y,z) = \begin{pmatrix} r\sin\phi\cos\theta \\ r\sin\phi\sin\theta \\ r\cos\phi \end{pmatrix}_{r=const} \tag{6}$$
$$0 \le \phi < \pi$$
$$0 \le \theta < 2\pi$$

We define the visibility problem in a 3D environment for this object in (7):

$$C'(x,y,z) \times (C(x,y,z) - V(x_0,y_0,z_0)) = 0 \tag{7}$$

where the 3D model parameterization is $C(x,y,z)$, and the viewpoint is given as $V(x_0,y_0,z_0)$. Integrating (6) to (7) yields:

$$\theta = \arctan\left(\frac{r\sin(\phi)}{v\_y} - \frac{1}{v\_y\left(v\_y^2 + v\_x^2\right)}\left(v\_x\left(r\sin(\phi)\,v\_x - \sqrt{-v\_y^2\,r^2\sin(\phi)^2 + v\_y^4 + v\_x^2\,v\_y^2}\right)\right), \frac{r\sin(\phi)\,v\_x - \sqrt{-v\_y^2\,r^2\sin(\phi)^2 + v\_y^4 + v\_x^2\,v\_y^2}}{v\_y^2 + v\_x^2} \right) \tag{8}$$

Where *r* is defined from sphere parameter, and $V(x_0,y_0,z_0)$ are changes from visibility point along Z axis, as described in (8). The visibility boundary points for a sphere, together with the analytic solutions for planes and cylinders, allow us to compute fast and efficient visibility in a predicted scene from local point cloud data, which are updated in the next state.

This extended visibility analysis concept, integrated with a well-known predicted filter and extraction method, can be implemented in real time applications with point clouds data.

IV. VISIBILITY-BASED DRONE AUTONOMOUS LANDING

The landing pad designed as a plate with five markers – one in the center and four others on each corner:

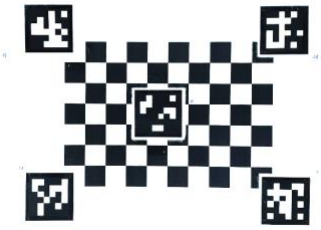


Figure 10. Landing pad with fiducial markers

Every ArUco marker has an ID as described in Figure 10, which can be determined when the marker gets detected, and

by that we can easily center the drone location above the landing pad even if only one or two markers are in view.

This landing pad has markers with ID values of {18,28, 17,25,4} selected randomly, but once selected they are very important to the implementation since the training data linked to the classifiers used as will be discussed later.

The proposed system takes each frame, and resolve all markers, then create a data vector of fixed length with all the necessary information of the markers.

Data format for each marker can be described as: [ID, rx, ry, rz, tx, ty, tz],

Where $\begin{bmatrix} r_x \\ r_y \\ r_z \end{bmatrix}$ is the rotation vector of a single marker and $\begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix}$ is the translation vector of that marker. This format repeats five times in each vector, where a tag ID has a fixed position for each tag. When a marker could not be found on a frame, the tag ID and all values of that marker will be set to zero.

Then send this vector to a classifier which will simply return strings telling us if the drone is centered above the landing pad or a correction movement is required. Possible answers are in the set: “CENTER”, “DOWNWARD”, “FORWARD”, “RIGHT”, “LEFT”.

For the Navigation we added more ArUCO tags surrounding this pad, in three sizes, so that they will be visible from varying distances along the navigation and descend process of the mechanism.

We used eight large tags, each surrounded by four medium tags and in between another five small tags as seen in Figure 11. The landing pad is printed on A4 page. And each of the eight patterns described here is also on an A4 page.

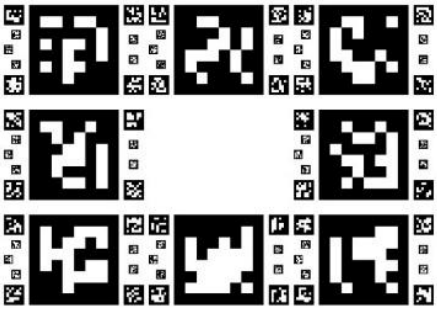


Figure 11. Navigation Assisting Tag Board Design

A. Training Data and Classifiers

In order to train the classifier, we used OpenGL as can be seen in Figure 12 to simulate the landing pad in a precisely controlled position and viewing angles. By that, we created a labeled data set, then use this precisely labeled data to fit in a variety of classifiers and test for accuracy. Following that, we tested several classifiers and selected the best performance for the purpose of the landing mechanism proposed.

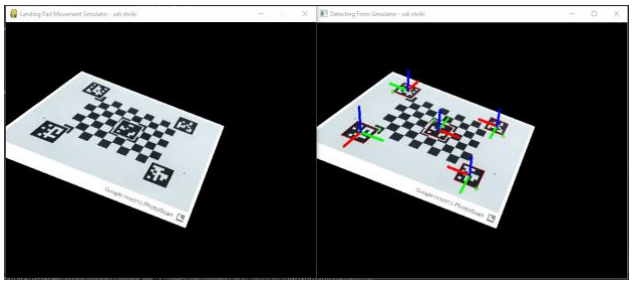


Figure 12. OpenGL Simulation for Training Data

The simulated platform rotating in roll, pitch and yaw - controlled by passing parameters, allowing me to tag every rendered frame as either safe for landing or not without visual computation (pre-label the data).

The simulator gets parameters from command line for setting some axis angle to run on a limited range, while rolling over all possible values of angles and positions, so the workload could be divided to parallel processes and even run on different machines.

After a few days running on several computers in parallel, the simulators generated a total of 15,193,091 vectors dataset, that could be used as training dataset for different models of classifiers.

Sci-Kit Learn package implements SVM with a fit function that takes labeled data as input in two variables: Y vector of y labels in a single column and X array of x vectors – each x vector is a line vector corresponds to the appropriate y in Y.

SVM does not allow incremental learning, i.e., it needs all data at once. This was quit an issue with the data size we tried to fit – fifteen million vectors. However, Sci-Kit Learn offers other types of classifiers, although all of them do not perform actual incremental learning (they do need all data at once), nonetheless, they do implement a partial fit function that can take each round a small portion of the data, and update the classifier’s support vectors.

For each classifier, we tried different parameters, and different sizes of the dataset by selecting randomly a fraction of the data. Then, test the model (using 25% of the data for test) to check it prediction accuracy.

TABLE I. CLASSIFIERS ACCURACY COMPARISSION

Classifier Type	**Best accuracy**
SGD, epsilon insensitive	57.341%
SGD, hinge	75.716%
SGD, huber	59.841%
SGD, log	73.658%
SGD, modified huber	73.362%
SGD, squared eps. insensitive	59.6%
SGD, squared hinge	73.857%
SGD, squared loss	57.171%
Perceptron	74.579%
Bernoulli NB	62.317%
Passive Aggressive Classifier	74.455%

The result in Table I shows that even the best classifier got only approximately 75% success in recall. This is insufficient for a safety mechanism even with filters added to the process of a final "safe" decision.

To further increase accuracy, we thought it would be more effective to use more than one classifier, in a voting manner, to decide together on the data. At first, we suggested a voting scheme that takes 10-15 of the best classifiers and check if more than 50% of them agree on a "safe" result, take that as the answer, we checked that over the data and results did not increase accuracy at all. Then we thought maybe a classifier of classifiers outputs could extract some new information in a smarter manner than a simple voting, and will help increase accuracy. We created a new dataset of the same size, only this time the vector consisted of zero for safe and one for unsafe result of a classifier over fifteen of the best classifiers (72%-75% accuracy) and trained this dataset on all types of classifiers with different parameters as before. This time, all classifiers listed above got around 76% accuracy, where the best classifier reached 76.8% accuracy. Approximately 2% improvement.

Finally, looking closely on live videos of the ArUco markers detections, we noticed that the axis drawn on the detected markers tend to shift rapidly usually around more "safe" angles, so we tried to manually correct the data, and remove some of the spiking data that is tagged as safe – i.e., the simulator created it as a safe angle, but detection errors made it as a vector that should rather be tagged as unsafe.

All data marked as safe, with "Z" axis angle in all detected markers, re-tag as unsafe, if a certain threshold is passed.

Before rectifying the dataset consisted of about 50% safe labels. This method reduced the number of "safe" tagged vector to about 20% of the data.

Fitting this new retagged dataset to all models as before, and testing again for accuracy, results improvements shown in details reported in Table III. The results improved drastically.

Best classifier selected for the mechanism is SGD (Stochastic Gradient Descend) with loss parameter set to logarithmic. This classifier showed 86% percent accuracy, which could be used with some filtering to suppress false alarm rate even more.

V. SPATIAL RAPID RANDOM TREES

In this section, the Rapid Random Trees (RRT) path planning technique is briefly introduced with spatial extension, which is the basic motion planning drone algorithm. RRT is dealing with high-dimensional spaces by taking into account dynamic and static obstacles including dynamic and non-holonomic robots' constraints.

The main idea is to explore a portion of the space using sampling points in space, by incrementally adding new randomly selected nodes to the current tree's nodes.

RRTs have an (implicit) Voronoi bias that steers them towards yet unexplored regions of the space. However, in case of kinodynamic systems, the imperfection of the underlying metric can compromise such behavior. Typically, the metric relies on the Euclidean distance between points, which does not necessarily reflect the true cost-to-go between states. Finding a good metric is known to be a difficult problem. Simple heuristics can be designed to improve the choice of the tree state to be expanded and to improve the input selection mechanism without redefining a specific metric.

A. RRT Stages

The RRT method is a randomized one, typically growing a tree search from the initial configuration to the goal, exploring the search space. These kinds of algorithms consist of three major steps:

1. **Node Selection:** An existing node on the tree is chosen as a location from which to extend a new branch. Selection of the existing node is based on probabilistic criteria such as metric distance.
2. **Node Expansion:** Local planning applied a generating feasible motion primitive from the current node to the next selected local goal node, which can be defined by a variety of characters.
3. **Evaluation:** The possible new branch is evaluated based on cost function criteria and feasible connectivity to existing branches.

These steps are iteratively repeated, commonly until the planner finds feasible trajectory from start to goal configurations, or other convergence criteria.

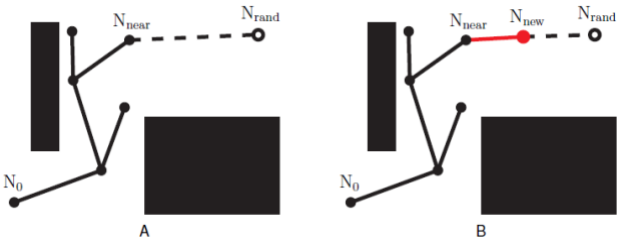


Figure 13. The RRT algorithm: (A) Sampling and node selection steps; (B) Expansion step.

A simple case demonstrating the RRT process is shown in Figure 13. The sampling step selects N_{rand}, and the node selection step chooses the closest node, N_{near}, as shown in Figure 13.A. The expansion step, creating a new branch to a new configuration, N_{new}, is shown in Figure 13.B. An example for growing RRT algorithm is shown in Figure 14.

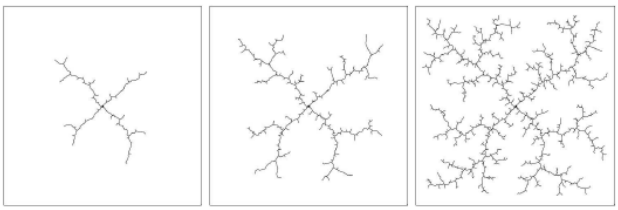


Figure 14. Example for growing RRT algorithm.

B. Spatial RRT Formulation

We formulate the RRT planner and revise the basic RRT planner for a 3D spatial analysis case for a continuous path from initial state x_{init} to goal state x_{goal}:

1. **State Space:** A topological space, *X*.
2. **Boundary Values:** $x_{init} \subset X$ and $x_{goal} \subset X$.
3. **Free Space**: A function $D: X \rightarrow \{true, false\}$ that determines whether $x(t) \subset X_{free}$ where X_{free} consist of the attainable states outside the obstacles in a 3D environment.
4. **Inputs:** A set, *U*, contains the complete set of attainable control efforts u_i, that can affect the state.
5. **Incremental Simulator:** Given a current state, $x(t)$, and input over time interval Δt , compute $x(t+\Delta t)$.
6. **3D Spatial Analysis:** A real value function, $f(x; u, OCP_i)$ which specifies the cost to the center of 3D visibility volumes cluster points (OCP) between a pair of points in X .

C. Spatial RRT Formulation

We present a revised RRT pseudo code described in Table II, for spatial case generating trajectory *T*, applying *K* steps from initial state x_{init}. The *f* function defines the dynamic model and kinematic constraints, $\dot{x} = f(x; u, OCPi)$, where *u* is the input and OCP_i set the next new state and the feasibility of following the next spatial visibility clustering point.

TABLE II. SPATIAL RRT PSEUDO CODE

```
Generate Spatial RRT (x_init; K; Δt)
T.init (x_init);
For k = 1 to K do
        x_rand ← random.state();
        x_near ← nearest.neighbor (x_rand; T );
        u ← select.input (x_rand; x_near);
        x_new ← new.state (x_near; u; Δt; f);
        T.add.vertex (x_new);
        T.add.edge (x_near; x_new; u);
End
   Return T
```

D. Search Method

Our search is guided by following spatial clustering points based on 3D visible volumes analysis in 3D urban environments, i.e., Optimal Control. The cost function for each next possible node (as the target node) consists of probability to closest *OCP*, P_{OCPi} , and probability to random point, P_{rand} .

In case of overlap between a selected node and obstacle in the environment, the selected node is discarded, and a new node is selected based on P_{OCPi} and P_{rand}.

E. STP Planner Pseudo-Code

We present our STP planner pseudo code described in Table III, for spatial case generating trajectory *T* with search space method presented above. The search space is based on P_{OCPi} and P_{rand}. We apply *K* steps from initial state x_{init}. The *f* function defines the dynamic model and kinematic constraints, $\dot{x} = f(x; u)$, where *u* is the input and *OCPi* are local target points between start to goal states.

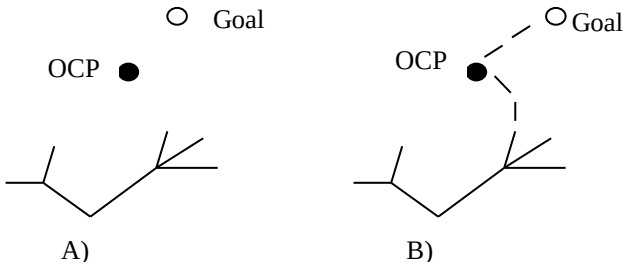


Figure 15. STP Search Method: (A) Start and Goal Points; (B) Explored Space to the Goal Through OCP

F. Completeness

Motion-planning and search algorithms commonly describe 'complete planner' as an algorithm that always provides a path planning from start to goal in bounded time. For random sampling algorithms, 'probabilistic complete planner' is defined as: if a solution exists, the planner will eventually find it by using random sampling. In the same manner, the deterministic sampling method (for example, grid-based search) defines completeness as resolution completeness.

Sampling-based planners, such as the STP planner, do not explicitly construct search space and the space's boundaries, but exploit tests with preventing collision with obstacles and, in our case, taking spatial considerations into account. Similarly, to other common RRT planners, which share similar properties with the STP planner, our planner can be classified as a probabilistic complete one.

VI. SIMULATIONS

The quadcopter we used in this research is a Parrot Bebop2 drone. It is a GPS drone with full HD 1080p wide-angle video camera with 3-axis digital stabilization, that can also take 14MB still pictures.

Bebop2 has GPS guided Return Home feature, strong 6" propellers, long rage communication (with WiFi extender or Skycontroller remote), which makes it suitable for a windy outdoors flight.

The Bebop2 drone uses seven different sensors simultaneously to keep it stable and produce an extremely stabilized video even when the drone makes tiny maneuvers to keep itself in place, the apparent view to the user looks like the drone is in fixed position as if it was hanging on a crane. Also, there are no moving parts when we pan/tilt the camera, it is done entirely by changing the relevant pane in the full fisheye image.

TABLE III. STP PLANNER PSEUDO CODE

```
STP Planner (x_init; x_Goal ;K; Δt; OCP)
T.init (x_init);
x_rand ← random.state();
x_near ← nearest.neighbor(x_rand; T );
u ← select.input(x_rand; x_near);
x_new ← new.state.OCP (OCP_1; u; Δt; f);
While x_new≠ x_Goal do
        x_rand ← random.state();
        x_near ← nearest.neighbor(x_rand; T );
        u ← select.input(x_rand; x_near);
        x_new ← new.state.OCP (OCP_i; u; Δt; f);
        T.add.vertex(x_new);
        T.add.edge(x_near; x_new; u);
end
return T;

Function new.state.OCP (OCP_i;u; Δt; f)
Set P_OCPi , Set P_rand
p←uniform_rand[0..1]
if   0 < p < P_OCPi
     return x_new = f(OCP_i,u,Δt);
else
    if P_OCPi < p < P_rand+ P_OCPi
then
     return RandomState();
end.
```

Parrot Ground SDK includes software development suite that provides a tool for developers to communicate and control with Parrot drones that uses AR.SDK3 framework, e.g., Mambo, Bebop, Disco, and Anafi. It also includes a simulator platform called Sphinx, built on Gazebo platform, with Parrot drones not just as models but with full featured firmware that are similar to the ones on the equivalent physical drones. This allows developers to fully test and debug their programs with real firmware feedback from a drone in mid-flight without the risk of injury or damages to equipment.

Ground SDK also provides a python wrapper called Olympe, to easily control drone objects. We preferred a third-party implementation named pyparrot, which is better documented and fully open-sourced, so it would be easier to add or change functionality to my needs.

A. ArUco Markers

The first problem we had to deal with, involves detection and identification of the landing pad. Afterword, we had to gather all planar information to pass to the decision mechanism for processing.

In order to simplify detection and get a fast and robust identification and planar information of the target, we used AR-tags on a specially designed landing pad.

Specifically, the use of off-the-shelf open source ArUCO seem to be a simple solution (other implementations of AR-tags, e.g., APRIL-TAGS may be suitable as well).

Implementation of ArUco marker detection exists in open-source library OpenCV, available for c/c++ and python. In order to get the marker real-world coordinates, we need the projection matrix of the camera and the distortion coefficients vector. To get these parameters a calibration is needed to be done once, then it could be loaded through a configuration file. The calibration process also available in OpenCV documentation, using a printed checkboard of known dimensions, and about twenty shots in different orientations and locations across the screen.

We incorporate different marker sizes to be able to detect markers in different distances from the target landing pad and follow the tags. ArUco Markers also have tag ID encoded in them so we even know which tag we are seeing and thus what size it is or where it is located on the board.

TABLE IV. IMPROVEMENTS IN ACCURACY OF CLASSIFIERS

Classifier type	Best accuracy	Before correction	Improvement
SGD, epsilon insensitive	83.450%	57.341%	26.11%
SGD, hinge	85.062%	75.716%	9.35%
SGD, huber	81.876%	59.841%	22.04%
SGD, log	**86.175%**	**73.658%**	**12.52%**
SGD, modified huber	86.131%	73.362%	12.77%
SGD, squared eps. Insensitive	82.019%	59.6%	22.42%
SGD, squared hinge	85.891%	73.857%	12.03%
SGD, squared loss	82.942%	57.171%	25.77%
Perceptron	85.470%	74.579%	10.89%
Bernoulli NB	81.664%	62.317%	19.35%
Passive Aggressive Classifier	84.041%	74.455%	9.59%

B. Implementation

To get control over a Bebop2 Drone, we found two python wrappers that we could use, and tested both of them. The first one comes with a Parrot Ground-SDK suite which includes the Sphinx Simulator, called Olympe. The Second wrapper pyparrot, originally developed for the Parrot Mambo, but now capable of controlling most of the newer generation Parrot drones.

We decided to work with pyparrot due to two main reasons: 1. Olympe used a closed virtual environment that made it harder to install additional packages using pip. 2. pyparrot is an open source, making it easy to adapt and change to my needs, it also suggests two types of video handling class: the first one uses FFMPEG and the other opens SDP file with VLC on a separate thread. Both methods were slow and missed critical frames especially in SEARCH mode, when the camera rotates to find the target. Sometimes the video smeared so badly we could barely recognize the landing pad even when we knew where it was there.

We changed the video handler to run on a separate thread (like the VLC option on pyparrot) only that in my

implementation we used standard OpenCV capturing module VideoCapture to open SDP file (contains IP, port, codec) for streaming coming from the drone or sphinx (depends on DRONE_IP parameter in the code), and another separate thread for the automation state machine that runs the different stages of this autonomous mission control and landing mechanism.

For proof of concept, all experiments were simulated in Gazebo based Sphinx simulator without moving wave simulations, or any automated changes in landing-pad angles or position. The changes were made manually by rotating the pad during simulation when the drone was waiting to get a safe signal from either classifier or calculations.

The experiment also did not simulate the use of "Return Home" functionality and assumed to start near target at about five meters in a random position.

The drone starts to search around to get a visual of the landing pad, then fly to set exactly above while looking directly down (-90 degrees below horizon).

Drone initiates with slow descend while keeping target in the middle of the frame, until reaches height of less than 50cm.

In this stage, decision mechanism under test should trigger "safe" when ArUco markers of the pad will be in a position that is regarded flat enough to be considered as safe.

In a preliminary experiment, we found that the classifier that we trained, could not get to a "safe" decision even when the landing pad was flat without any movements. Same classifier was tested with images from web-cam input seems to work fine, this could be issue caused by miscalibration of the camera. These inaccuracies cause ArUco functions that heavily relay on camera calibration, to produce different range of data relative to what the classifier was trained with (data from an OpenGL graphics drawn landing pad). This method should be further explored in future work.

Simplified manual calculation that work directly on data from ArUco functions output, could also be easily recalibrated and adjustable to fit with data ranges of mis-calibrated data. Finally, running full scenario of the experiment with landing pad on unsafe initial position got the drone flying above it and waiting, then manually flatten the landing pad, made the decision mechanism to trigger "safe" and send a landing command to the drone, which landed in the desired spot.

VII. Conclusion and Future Work

In this work we introduced a mechanism for autonomous landing a quadcopter in. The work focused to assist in the final stage of an autonomous mission, when drone returned to home, but still needs to find exact position of landing on the target and dealing with sea-level motion of the target.

In this study we developed a training simulator to create large data set of visual input, produced by OpenGL graphics in a controllable manner.

Also, we compared different types of trained classifiers to find best match to our particular data, and competed best classifier vs. direct observation and improvements as can be seen in Table IV.

For conclusion, the ArUco functions produce enough information regarding marker positions to be used manually and get a satisfying result for that manner. It is fast and robust and easily read to get a quick answer to whether it is safe or not, and the use of a classifier is not necessary.

References

[1] O. Gal and Y. Doytsher, "Autonomous Drone Landing in 3D Urban Environment Using Real-Time Visibility Analysis," GEOProcessing 2023, The Fifteenth International Conference on Advanced Geographic Information Systems, Applications, and Services, pp. 67- 72, 2023.

[2] Parrot Inc., "developer.parrot.com," 2020. [Online]. Available: https://developer.parrot.com/docs/olympe/userguide.html.

[3] A. McGovern, "pyparrot github repository," Jan. 2020. [Online] https://github.com/amymcgovern/pyparrot.

[4] OpenCV, Open Source Computer Vision Library, 2015.

[5] K. Williams, "A Summary of Unmanned Aircraft Accident/Incident Data: Human Factors Implications," The Federal Aviation Admistrator Oklahoma City, 2004.

[6] A. F. Cobo and F. C. Benıtez, "Approach for Autonomous Landing on Moving Platforms based on computer vision," The International Journal of Computer Vision, vol 4., 2016.

[7] L. Daewon, R. Tyler, and K. H. Jin, "Autonomous landing of a VTOL UAV on a moving platform using image-based visual servoing," IEEE International Conference on Robotics and Automation, pp. 971-976, 2012.

[8] T. Merz, S. Duranti, and G. Conte, "Autonomous Landing of an Unmanned Helicopter Based on Vision and Inertial Sensing," Experimental Robotics IX, pp. 343-352, 2006.

Building a Collaborative Platform for Evaluating and Analyzing Source Code Quality

Tugkan Tuglular
Department of Computer Engineering
Izmir Institute of Technology
Izmir, Turkiye
email: tugkantuglular@iyte.edu.tr

Onur Leblebici
Univera
Izmir, Turkiye
email: onur.leblebici@univera.com.tr

Emre Baran Karaca
Department of Computer Engineering
Izmir Institute of Technology
Izmir, Turkiye
email: emrekaraca@std.iyte.edu.tr

Naşit Uygun
Department of Computer Engineering
Izmir Institute of Technology
Izmir, Turkiye
email: nasituygun@std.iyte.edu.tr

Osman Anıl Hiçyılmaz
Department of Computer Engineering
Izmir Institute of Technology
Izmir, Turkiye
email: osmanhicyilmaz@std.iyte.edu.tr

Cem Sakızcı
Research Ecosystems
Izmir, Turkiye
email: sakizcicem@gmail.com

***Abstract* - The typical approach to data analysis is to store, query, and analyze data in a central location. In the case of source code, where multiple organizations or partners in a consortium contribute to a software, the repositories would be distributed and might be private. Within such a setting, one goal would be achieving and maintaining a certain level of source code quality across the consortium. One solution is to consider each partner as a node in a federated network. This paper proposes a federated code quality query and analysis platform. It further presents the features, the design, and the implementation of this platform.**

Keywords - source code quality; federated network; federated query; federated analysis.

I. INTRODUCTION

The proposed method in this paper improves the federated source code quality query and qnalysis platform presented in [1]. There are cases where each partner in a consortium, such as in the NESSI-SOFT project [2] in the Sixth Framework Program and in the MODUS project [3] in the Seventh Framework Program, does not want to share all of its source code but needs to be queried whether holding a pre-determined minimum source code quality level so that a certain level across the consortium is achieved and maintained. For such cases, one solution is to build a federated network so that each node in this network has its privacy, but shares required quality information. This paper considers this setting for source code quality and proposes a Federated Source Code Quality Query and Analysis (FSCQQA) platform. The setting is visualized in Figure 1.

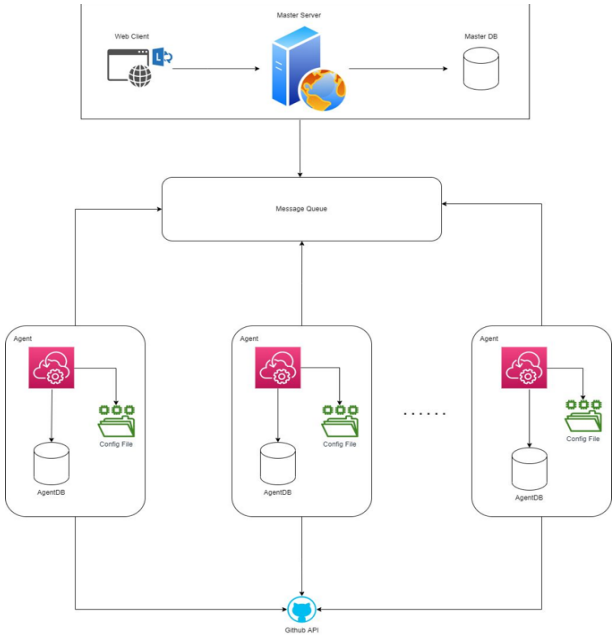


Figure 1. The FSCQQA platform overview.

The FSCQQA platform consists of a central site as seen at the top of Figure 1 and multiple sites, which are peers. It is a kind of peer-to-peer network, where the peers accept and

follow a general policy and corresponding rules. In addition, the central site is responsible for inclusion and removal of peer sites with respect to the general policy. Such platforms are on the rise especially in the health field, where privacy regulations and expectations are high, and accountability is enforced at state level. The proposed FSCQQA platform is one of the early attempts, where the idea is applied to source code, but not health records. Therefore, we believe that there is a practical gain from such a platform proposal.

The proposed platform is not only for consortiums to utilize. A global software company with development sites in various countries can also benefit from the FSCQQA platform. In this setting, concerns like revealing too much information about the software under development and the software development team may be relieved.

The FSCQQA platform offers opportunities for querying and monitoring source code quality across a consortium. This platform can facilitate analyzing how source code improvements are performed and how defect numbers are minimized. The FSCQQA platform has the following features:

- Analyze software quality with defect and source code metrics.
- Share defect and source code metrics with peers and consortium administration/management.
- Follow trends and improve.
- Compile federated historical data on defects and source code quality.

The features are kept at minimum in the paper, but they can be extended easily. To serve these features, the FSCQQA platform provides a data infrastructure, a software stack, and the operations on them. The proposed design is novel. The FSCQQA platform can be used for source code quality and defect prediction in the future.

As of today, there are multi-site software development companies whose sites are globally distributed. Each site is autonomous to some degree, but they are subject to central management rules. In such a setting, tracking each site's software quality and achieving an overall performance is not easy. Such a platform would be beneficial to them as well.

The paper is organized as follows: Section II presents the bug, or defect, datasets and source code quality metrics. Section III explains the proposed platform. Section IV outlines related work, and the last section concludes the paper.

II. FUNDAMENTALS

A. Bug Datasets

Lately, bug datasets are composed for bug or defect prediction. Following this, Ferenc et al. [4] compiled and standardized existing public bug datasets. The same group [5] extended their bug dataset and made the dataset publicly available at [6]. Several research works have produced and utilized bug datasets to develop and evaluate novel bug prediction methods. The objective of their study is to collect and combine current public source code metrics-based bug databases. In addition, they evaluated the abundance of gathered metrics and the bug prediction skills of the unified bug dataset. One research direction in this field moves toward combining bug datasets with software code quality metrics for better prediction. One of the first attempts is published by Osman et al. [7]. They evaluated sixty distinct bug prediction setting combinations on five open-source Java projects using a cost-aware evaluation scheme. Change measurements combined with source code metrics were discovered to be the most cost-effective option for developing a bug predictor. Another example of this work is presented by Mashhadi et al. [8]. They conducted a quantitative and qualitative study on two prominent datasets (Defects4J and Bugs.jar) utilizing 10 common source code metrics, as well as two popular static analysis tools (SpotBugs and Infer), for the purpose of evaluating their capacity to anticipate flaws and their severity.

B. Source Code Quality Metrics

Software quality metrics have been proposed for decades. The literature starts in 1970s. In the 1980s and 1990s, design metrics and their impact on software and source code were mainly studied. Henry and Selig [9] published a book on design metrics, which predicts source code quality. Two early research works specifically on source code quality metrics are by Pearse and Oman [10] and by Welker et al. [11]. They worked on the maintainability of source code.

With the popularity of object-orientation, the research in this area was intensified. Nuñez-Varela et al. [12] did a comprehensive mapping investigation on 226 articles that were published between 2010 and 2015 and discovered nearly 300 source code metrics. Even though object-oriented metrics have received a great deal of attention, there is a need for greater research on aspect and feature-oriented measurements. Prediction of software faults, complexity, and quality evaluation were recurring themes in these investigations.

Currently, there are separate tools as well as tools embedded into platforms, which not only produce source code quality metrics but also calculate technical debt. The next step for these tools seems to be towards predictions and suggestions for better code quality. Our vision and current attempt are in the same direction.

III. PROPOSED PLATFORM

We propose a federated code quality query and analysis platform, called FSCQQA. In this section, we first explain our design goals, such as "authentication and authorization" and "logging and monitoring" and continue with the services the FSCQQA platform provides. Some local services may vary between sites, but standardized procedures and rules will be implemented to ensure uniform administration and oversight. Finally, we present our user interface design to give a sense of use cases for the FSCQQA platform.

A. Design Goals

The major design goals are as follows:

Authentication & Authorization (AA): Each partner or site may have its own AA mechanism implemented. Then, each partner is responsible for the FSCQQA platform for its users'

queries. Each query includes the user and site identification; the site is responsible for logging the queries.

Access Control (AC) Policies: Each site may have its policies and regulations depending on the country where the site is. Therefore, the response to each query is filtered locally before sending. Each site should guarantee that any response does not contain any personal identifiable information.

Secure Communication: Each site must be able to communicate securely with trustworthy peers. All nodes exchange secure Public Key Infrastructure certificates in order to establish trust. While the FSCQQA platform is a federated network, the security of the nodes is only as strong as the network's weakest link.

Logging and Monitoring: Every query executed by a node should be recorded in an audit trail that the peer sites could view. The logs will be monitored by the central site for anomalies.

Standard APIs: Each site should provide standard APIs defined by the FSCQQA platform. Although the FSCQQA platform provides a software agent called FSCQQA agent to fulfil this requirement, the site may choose to implement its own software agent.

Source Code Repositories: The FSCQQA platform provides a software agent to work with GitHub [13] repositories. However, this is not a must. Any site can work with any source code repository but must ensure that standard APIs required by the platform are provided.

Management of Federated Platform: There is a central site responsible for the management of partners and their sites. These management operations include adding and removing partners and sites (a partner may have more than one site), constantly informing partners about other alive partners and sites, and collecting velocity and trend information from site.

B. Services

The FSCQQA platform defines two types of services, one provided by the FSCQQA agent and the other by the standard FSCQQA APIs. The FSCQQA agent is customizable through configurations with the following parameters:

- GitHub repository address
- GitHub repository access rights

The FSCQQA agent automatically generates local defect database for each site from a GitHub repository by extracting commit/issue histories and analyzing them. At the same time, it collects software metrics, such as lines of code and cyclomatic complexity, for each commit/issue. The defect information with software metrics will represent source code quality of the software developed at a site. Moreover, the FSCQQA agent extracts source code related metrics for a specific version using tools, such as OpenStaticAnalyzer [14]. The process is presented as a Unified Modeling Language (UML) sequence diagram in Figure 2. The FSCQQA agent is also responsible for the management of the local database for defects and metrics. To mitigate security concerns related to such an agent software, its source code should be open.

The standard FSCQQA APIs provide the services of the FSCQQA platform with respect to Open-API specifications [15]. The services are grouped as follows:

- Defect related metrics: number of existing (active) defects, defect density, defect resolve velocity, longest unresolved defect.
- Source code related metrics: class metrics, method metrics, coupling metrics, cohesion metrics, cyclomatic complexity metrics.

The services provide data for a specific version. They can be extended to supply data between two versions, but it may complicate the presentation of information and is, therefore, left as future work. The service calls can be for a specific metric or a set of metrics from a specific site or the whole network. If the whole network is queried, the query site requests all alive sites from the central site and queries each one individually then accumulates the results.

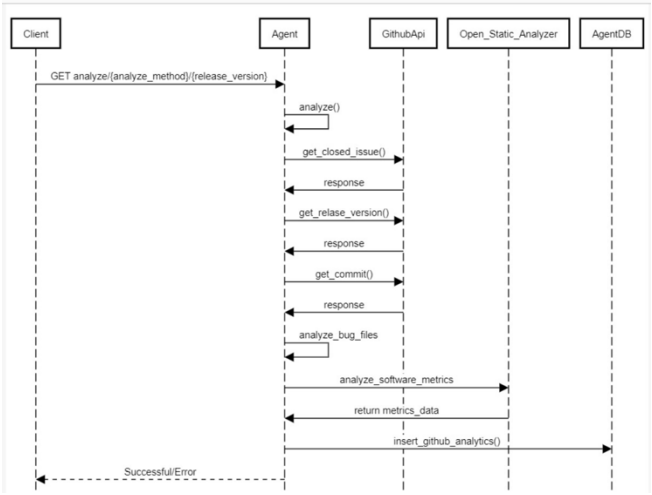


Figure 2. The FSCQQA platform operation.

The central site keeps a list of alive sites in the federated network by recording their heartbeats. Each site is expected to send a heartbeat every hour. If a site's heartbeat is missing necessary notifications are performed. The central site also holds summarized metrics for the whole network, such as overall defect resolve velocity and its trend over some time.

C. User Interface Design

The user interface design is presented via Figures 3-5. A user either in a site or in the central site can see the repositories with proper access rights, as shown in Figure 3. To mimic this operation, Figure 1 presents some public GitHub repositories. This project repository and selection window also indicates the status of the project with four states: "Not Analyzed", "Analyzing", "Analyzed", and "Failed". After selecting a project, a window like the one in Figure 4 is shown and if the status is neither "Analyzing" nor "Analyzed", the "Analyze" button appears. If it has already been analyzed, the details of the analyze operation are shown. To see the metrics, the metrics button should be pressed, and it takes the user to a window like the one shown in Figure 5. It is called the dashboard and presents various

metrics with charts and graphs. Metrics, charts, and graphs are all customizable.

IV. Prototype Implementation

In this section, we present our prototype implementation. The implementation is composed of the following components:

- Web Client: is the main interface for users to interact with the system.
- Master Application: manages a database with details of all agents such as IP and port. It efficiently routes requests from the web client to the appropriate agent and forwards the results back.
- Agent: specializes in executing detailed code analysis tasks, securing sensitive data, and then reporting findings back to the master. Each site is managed by an agent. In this protype implementation, we mimic each site with a GitHub repository.
- GitHub API: is an interface that allows us to access project source codes and general information, facilitating integration with various Github repositories.

The master application has a layered architecture with the following layers:

- Controller: processes various HTTP requests and routes the flow according to the request type.
- Service: performs business operations, coordination tasks, and interaction with the Controller and Repository layers.
- Repository: provides uninterrupted access to the database by facilitating basic data operations.

The master application provides the API endpoints shown in the Appendix. Each endpoint has a controller, and the service layer provides necessary operations with the help of Repository layer.

Each agent registers with the master application before starting any operations. When the user prompts the agent requests repository and project information from GitHub API. Then the agent stores the fetched repository and project information to the database through the master application.

The GitHub repository information contains the following elements:

- Watchers: Indicates the number of users monitoring the repository for changes.
- Topics: Tags or subjects associated with the repository.
- License: Details about the repository's licensing, including its type, URL, and some specific attributes.
- Visibility: Shows if the repository is public or private. In this case, it's public.
- Forks Count: The number of times this repository has been forked by other users.
- Stargazers Count: The number of users who have "starred" the repository, indicating their appreciation or interest.
- Default Branch: The primary branch of the repository, commonly where main development takes place.
- Homepage: The official homepage or documentation link for the repository.
- Number Of Contributor: The number of users who have contributed to the repository.

Figure 3 shows an example of a repository and project versions in that repository. The user can choose a version to be analyzed. As an example, it is 8.0.1.Final version of hibernate-validator in Figure 3. The prototype implementation uses OpenStaticAnalyzer v5.1.0. It provides source code analysis with 46 metrics, such as "Lines of Code", "Comment Lines of Code", "Lines of Duplicated Code", and "Total Number of Statements".

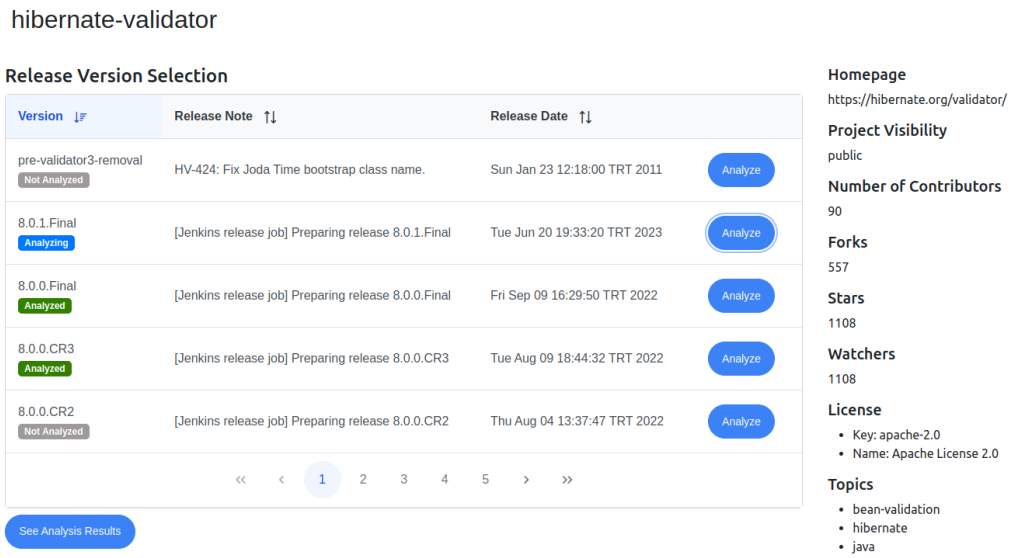


Figure 3. Project repository and release version selection user interface.

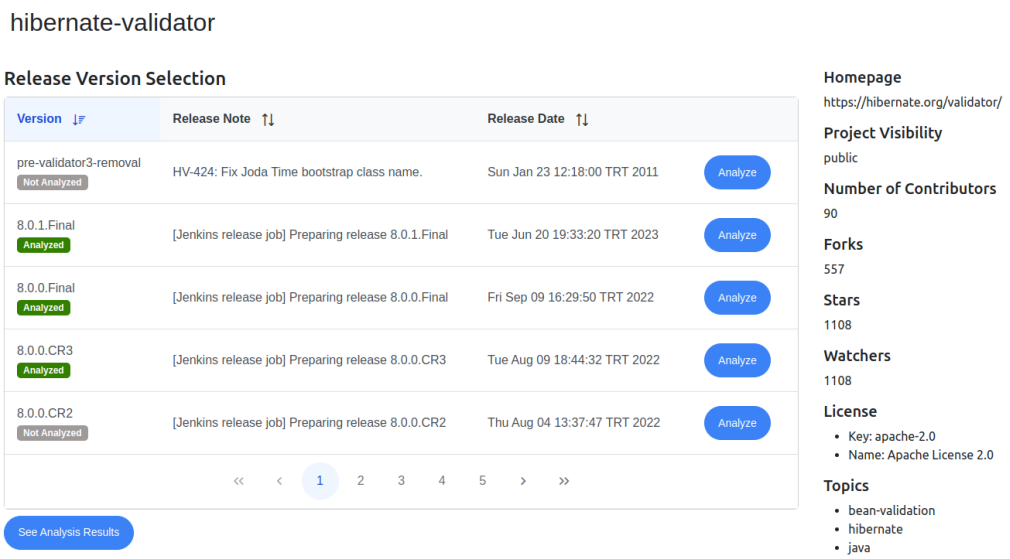


Figure 4. List of analyzed release versions user interface.

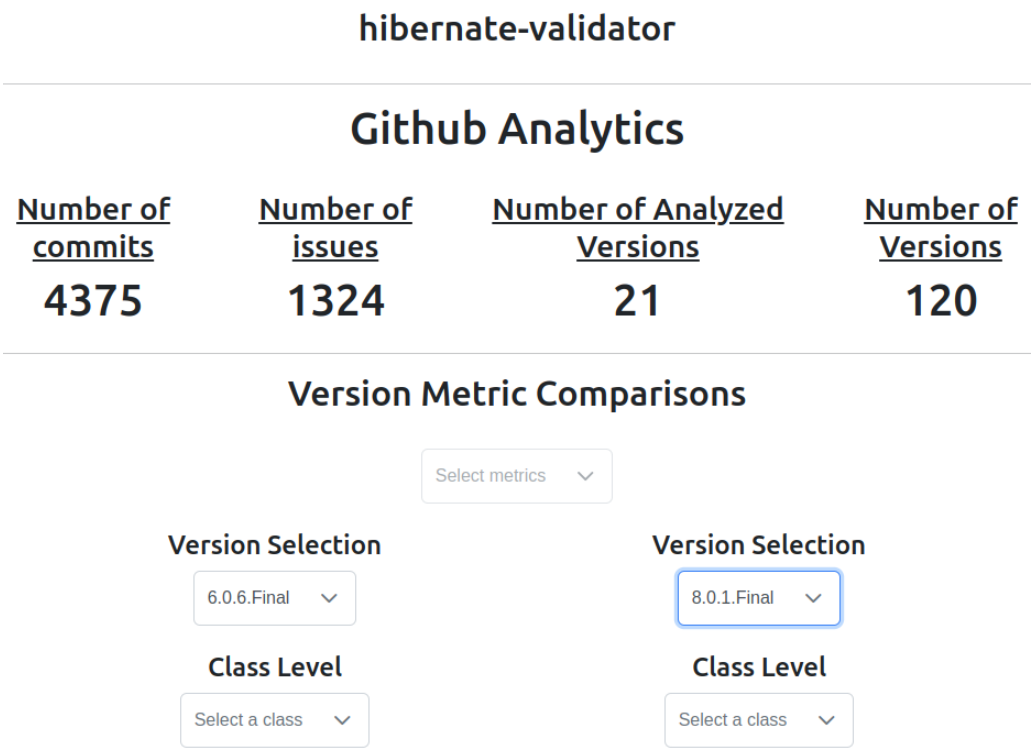


Figure 5. Version metric comparisons user interface.

Figure 4 shows an example list of analyzed release versions. At any time, the user can click the "See Analysis Results" button and the web clients shows a GUI such as in Figure 5. At the top, the following information gathered from GitHub is presented along with the number of analyzed versions:

- Number Of Commits
- Number Of Versions
- Number Of Issues

Analyzed versions becomes important if the user wants to compare versions. The metrics mentioned above are listed as shown in Figure 6. The user must choose minimum three metrics. Selection of metrics up to five is allowed. In addition to metrics selection, the user is asked to choose the classes from two different versions to be compared. As shown in Figure 6, the ReflectionHelper class is chosen as an example.

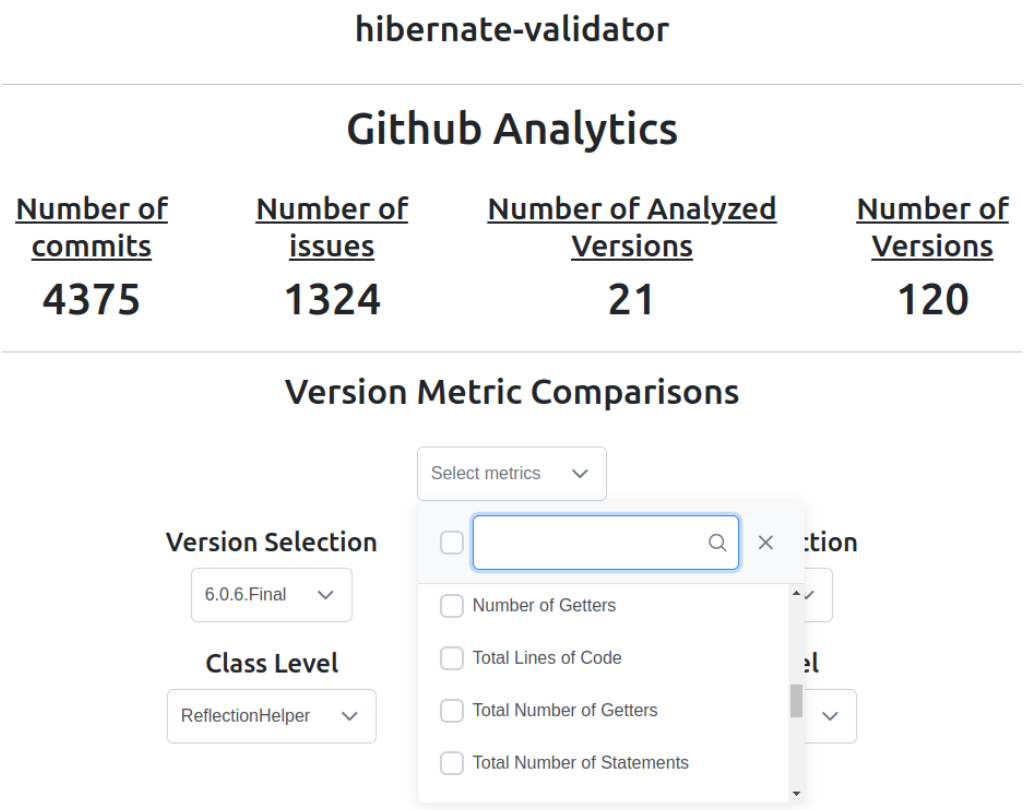


Figure 6. Version metric comparisons user interface with metric and class selection.

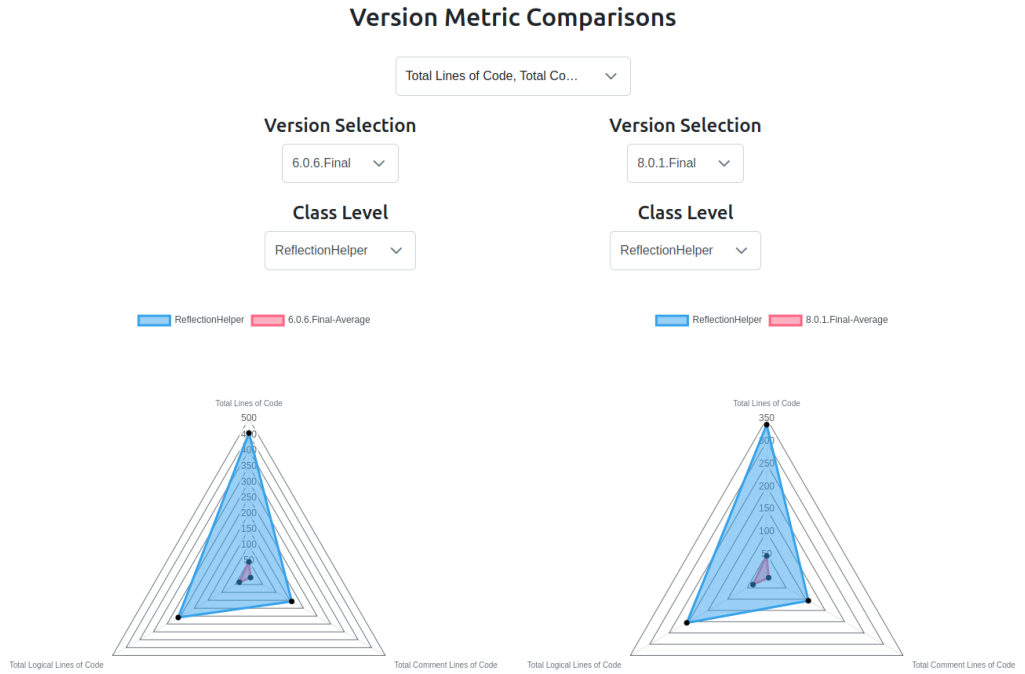


Figure 7. Version metric comparisons user interface with a three-dimensional spider chart.

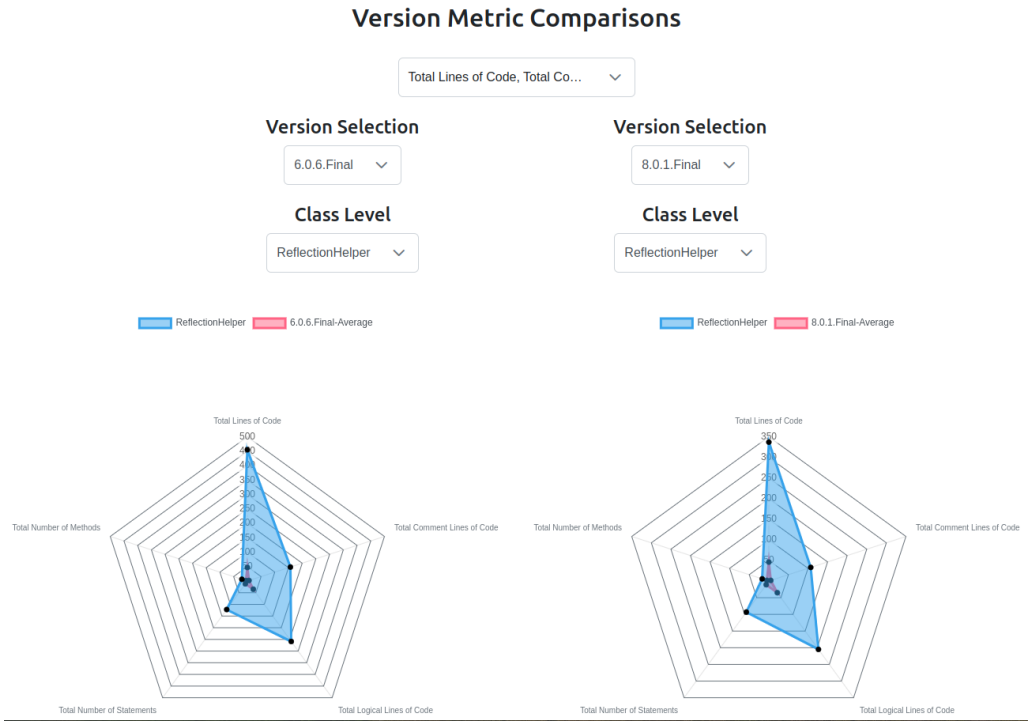


Figure 8. Version metric comparisons user interface with a five-dimensional spider chart.

Figure 7 shows version metric comparisons of the user interface with a three-dimensional spider chart. The dimensions are the chosen metrics. For this example, they are "Total Lines of Code," "Total Comment Lines of Code", and "Total Logical Lines of Code". The spider chart illustrates class metric values and average values for all classes in that version. This presentation technique enables users to compare values intra-version and inter-version. The spider chart is specifically chosen because of the patterns/shapes that appear. They are easier to compare.

The specific example in Figure 7 is indicative in terms of code quality. It is observed that the ReflectionHelper class is larger compared to other classes but throughout the versions it becomes smaller. The shape not being changed means the ratios between the metrics are preserved.

New metrics can be added to the spider chart without effort, as seen in Figure 8. For example, "Total Number of Statements" and "Total Number of Methods" are also chosen, which makes a five-dimensional spider chart. This five-dimensional spider chart makes the change in the ReflectionHelper class more apparent.

The prototype implementation does not contain all the features explained above. It is performed to show the feasibility of the proposed approach and design. This way we aimed to validate the main features of the Federated Source Code Quality Query and Analysis Platform.

V. Related Work

The concept of federated networks is not new, and they are not limited to a certain field. The services are called federated if their service architecture spans numerous independent control domains [16]. It is challenging to manage federated services and provide effective customer assistance since only a tiny portion of the environment can be monitored and controlled by any given authority. Bhoj et al. [16] characterized many facets of federated networks as early as 1997.

Some other examples of federated networks are as follows. For instance, Afsarmanesh et al. [17] proposed the PRODNET architecture for federated information management. Another example is Open Cirrus [18], which is proposed to federate a multitude of sites with diverse hardware, services, and tools for providing federated data centers for open source systems and services research. The sites reside on different continents and are subject to different privacy legislation and concerns.

The health domain is currently running federated networks. For instance, CanDIG [19] is a Canadian national health federated research data platform designed to assist the finding, querying, and analysis of permitted health research data across institutions and projects. CanDIG is the first Canadian federation of many human genomes and biomedical data projects. Another proposal for health domain is the Cross-Institutional Clinical Translational Research project [20], which investigated a federated query tool and examined how this tool can facilitate the discovery of clinical trial cohorts by controlling access to aggregate patient data housed in academic medical centers that are not linked.

VI. Conclusion

Each day, new features are added to software, and with each new feature, extra bugs may be introduced, and source quality may suffer. The scenario becomes more complicated if the software development is distributed with specific privacy and trade secret considerations. When addressing the challenges mentioned above, it is desired that the software quality be maintained above a particular threshold. Toward this goal, this paper proposes a federated source code quality query and analysis platform called FSCQQA.

With the proposed platform, sites are not required to disclose their codes with any other site while aiming for high source code quality and low defect ratio. At each site, local defect datasets will be generated and analyzed. The analysis results as defect metrics and the source code metrics obtained from the static analysis will be shared within the federated network and can be queried. Furthermore, trend analysis can be conducted at the central site and shared with consortium sites.

As future work, federated analytics and prediction using the local datasets are planned. The sites may push defect-related features to the central site for future machine learning. Such a defect database is valuable in terms of following the reliability of each site but also in improving defect-free development by providing in-depth analysis, such as root-cause analysis, and suggesting training and education. Then, the prediction model will generate predictions on sites. The prediction model will be updated and enhanced based on further coming data, meaning new source code. As developer data will not be exchanged, there will be no privacy concerns.

Acknowledgment

The authors would like to thank Univera Inc. for valuable guidance.

References

[1] T. Tuglular, O. Leblebici, E. B. Karaca, N. Uygun, and O. A. Hiçyılmaz, "A Federated Source Code Quality Query and Analysis Platform," presented at the SOFTENG 2023, The Ninth International Conference on Advances and Trends in Software Engineering, Venice, Italy, Apr. 2023, pp. 41–46.

[2] "Networked European Software and Services Initiative-support office team." https://cordis.europa.eu/project/id/034359 (accessed Mar. 19, 2023).

[3] "Methodology and supporting toolset advancing embedded systems quality." https://cordis.europa.eu/project/id/286583 (accessed Mar. 19, 2023).

[4] R. Ferenc, Z. Tóth, G. Ladányi, I. Siket, and T. Gyimóthy, "A public unified bug dataset for java," presented at the Proceedings of the 14th international conference on predictive models and data analytics in software engineering, 2018, pp. 12–21.

[5] R. Ferenc, Z. Tóth, G. Ladányi, I. Siket, and T. Gyimóthy, "A public unified bug dataset for java and its assessment regarding metrics and bug prediction," *Software Quality Journal*, vol. 28, pp. 1447–1506, 2020.

[6] "Unified Bug Dataset." http://www.inf.u-szeged.hu/~ferenc/papers/UnifiedBugDataSet/ (accessed Mar. 19, 2023).

[7] H. Osman, M. Ghafari, O. Nierstrasz, and M. Lungu, "An extensive analysis of efficient bug prediction configurations," presented at the Proceedings of the 13th international conference on predictive models and data analytics in software engineering, 2017, pp. 107–116.

[8] E. Mashhadi, S. Chowdhury, S. Modaberi, H. Hemmati, and G. Uddin, "An Empirical Study on Bug Severity Estimation Using Source Code Metrics and Static Analysis," *arXiv preprint arXiv:2206.12927*, 2022.

[9] S. M. Henry and C. L. Selig, *Design Metrics which Predict Source Code Quality*. Department of Computer Science, Virginia Polytechnic Institute and State University, 1987.

[10] T. Pearse and P. Oman, "Maintainability measurements on industrial source code maintenance activities," presented at the Proceedings of International Conference on Software Maintenance, IEEE, 1995, pp. 295–303.

[11] K. D. Welker, P. W. Oman, and G. G. Atkinson, "Development and application of an automated source code maintainability index," *Journal of Software Maintenance: Research and Practice*, vol. 9, no. 3, pp. 127–159, 1997.

[12] A. S. Nuñez-Varela, H. G. Pérez-Gonzalez, F. E. Martínez-Perez, and C. Soubervielle-Montalvo, "Source code metrics: A systematic mapping study," *Journal of Systems and Software*, vol. 128, pp. 164–197, 2017.

[13] "GitHub." https://github.com/ (accessed Mar. 19, 2023).

[14] Department of Software Engineering, University of Szeged, Hungary, "OpenStaticAnalyzer." https://openstaticanalyzer.github.io/ (accessed Mar. 19, 2023).

[15] "OPENAPI Initiative." https://www.openapis.org/ (accessed Mar. 19, 2023).

[16] P. Bhoj, D. Caswell, S. Chutani, G. Gopal, and M. Kosarchyn, "Management of new federated services," presented at the Integrated Network Management V: Integrated management in a virtual world Proceedings of the Fifth IFIP/IEEE International Symposium on Integrated Network Management San Diego, California, USA, May 12–16, 1997, Springer, 1997, pp. 327–340.

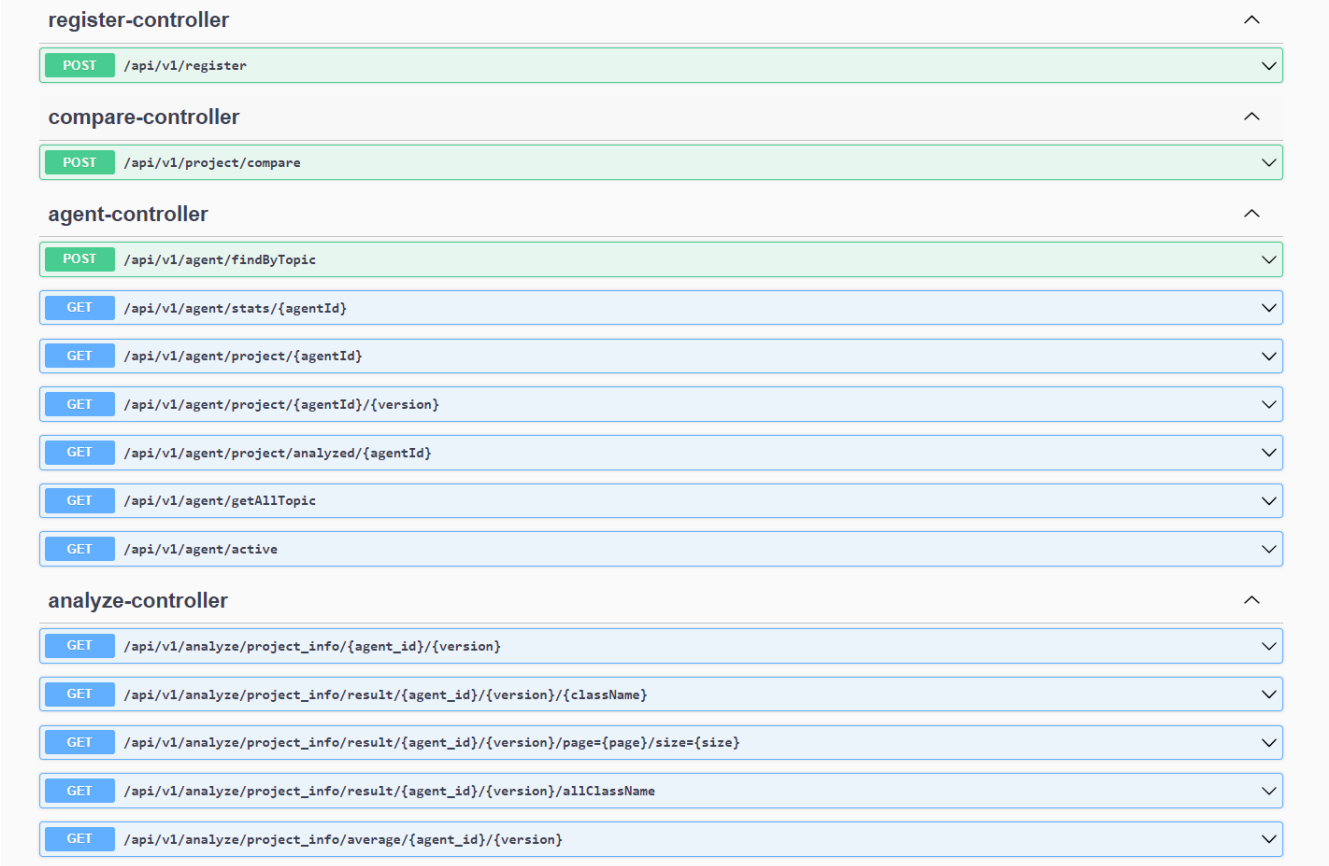
[17] H. Afsarmanesh, C. Garita, Y. Ugur, A. Frenkel, and L. O. Hertzberger, "Design of the federated information management architecture for PRODNET," presented at the Infrastructures for Virtual Enterprises: Networking Industrial Enterprises IFIP TC5 WG5. 3/PRODNET Working Conference on Infrastructures for Virtual Enterprises (PRO-VE'99) October 27–28, 1999, Porto, Portugal 1, Springer, 1999, pp. 127–146.

[18] R. H. Campbell *et al.*, "Open Cirrus™ Cloud Computing Testbed: Federated Data Centers for Open Source Systems and Services Research.," *HotCloud*, vol. 9, pp. 1–1, 2009.

[19] L. J. Dursi *et al.*, "CanDIG: Federated network across Canada for multi-omic and health data discovery and analysis," *Cell Genomics*, vol. 1, no. 2, p. 100033, 2021.

[20] N. Anderson *et al.*, "Implementation of a deidentified federated data network for population-based cohort discovery," *Journal of the American Medical Informatics Association*, vol. 19, no. e1, pp. e60–e67, 2012.

APPENDIX



Comparative Analysis of Small Data Acquisition Strategies in Machine Learning Regression Tasks Addressing Potential Uncertainties

Xukuan Xu
Aschaffenburg University of Applied Sciences
Aschaffenburg, Germany
e-mail: xukuan.xu@th-ab.de

Felix Conrad
Dresden University of Technology
Dresden, Germany
e-mail: felix.conrad@tu-dresden.de

Xingyu Xing
Aschaffenburg University of Applied Sciences
Aschaffenburg, Germany
e-mail: xingyuxing0630@gmail.com

Oskar Loeprecht
Dresden University of Technology
Dresden, Germany
e-mail: oskar.loeprecht@yahoo.de

Michael Moeckel
Aschaffenburg University of Applied Sciences
Aschaffenburg, Germany
e-mail: michael.moeckel@th-ab.de

***Abstract*—As the algorithms mature, the bottleneck in applying Machine Learning (ML) to engineering, in particular to process analysis, monitoring and control, is often caused by the limited availability of suitable data and the cost of data acquisition. For many ML projects, datasets have been collected independently of subsequent analysis. In laboratory-based development, data acquisition and coverage of possible process uncertainties pose challenges to the preparation of datasets suitable for ML. This paper benchmarks existing design of experiments (DOE) strategies based on data generated by a simulation model, discussing their aptitude for training accurate ML regression models. 11 representative sampling strategies have been investigated to provide guidance for data collection under data acquisition constraints, including consideration of possible measurement uncertainties. As the optimal DOE depends on available data volume and the uncertainty level, recommendations for DOE selection are given.**

Keywords-Small-data; Process uncertainty; Design Of Experiments; Machine learning; Model-based sampling; Auto-sklearn.

I. INTRODUCTION

ML makes it possible to efficiently mine valuable information from data due to its powerful data analysis capabilities. With the prosperous advancement of algorithm research, model building is no longer a challenge limiting ML applications [1][2]. In fact, according to a survey from Crowdflower in 2016 [3], the efforts of data scientists are mainly (60%) consumed by data organizing and data cleaning. After this, 19% of the time is spent collecting datasets. This shows that data preparation involves considerable effort of ML applications in the current stage. However, this difficulty is often overlooked by the informatics community. In most cases, the datasets are pre-existing. With this standpoint, they simply optimate the algorithm at the software side for data analysis. However, the dataset's quality determines the upper limit of data analysis. Therefore, in some cases, it may be unfeasible to look at a solution only from the ML model side. Only recently, the intersection of experimental design towards data collection and ML has come to the fore. R. Arboretti et al. systematically reviewed the joint application of DOE and ML in areas such as industrial production, which identified the current status of research in terms of DOE selection for ML [4]. In this context, a preliminary study of the relationship between DOE selection and ML was conducted based on simulation models [5]. Roberto Fontana et al. benchmarked the performance of ML models obtained from data collected with different DOE strategies, where the potential of an Active Learning (AL) approach for dataset acquisition was investigated [6]. However, their experiments were limited to a specific amount of data without further guidance of DOE selection for varying data volumes.

It is both a challenge and an advantage to look at data preparation from the perspective of a production engineer. Collecting a single element of the dataset requires that a product is physically produced and the relevant data is measured during the manufacturing process. In practice, an extra number of products is required to account for deficient outcomes. This limits the amount of usable data for ML analysis. The cost considerations often constrain the overall amount of data. However, pre-existing knowledge, experience or even intuition of the process often allows an engineer to focus the data generation on particularly relevant subsets of an overly complex parameter space.

Purpose-built datasets for ML modeling may address two possible directions [7]:

I. Finding the control variables and their optimal values that result to an optimal response
II. Exploring the neighborhood around the optimal values to generate knowledge for monitoring, anomaly detection and control

This article investigates the latter under the constraint of limited resources (e.g., time, budget) for data acquisition and fixed overall statistical process uncertainty. Based on the data obtained from an experimental lithium-ion battery (LIB) production line realized within the KIproBatt project [8], we describe the practical difficulties in preparing datasets for ind-

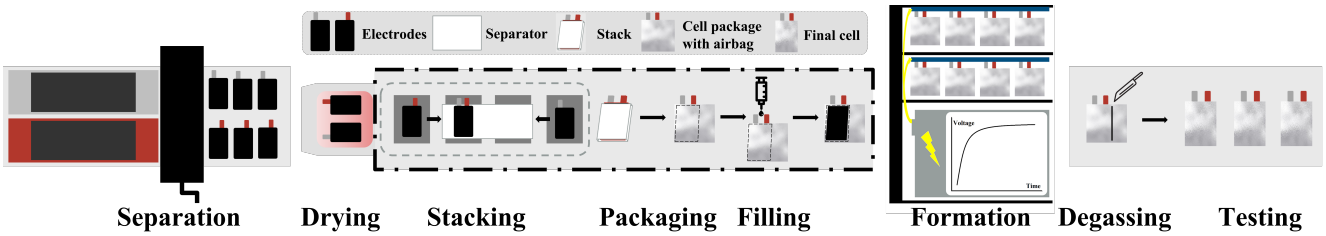


Figure 1. LIB cell assembly process from separation to EOL-tests

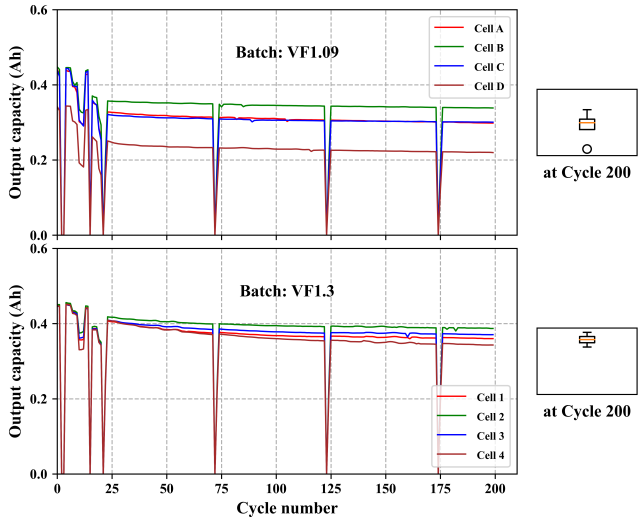


Figure 2. Cell output capacity related to cycle number in a cycling test

ustrial process development in Section II. In Section III, existing DOE approaches are described. The ML package Auto-sklearn [9] and its modelling capabilities are introduced. Finally, the experimental setups for comparing the potential of different DOEs with predefined data resources are presented. The experiments include both cases with and without process uncertainties. The results of the experiments and the discussions are documented in Section IV. Guidelines for DOE selection based on experimental data are given at its end. Considerations on the application of an iterative DOE in case of small data are given with Emukit DOE as an example. Section V summarizes the contributions of this article and possible further expansions of this research are envisaged.

II. DESCRIPTION OF SMALL-DATA CONTEXT

A. Small data problem

Small-batch production is often unavoidable in laboratory research, on a pilot production stage prior to upscaling [10], or in customer-specific (individualized) manufacturing [11]. Often, data acquisition is limited by budget or time constraints to datasets with less than one thousand elements. The particular choice of selected data points affects the outcomes of subsequent analysis. For illustration, we consider the project KIproBatt as an example of a typical small-scale data generation: a total of ca. 500 Li-ion battery cells is to be produced with a semi-automatic production line in a laboratory environment. Research questions include the impact of process deviations on the quality of final cells as well as the exploration of complex correlations among process parameters. Note that one cannot define the "small-data problem" by sole reference to a fixed amount of data. Instead, the characteristics and complexity of both the research objectives and the applied ML methods have to be considered.

B. Lack of process knowledge & complexity of the production process

The number of required data depends on the complexity of the process. A large number of features, non-linear relationships and interactions between features increase the complexity of the process and thus the number of data points required. These conditions are often found in industrial production processes [11]. The assembly process of a LIB pouch cell is an example of such a complex process and is depicted in Fig. 1: cell assembly starts with electrode separation. Then, the anodes and cathodes are dried and fed into a glove box with a controlled atmosphere. Next, a stacking machine assembles the electrodes with a separator into cell stacks (Z-fold stacking). After the packaging, sealing and electrolyte filling, the cell is activated by the first charge and discharge (formation). The gas generated in this procedure is removed and the cell is finally sealed.

The complexity of this multi-step process leads to manifold variable interdependencies. Hence, an effective analysis should be based on a ML approach. However, it is challenged by limited data, which may lead to under sampling of the parameter space and a lack of convergence of the ML models. We define this as the fundamental characteristic of small-data context.

C. Process uncertainty

Complex processes are normally investigated for a limited set of process parameters only. While the remaining parameters are, in theory, assumed to remain constant, their unavoidable fluctuations contribute to statistical uncertainty in all measured data. Other sources for uncertainties lie, for instance, in the measurement uncertainties of the used sensors. This uncertainty is manifested in the data as identical input parameters will lead to a statistical spreading in the target responses.

In the KIproBatt project, using the injected electrolyte volume as the only tunable factor with two levels, we produced four cells at each level while ensuring that the rest of the process parameters were unchanged. Each cell was then tested according to the same cycling protocol to evaluate its performance. The cycling protocol also includes non-cycling tests such as pulse tests, c-rate test and quick charge test. Pulse tests are designed to obtain information regarding battery resi-

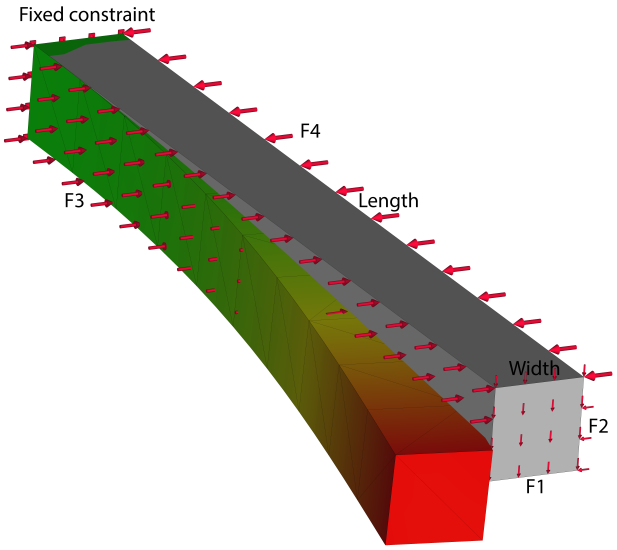


Figure 3. Constraints and forces in the FEM model.

TABLE I. INPUT PARAMETERS IN THE FEM SIMULATION

	Variables	Range	Unit
Input	Ym	50,000 – 600,000	Mpa
	Pr	0.1 – 0.45	1
	L	8000 – 10,000	mm
	W	1000 – 2000	mm
	F1	1000 – 10,000	kN
	F2	1000 – 5000	kN
	F3	1000 – 10,000	kN
	F4	1000 - 5000	kN
Output	Displacement	Ca. 0.7 – 400	mm

Stance, which are labelled as 0 during data processing. The results, using output capacity (OC) as an indicator, are shown in Fig. 2.

It can be seen that the performance of the battery cells within each batch varies. As the box plot illustrates, the process uncertainty is so evident in batch VF1.09 that cell D can be judged as an outlier (box plot).

The reasons for this might be processing errors due to human operations, a lack of process understanding that leaves some potential variables uncontrolled, or measurement errors in the hardware. But in the end, what emerges is the uncertainty of the OC.

When the process uncertainty exceeds the variation imposed on control variables, no direct conclusion can be derived. Normally, uncertainty reduction could be achieved either by optimizing hardware or by repeated measurement and averaging. However, for fixed measurement capacity, the latter implies a reduced ability for parameter space exploration. Therefore, DOE strategies can be developed further to find new compromises between resource allocation for uncertainty reduction and for parameter space sampling.

III. SETTING OF THE EXPERIMENTS

In this section, the potential of various sampling methods to build a regression model under different data volumes and levels of uncertainty are investigated. The analysis is divided into two parts:

1. The first part is to understand the performance of different DOEs through training an optimal regression model as the data volume varies.
2. The second analysis is to investigate the potential of these DOEs under varying levels of uncertainty, where different uncertainties are introduced to the target parameter.

An independent test dataset is obtained using Latin Hypercube Sampling (LHS), which consists of 2,500 data points. The root mean square error (RMSE) of the predicted displacement versus the output from the Finite Element Method (FEM) simulation is used to measure the true error of the ML model. The R^2 Score is also employed to evaluate the model [12]. The best achievable performance with the given training dataset of these models on the test dataset is considered as the potential of the corresponding DOEs.

FreeCAD was chosen as the platform for building simulation. It supports building models with python code and provides an application programming interface to facilitate the import and export of data. The simulation model includes eight input parameters: Young's modulus (Ym), Poisson's ratio (Pr), length (L) and width (W) of the beam with four force constraints applied to the beam. The displacement magnitude of the beam is defined as the target parameter. Table I and Fig. 3 provide further information about this simulation model.

Twelve algorithms covered by Auto-sklearn are used to build the regression models for the prediction of the target parameter in the parameter space [13]. In order to provide an objective comparison among the DOEs without potential deviations during the training process, the settings of the hyperparameters in Auto-sklearn should be tuned to appropriate values [14]. Thus, it can be ensured that the potential of DOEs are effectively compared without the influence of non-optimal model training.

A. *Tested DOE strategies*

DOE is an established approach to systematically collect information about a system or process. It aims at delivering the most relevant experimental data for addressing a given research objective. The origin of classical DOE can be traced back to the Analysis of Variance (ANOVA) proposed by FISHER in the 1920s [15]. Conventional DOE has a set of proven paradigms: screening design, e.g., full factorial design (FFD) for identifying relevant parameters; response surface design, including central composite design (CCD), Box–Behnken design (BBD), for detailed investigation of optimal parameter configurations [16]. With the development of data science and easier access to data, ML tools have been

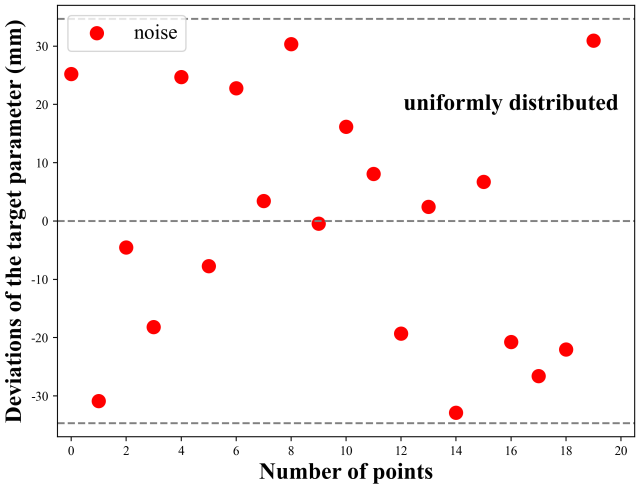


Figure 4. Deviations of displacements from simulated values due to a 10% uncertainty (up to 10% of the maximal change of the displacement in the predefined input dataspace)

successfully applied to many data analysis problems. ML has unparalleled efficiency advantages in analyzing big data (compared to the volume of data in conventional DOEs) with complex interdependencies. However, little attention has been paid to the interplay of data set generation and ML-based data analysis. Represented by LHS, space-filling design is able to partition the data space isometrically into multiple levels [17][18]. This feature makes LHS well-suited to drive ML schemes. A series of studies have conducted the generation of datasets for ML based on conventional DOEs in the past five years [19][20]. In addition, motivated by some ML algorithm developments, iterative data acquisition schemes have been discussed.

Emukit provides such a model-based iterative DOE scheme within a Bayesian optimization framework [21][22]. The Emukit DOE tool starts from a set of given initial data points and iterates the following three steps to generate sample points in a given input space:

- fit a prediction model to the existing data
- find the next point with the highest marginal predictive variance as predicted by the prediction model
- add this new data point to the existing dataset

Such iteration allows for the most efficient allocation of a limited number of data points based on certain metrics, such as marginal predictive variance of the model. This model-based scheme works well with ML data analysis since a prediction model (e.g., Gaussian process model, GP model) is used to predict the target response and calculate the variance during each iteration of data acquisition [22].

Table II contains a summary of the different DOEs which have been tested. Different settings for the CCD, criteria in the LHS and different acquisition functions in Emukit were considered as different DOEs. The range of the training data volume is set from 40 to 320. Since conventional DOEs (FFD, BBD, CCD) are predetermined by the number of input factors, levels and the DOE strategies, it is not possible to change the data volume continuously to build multiple datasets with a specified amount of data. As an example, given 8 variables, the dataset generated according to FFD must consist of 2^8 data points. The adopted solution was to use the D-optimal criterion [23] to filter the required optimal design. For example, the use of the D-optimal criterion enables the construction of any subsets with less than 256 data points, which makes it possible to continuously change the amount of data within a certain range.

TABLE II. DOEs AND THEIR ABBREVIATION CODES

Abbreviation	Sub	Descriptions
FFD		Full-Factorial design
CCD	CCD_c	Central-Composite design, where the star points are at the same distance from the center
CCD	CCD_i	A scaled down CCD_c design with each factor level of the CCD_c design divided by a given constant
CCD	CCD_f	Star points are at the center of each face of the factorial space
BBD		Box-Behnken design
LHS	LHS_c	Latin-Hypercube sampling, which centers the points within the intervals
LHS	LHS_m	Maximize the minimum distance between points, randomly distribute points within the intervals
LHS	LHS_cm	Maximize the minimum distance between points, centered them within the intervals
LHS	LHS_cor	Minimize the maximum correlation coefficient
Emukit	Emukit_us	Iterative sampling strategy, choose the next point according to the marginal predictive variance of a GP model [23]
Emukit	Emukit_ivr	Choose the next point such that the total variance of the model is reduced maximally [25]

B. *ML modeling*

The model training using Auto-sklearn is repeated five times. The best performance among them, i.e., the performance of the best model that can be obtained for this training dataset, will be recognized as the potential of the corresponding DOE used for collecting the training dataset. The experiments were conducted on a Dell workstation (Intel® Xeon® W-2295 Processor: 3.00 GHz * 36, memory: 128GiB). The settings of hyperparameters in Auto-sklearn used for modeling are shown in Table III.

TABLE III. HYPERPARAMETERS IN AUTO-SKLEARN

Hyperparameters in Auto-sklearn	Value
time_left_for_this_task	300s
per_run_time_limit	30s
initial_configurations_metalearning	25
memory_limit	20480 MB
resampling_strategy	"cross validation"
resampling_strategy_arguments	"folds: 5"
n_jobs	18

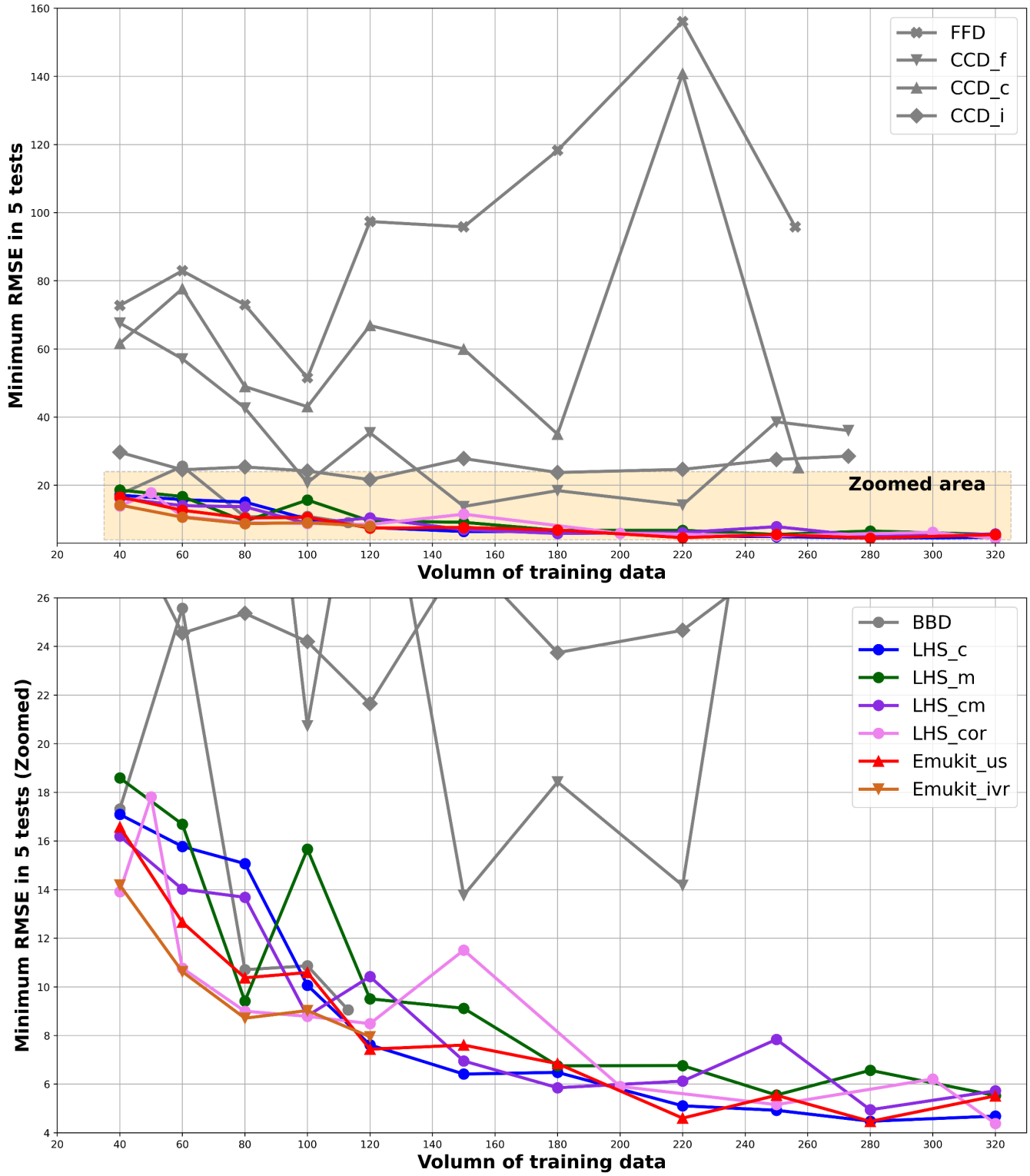


Figure 5. The potential of tested DOE strategies

C. *Settings of the Uncertainty*

Uniform distributed noise was added to the target parameter to mimic the process uncertainty described in Section II C. The reason for choosing uniform distribution over Gaussian distribution lies in the fact that Gaussian noise will produce a large number of low-level noise points around zero. Such noise points cannot represent the set level of uncertainty.

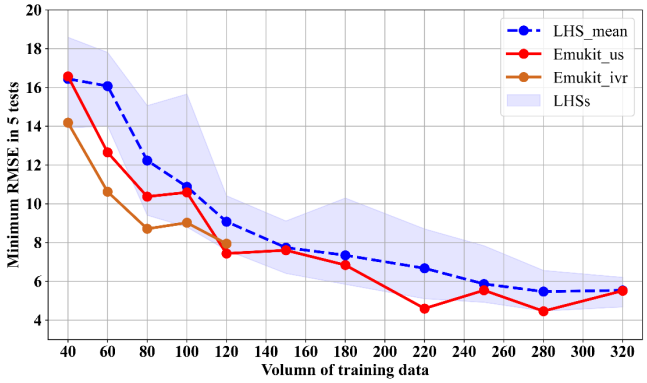


Figure 6. The potential of Emukit strategies compared to the average potential of LHSs

The tested range of process uncertainty was set to be 0-20% of the variation range of the target parameter. To generate the process uncertainty, the random function in the Python Numpy library was employed. Fig. 4 shows the distribution of 20 generated uncertainty points at the 10% level of uncertainty.

Owing to the instability in the noise points, ten sets of noise points were generated independently and added to the target parameter to create the ten different training datasets for each DOE strategy. For each training dataset, the modeling process was performed only once.

It should be noted that for conventional DOEs or LHS strategies, the uncertainty addition scheme adopted is to generate all data points without uncertainty all at once. The uncertainties are then added directly to the output displacement as the final step in the data generation. For Emukit, however, this scheme does not correspond to the actual experiment procedure. The following data generation scheme is iterated to generate uncertainty-containing training data for Emukit:

- fit a prediction model to the existing data
- find the next point with the highest marginal predictive variance as predicted by the prediction model
- add this new data point to the existing dataset

The test dataset without uncertainty was used to evaluate the trained models. The average performance of the ten trained models is recognized as the potential of the corresponding DOE used for the training dataset.

IV. Results and Discussions

A. Without uncertainty

For a relatively complex parameter space consisting of eight input factors, most of the conventional DOE methods cannot build a promising training dataset. As can be seen from the first half in Fig. 5, the performance of conventional DOEs (FFD, CCD_f, CCD_c, CCD_i) are not comparable to that of LHS or Emukit under the same amount of data. BBD is the best strategy among conventional DOEs, which performs almost similarly. However, as mentioned above, one of the major drawbacks of conventional DOEs is their inability to generate a specified amount of data as required. With the aid of D-optimal design, the BBD strategy is also only capable of planning data points within its given range. Such a drawback greatly limits the use of conventional DOE in the ML domain.

Also, the LHS and Emukit strategies outperform the conventional DOEs except for BBD at any amount of data. For the LHS family, with the exception of a few data points (LHS_m at 100, LHS_cor at 150), the LHSs perform essentially similarly with the same amount of data. It cannot be concluded that one certain LHS is necessarily better than other LHS strategies. As a kind of space-filling design, LHS is able to evenly distribute the limited data resource in a given data space to explore as much data space as possible. It is certainly a DOE suitable for ML data analysis.

Both Emukit strategies (Emukit_us & Emukit_ivr) are safe choices compared to the LHSs. In other words, Emukit strategies never perform the worst at any data volume, not to mention that the Emukit_ivr has the top performance with small data volumes (40 - 100).

Fig. 6 demonstrates this conclusion more clearly. The dashed line in Fig. 6 shows the average performance of the four LHS strategies. Both Emukit strategies outperform the average performance of LHSs over their data volume interval. This difference is particularly noticeable when the amount of data is relatively small (<120). Whereas, when the amount of data is sufficient (>250), the performance of LHSs can converge to Emukit_us. It can be concluded that one of Emukit's advantages is its ability to efficiently allocate data resources when data volumes are insufficient.

Both LHS and Emukit can generate DOEs with the requirement of training data volume based on the number of input factors. As an iterative scheme, Emukit is more flexible than space filling DOE: it can continuously generate additional data points besides existing data. In contrast, LHS requires that the amount of data volume be specified at the beginning, which isn't compatible with additional data generation.

But Emukit is not always the optimal choice. It needs an initial amount of data for subsequent iterations. If the model trained with the initial dataset does not drive Emukit correctly, then the results out of the iterations can be disastrous. This is further discussed in paragraph C in this section.

B. With uncertainty

According to the uncertainty generation scheme in Section III. A, 10 different sets of noisy data were generated for each dataset at each data volume. Most conventional DOEs showed inferior performance compared to LHSs. Thus, only CCD_i was selected from conventional DOEs for comparison in this phase. LHS_c from the LHS family was selected as a representative strategy. Since the CCD is a pre-set conventional DOE, the test range of CCD_i in the uncertainty test was set to 40-250. For LHS_c, the upper limit on the amount of data is extended to 700 for observing the improvement in model performance despite the existence of

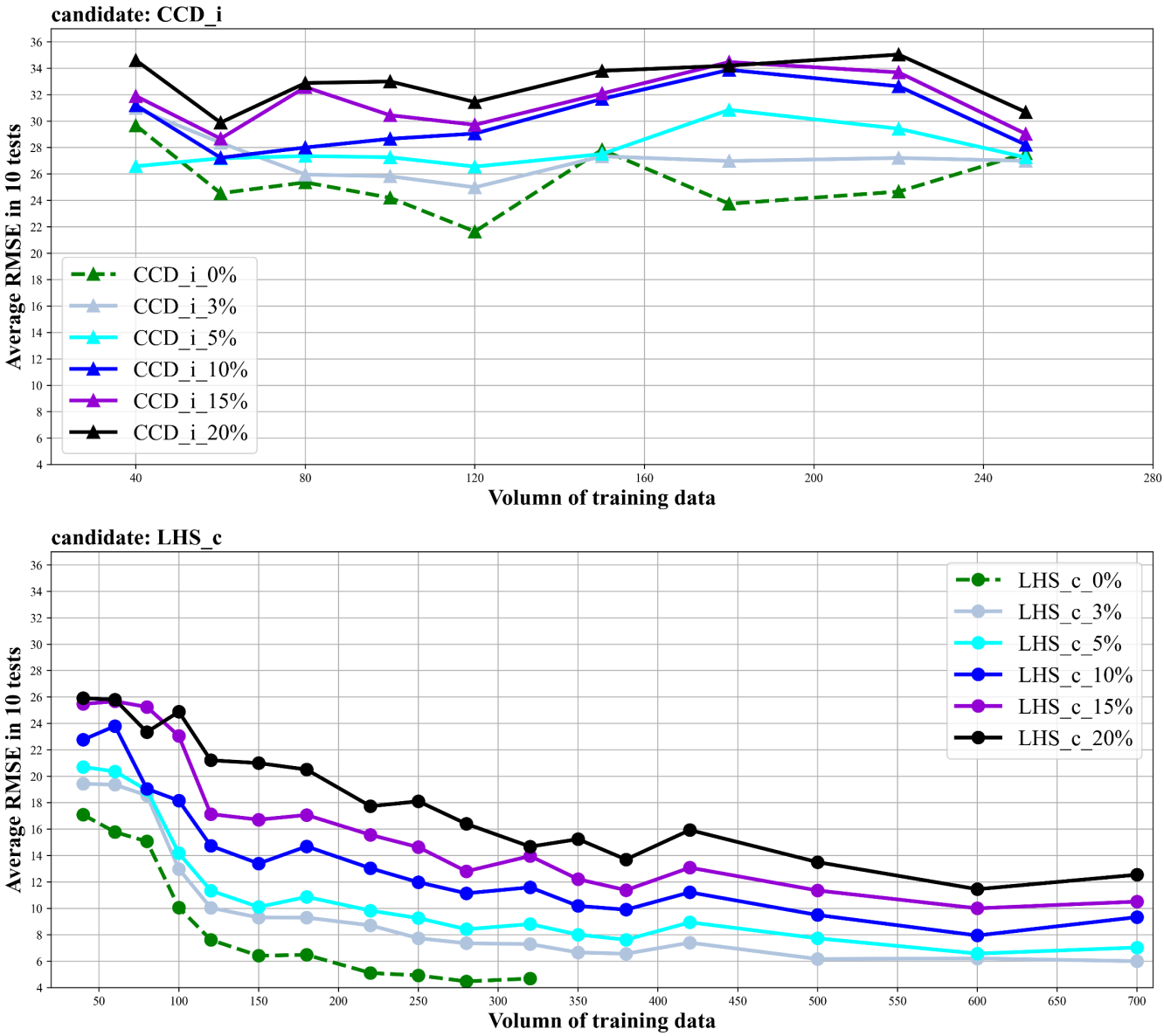


Figure 7. Potential of LHS_c and CCD_i strategies with varying uncertainties

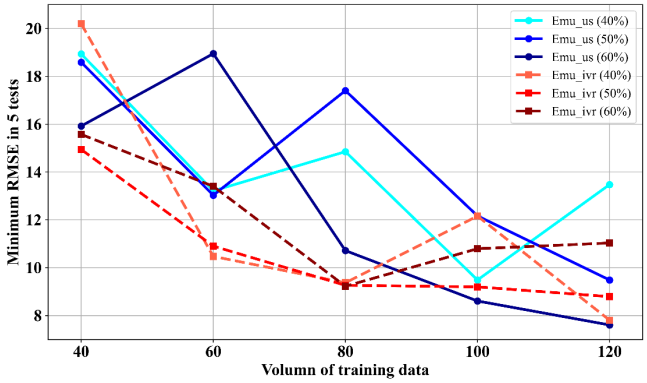


Figure 8. Potential of Emukit strategies with different settings

uncertainties. Emukit's performance under the impact of uncertainty is placed in paragraph D in this section. The experimental results are shown in both Fig. 7 and Fig. 9.

It is clear that for both of the measured DOEs, increasing uncertainty leads to deterioration of model performances. The experiment results of Emukit demonstrate the same trend. Therefore, this conclusion is generalizable to all three types (conventional, space-filling, model-based iterative) of DOE strategies.

It can be observed from the second half of the Fig. 9 that the adverse effect due to uncertainty is gradually compensated for as the amount of training data rises. In the case of LHS_c, for example, the performance of the model obtained using 600 noisy data with a 10% level of uncertainty is approximately the same as the performance of the model trained with 100 training data without any uncertainty. This suggests that "big

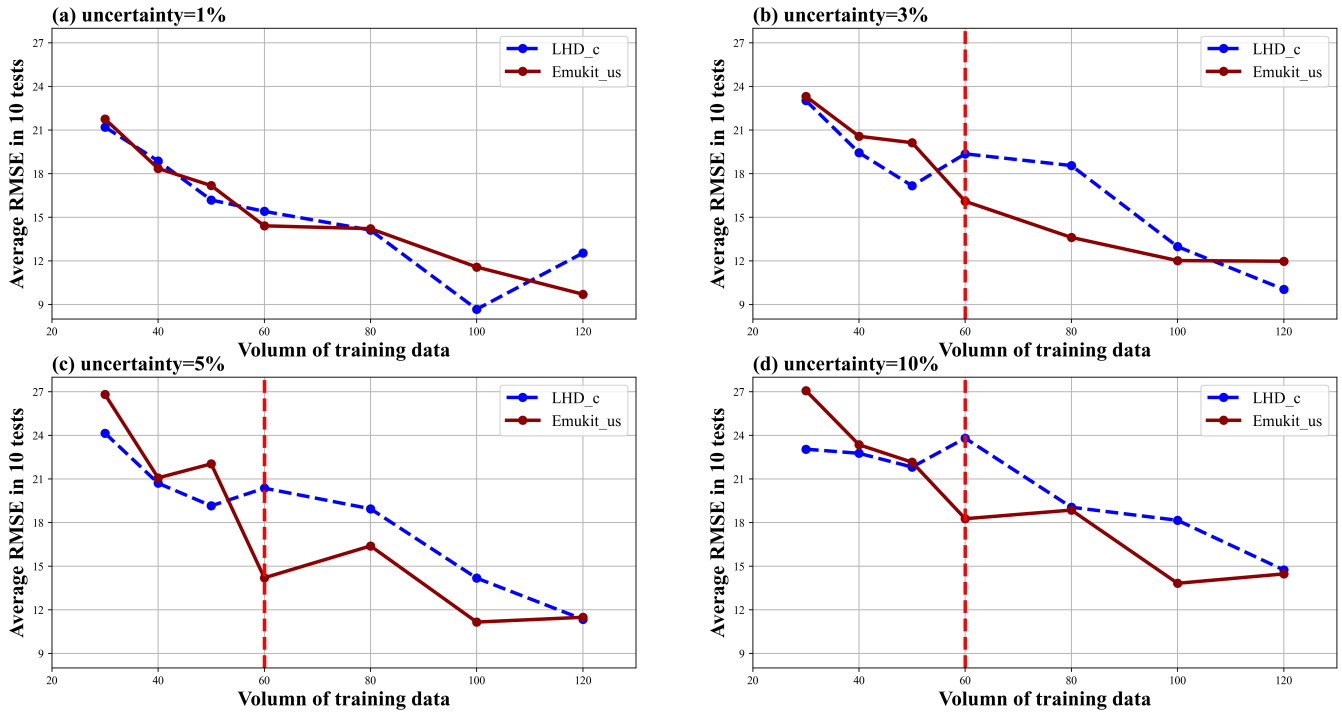


Figure 9. The potential of Emukit_us compared to the Potential of LHS_c in small-data context with varying uncertainties. In the case of small uncertainty (a) no significant differences can be observed. For larger uncertainty (b-d) Emukit outperforms LHS as soon as a critical amount of data (red line) becomes available.

data" can compensate the effect of uncertainty to some extent. However, no such trend appears in the results of CCD_i. Thus, it can be concluded that a space-filling design becomes preferable to a pre-determined conventional DOE under the influence of uncertainty. By employing more data, a predictive model, which is closer to the ground truth can be obtained. Even so, the effect of uncertainty remains at larger data volumes (600-700).

C. *Guide for iterative DOE: Emukit as an example*

As an iterative model-based DOE strategy, Emukit is governed by three hyperparameters:

- the integrated GP-model
- the acquisition functions
- initial data volume as a percentage of total data volume

It is not difficult to imagine that if the GP model has limited predictive power, then its predictions about data points are unreliable. Therefore, the first step in using Emukit is to optimize the GP model. The tuning of the GP model can be found in many references [24-27]. In this regard, the effects of the other two hyperparameters have been explored through experiments. A comparative experiment is conducted within the data volume from 40 to 120. The results are shown in Fig. 8.

As can be seen in Fig. 8, the potential of Emukit_ivr at small data volume (<80) outperforms that of Emukit_us. The advantage of the ivr acquisition function does not exist afterwards, where there is no longer a clear superior choice. Details about these acquisition functions in Emukit are available in [23] [25]. Considering the whole tested range of the training data, 50% initial data share is a safe choice for both acquisition functions. However, in case of extremely small data resources (<40), allocating more resources to the initial dataset seems to be a safe choice. It is also worth noting that the ivr acquisition is time consuming, which consumes at least twenty times as much time as the us acquisition. Considering the efficiency factor, us acquisition is a valid choice when the data resource is large enough.

Discussion on the usage of Emukit with small data resources continues with the interference of uncertainties. With this purpose, we conducted experiments with a training data volume of 30-120 and the tested uncertainty level was set to 3%-10%. Uncertainty was added according to the settings described in Section III C. LHS_c and Emukit_us (50% initial data share) were selected as candidates for the experiment. The results are recorded in Fig. 8. The discussion of the results is presented in paragraph D.

D. *Tutorial on DOE selection in small-data context with uncertainty*

In this section, we provide a preliminary generalization towards DOE selection based on our experimental results. Again, it is important to state that our conclusion towards DOE selection is restricted to ML regression models. The goal of the DOE is to explore the predefined parameter space for a prediction model. The selection of DOEs is considered on the

basis of "you only get one chance" principle. Therefore, in addition to comparing the best accuracy of the trained models that each DOE can deliver, the reliability of this DOE in the worst-case scenario also has a decisive influence. More specifically, a DOE strategy that consistently brings the models to an R^2 of around 0.8 regardless of the uncertainties is preferable to a DOE that only in the best-case scenario enables a model to reach 0.9 while, in other cases leaves the models only managing an R^2 score of 0.7.

An empirical conclusion from the ML community regarding the estimation of the required amount of training data is "two subjects per variable" (2SPV) rule of thumb [28] [29]. This rule is certainly influenced by the complexity of the model. The object of the unknown relationship lies in a multidimensional parameter space. A complex relationship between the target parameter and the input parameters demands a larger training dataset. Following this empirical law, an estimation (Est) of the amount of data required to mimic the exemplary FEM using multivariate linear regression with quadratic terms can be determined.

$$Est = 2 * (8 + 8 + 28 + 1) = 90 \quad (1)$$

The first term in brackets in (1) is the number of primary linear coefficients, the second and the third terms are the number of quadratic coefficients. At last, there is one constant coefficient. Therefor, for this FEM model, less than 90 data can be roughly recognized as a small-data context according to the 2SPV rule.

Both LHS family and Emukit strategies are appropriate candidates when the data resources far exceed (>2Est) small-data context. At this point, the main factor affecting the DOE selection is the time efficiency, which has been interpreted in Section IV A. The presence of uncertainty (<20%) leads to deterioration in model performance. To obtain well-performing models it requires more data to compensate for the uncertainty (see Fig.7).

The Emukit is the best choice in terms of best achievable prediction accuracy when the available data resource is 1-2 times the size of the small-data context (Est - 2Est). This choice is safe when the uncertainty level stays below 10%. The application of Emukit demands discretion when the uncertainty level goes higher. In such cases, LHSs are safe candidates.

The impact of uncertainty cannot be ignored in small-data context (<Est), where the available data resource is less than the estimation according to the empirical law. As shown in Fig. 8, for each uncertainty level, the amount of data for which Emukit exceeds LHS for the first time is marked with a red dotted line. It can be found that Emukit outperforms LHS only when the amount of data at its disposal exceeds 60. i.e., Emukit requires a minimal amount of initial training data in order to allocate data points correctly. If the uncertainty remains at a very low level (below 1%), Emukit could still be a good choice compared to LHS. As shown in the first plot of Fig. 8, the potential of LHS and Emukit are comparable within the data amount from 30 – 80. The above discussion on DOE selection is summarized in Table IV, where I denotes iterate sampling (represented by Emukit) and S denotes space-filling design (represented by LHS).

TABLE IV. APPROPRIATE DOE FOR DATA ACQUISITION

Uncertainty / Data volume	< 1 %	3% - 10%	> 10%
< 0.5Est	I ≈ S	S > I *	S
0.5Est - Est	I > S	S ≈ I *	S
Est - 2Est	I > S	I > S	S
> 2Est	I > S	I > S	S

Note that for cases marked with an asterisk in Table IV, iterative sampling is still reliable if the initial training dataset is able to yield a decent model until the effects of uncertainty become significant, or the available initial data is insufficient to enable the core model to deliver an effective predictive model. It is recommended to examine the performance of the model trained with the initial dataset. According to the experiments with Emukit, Gaussian Process models trained with limited initial data perform best if a positive R^2 score ($R^2 > 0$) can be reached.

V. CONCLUSION

This article discussed characteristic aspects of the "small data problem" with process uncertainties. The performance of some existing DOE strategies was tested with data collected from a self-built FEM simulation. The accuracy of different ML regression models trained with data collected according to a specific DOE at given data volume are systematically compared. The effect of uncertainties on different DOEs was also quantified experimentally.

On the basis of the experimental results, a preliminary discussion on how to select an appropriate DOE for data acquisition under the constraints of fixed data volume and a given level of measurement uncertainty is presented. Our study shows that space-filling design and iterative sampling strategy outperform conventional pre-determined DOE schemes for exploring tasks. The iterative sampling strategy is even superior to space-filling design in an ideal scenario with almost no uncertainty (<1%). However, when the effects of process uncertainty cannot be ignored (>3%), model-based iterative sampling strategy requires a certain amount of initial data to obtain a functional kernel model. In such circumstances, space-filling strategy is a safe alternative, particularly when data resources are constrained. Furthermore, we give recommendations on how to correctly drive a model-based iterative sampling strategy.

In subsequent work, we will extend this research procedure to multiple models of varying complexity with a view to generalizing our conclusions about the DOE selection. Other sorts of process uncertainties will be taken into account.

CODE AVAILABILITY

The data generation scripts and the model training scripts mentioned in the paper and the associated data are compiled on Github: https://github.com/xinchengxxc/Small-Dataset-Acquisition-for-Machine-Learning-Analysis.

ACKNOWLEDGMENT

This work was supported by the German Federal Ministry of Education and Research (BMBF) through the project KIproBatt (grant number 03XP0309C) and the interdisciplinary PhD school (iDOK) at the University of Applied Sciences Aschaffenburg.

REFERENCES

[1] X. Xu, F. Conrad, A. Gronbach, and M. Möckel, "Small Dataset Acquisition for Machine Learning Analysis of Industrial Processes with Possible Uncertainties" The Ninth International Conference on Big Data, Small Data, Linked Data and Open Data (ALLDATA 2023) IARIA, Apr. 2023, pp. 35-38, ISSN: 2519-8386, ISBN: 978-1-68558-041-4.

[2] F. Conrad, M. Mälzer, M. Schwarzenberger, H. Wiemer, and S. Ihlenfeldt, "Benchmarking AutoML for regression tasks on small tabular data in materials design", Sci Rep, vol. 12, no. 1, Art. no. 1, pp. 19350, Nov. 2022, doi: 10.1038/s41598-022-23327-1.

[3] Figure Eight. *CrowdFlower: Data science report*. [Online]. Available from: https://visit.figure-eight.com/rs/416-ZBE-142/images/CrowdFlower_DataScienceReport_2016.pdf [retrieved: 02, 2023].

[4] R. Arboretti, R. Ceccato, L. Pegoraro, and L. Salmaso, "Design of Experiments and machine learning for product innovation: A systematic literature review", Quality and Reliability Engineering International, vol. 38, no. 2, pp. 1131–1156, Nov. 2022, doi: 10.1002/qre.3025.

[5] R. Arboretti, R. Ceccato, L. Pegoraro, and L. Salmaso, "Design choice and machine learning model performances", Qual Reliab Eng, vol. 38, pp. 3357-3378, Jan. 2022, doi: 10.1002/qre.3123.

[6] R. Fontana, A. Molena, L. Pegoraro, and L. Salmaso, "Design of experiments and machine learning with application to industrial experiments", Stat Papers, vol. 64, pp. 1251-1274, Mar. 2023, doi: 10.1007/s00362-023-01437-w.

[7] A. Dean, D. Voss and D. Draguljić, Design and Analysis of Expermients, 2nd Edition. New York, NY: Springer, 2017.

[8] KIproBatt. *Exploring smart battery cell production based on a generic system architecture and an AI-enhanced process monitoring*. [Online]. Available from: https://doi.org/10.13140/RG.2.2.11573.76006, 2023.11.13.

[9] M. Feurer, K. Eggensperger, S. Falkner, M. Lindauer, and F. Hutter, "Auto-sklearn 2.0: hands-free AutoML via meta-learning", J. Mach. Learn. Res., vol. 23, no. 1, p. 261:11936-261:11996, Jan. 2022.

[10] J. Fleischer, G. Lanza and K. Peter, "Quantified Interdependencies between Lean Methods and Production Figures in the Small Series Production," Manufacturing Systems and Technologies for the New Frontier, pp. 89–92, 2008, doi: 10.1007/978-1-84800-267-8_17.

[11] M. Westermeier, *Qualitätsorientierte Analyse komplexer Prozessketten am Beispiel der Herstellung von Batteriezellen.* [online]. Available from: https://www.mec.ed.tum.de/fileadmin/w00cbp/iwb/Institut/Dissertationen/322_Westermeier_Markus.pdf [retrieved: 02, 2023].

[12] D. Chicco, M. Warrens, and G. Jurman, "The coefficient of determination R-squared is more informative than SMAPE, MAE, MAPE, MSE and RMSE in regression analysis evaluation", PeerJ Computer Science, 7:e623, Jul. 2021, doi: 10.7717/peerj-cs.623.

[13] M. Feurer et al., "Efficient and Robust Automated Machine Learning", Advances in Neural Information Processing Systems, vol. 2, pp. 2962-2970, Dec. 2015, doi: 10.1007/978-3-030-05318-5_6.

[14] F. Fabris and A.A. Freitas, "Analysing the Overfit of the Auto-sklearn Automated Machine Learning Tool", In: International Conference on Machine Learning, Optimization, and Data Science, 2019. [Online]. Available from: https://api.semanticscholar.org/CorpusID:210131212.

[15] R.A. Fisher, The Arrangement of Field Experiments in Breakthroughs in Statistics. New York, NY: Springer, 1992.

[16] G. Box and D. W. Behnken, "Some New Three Level Designs for the Study of Quantitative Variables", Technometrics, vol. 2, no. 4, pp. 455-475, Nov. 1960, 2:4, doi: 10.1080/00401706.1960.10489912.

[17] F. Viana, "A Tutorial on Latin Hypercube Design of Experiments", Qual. Reliab. Engng. vol. 32, pp. 1975-1985, Nov. 2015, doi: 10.1002/qre.1924.

[18] J.-S. Park, "Optimal Latin-hypercube designs for computer experiments", Journal of Statistical Planning and Inference, vol. 39, no. 1, pp. 95-111, Apr. 1994, doi: 10.1016/0378-3758(94)90115-5.

[19] L. Salmaso et al., "Design of experiments and machine learning to improve robustness of predictive maintenance with application to a real case study", Communications in Statistics - Simulation and Computation, vol. 51, no. 2, pp. 570-582, Feb. 2022, doi: 10.1080/03610918.2019.1656740.

[20] Z. Liu et al., "Machine learning with knowledge constraints for process optimization of open-air perovskite solar cell manufacturing", Joule, vol. 6, no. 4, pp. 834-849, Apr. 2022, doi: 10.1016/j.joule.2022.03.003.

[21] M. Zhang, A. Parnell, D. Brabazon, and A. Benavoli, "Bayesian Optimisation for Sequential Experimental Design with Applications in Additive Manufacturing". arXiv, Nov. 23, 2021. doi: 10.48550/arXiv.2107.12809.

[22] A. Paleyes et al., "Emulation of physical processes with Emukit". arXiv, Oct. 25, 2021. doi: 10.48550/arXiv.2110.13293.

[23] P.F. de Aguiar, B. Bourguignon, M.S. Khots, D.L. Massart, and R. Phan-Than-Luu, "D-optimal designs", Chemometrics and Intelligent Laboratory Systems, vol. 30, no. 2, pp. 199-210, Oct. 1994, doi: 10.1016/0169-7439(94)00076-X.

[24] C. E. Rasmussen, Gaussian processes in machine learning in Advanced Lectures on Machine Learning (ML 2003), pp. 63-71. Berlin, Heidelberg: Springer, 2004. US

[25] G. Kopsiaftis, E. Protopapadakis, A. Voulodimos, N. Doulamis, and A. Mantoglou, "Gaussian Process Regression Tuned by Bayesian Optimization for Seawater Intrusion Prediction", Computational Intelligence and Neuroscience, vol. 2019, p. e2859429, Jan. 2019, doi: 10.1155/2019/2859429.

[26] J. Sacks, W. J. Welch, T. J. Mitchell, and H. P. Wynn, "Design and Analysis of Computer Experiments", Statistical Science, vol. 4, no. 4, pp. 409-423, Nov. 1989, doi: 10.1214/ss/1177012413.

[27] X. Yue, Y. Wen, J. H. Hunt, and J. Shi, "Active Learning for Gaussian Process Considering Uncertainties With Application to Shape Control of Composite Fuselage," in IEEE Transactions on Automation Science and Engineering, vol. 18, no. 1, pp. 36-46, Jan. 2021, doi: 10.1109/TASE.2020.2990401.

[28] R. Figueroa, Q. Zeng-Treitler, S. Kandula, and L. Ngo, "Predicting sample size required for classification performance", BMC Med Inform Decis Mak, vol. 12, no. 8, Feb. 2012, doi: 10.1186/1472-6947-12-8.

[29] P. Austin and E. Steyerberg, "The number of subjects per variable required in linear regression analyses", J Clin Epidemiol, vol. 68, no. 6, pp. 627-636, Jan. 2015, doi: 10.1016/j.jclinepi.2014.12.014.

Lightweight Approach to Java Sample Code Recommendation System Using Apriori-Based Soft Clustering

Yoshihisa Udagawa
Faculty of Informatics
Tokyo University of Information Sciences
Chiba-city, Chiba, Japan
e-mail: yu207233@rsch.tuis.ac.jp

***Abstract*— One effective way of learning programming techniques is to refer to sample programs. However, it becomes difficult and time-consuming to find a suitable sample program visually for a complex programming subject. To overcome this shortcoming, research and development of recommendation systems for software engineering have been actively conducted. This paper discusses a recommendation system for Java sample programs using unsupervised machine learning techniques. The proposed system includes three major steps: (1) extracting invoked methods used in each sample program, (2) soft clustering the sample program by applying a data mining technique to the extracted methods, and (3) ranking programs in a cluster by calculating a weighted average concerning the extracted methods. Experiments using sample programs related to a graphical user interface and string handling have confirmed the effectiveness of the proposed recommendation system. A generative artificial intelligence model can successfully generate a description of the cluster using invoked method names that specify each soft cluster. The proposed recommendation system and a generative artificial intelligence model can collaborate for improving a programming education environment.**

Keywords-component; Recommendation System for Software Engineering; Maximal Frequent Itemset; Unsupervised Machine Learning; Soft clustering.

I. INTRODUCTION

This research paper is an extension of the previously reported contribution to the Java sample program recommendation system [1]. The study includes improvement of algorithms for analyzing program structure, and performing soft clustering. Additional experiments on programming subjects covering standard Java classes including a Graphical User Interface (GUI), string handling, file Input/Output (I/O), socket, multithreading, collection, etc. are performed.

It is widely recognized that sample programs provide an effective means for learning new programming techniques. In particular, sample programs for using Application Programming Interfaces (API) related to open-source programs are widely available on the Internet. Since the amount of publicly concerning sample programs becomes enormous, it might become time-consuming and error-prone to find an appropriate sample program visually. Over the past few decades, there has been a great deal of research and development on the software recommendation systems that provide useful programming information for students and developers.

Recommendation systems are originally employed in online stores and video/music websites, where rankings of items are calculated based on users' reactions and similarities among products and/or works. The recommendation system for software development is intended to assist programmer's effort. It is designed to deals with artifacts, such as sample programs, specifications, test cases and bug reports. Several techniques have been developed to collect, rank, and visualize similar artifacts based on various indicators reflecting their nature. These techniques are often specific to software engineering and cause a recommendation system to be called a Recommendation System for Software Engineering (RSSE) [2].

This paper discusses a sample program recommendation system using a soft clustering technique. The proposed system deals with Java sample programs that are collected from the Internet. Assuming that a characteristic of a Java program is determined by the API calls, the name of a declared method and the names of invoked methods are extracted from these sample programs. The system automatically clusters Java sample programs based on the invoked methods applying a data mining technique named *Apriori* algorithm [3]. Because the *Apriori* algorithm is based on a set theoretic relation, the algorithm implements soft clustering, where one sample program belongs to multiple clusters. The system ranks the sample programs in each cluster using a Term Frequency-Inverse Document Frequency (*tf-idf*) [4] or weighted vector space model. Experiments confirm that higher ranked samples tend to contain more types of invoked methods than those ranked lower, which means this system assists a student in selecting sample programs suitable for learning.

The contributions of this study are as follows:

I. In general, API call patterns differ from one programming subject to another. This system can soft cluster sample programs for each programming subjects based on the API call patterns. This process is automatic, as the system automatically determines parameters for optimal soft clustering, which is newly implemented in this study.

II. The RSSEs proposed so far employ hard clustering, if any. In hard clustering, the results depend on the initial values and have the restriction that one sample belongs to only one cluster. This study employes soft clustering supported by a set theoretic relation. Therefore, a sample program can belong to multiple clusters, and a cluster only contains programs related by set theory. Soft clustering provides the optimal access paths that reflect characteristics of the sample program.
III. By modifying *tf-idf* to give greater weights to the methods that are used to define a cluster, sample programs that fit the subject of a cluster and include rare APIs are ranked higher.
IV. The proposed system employs unsupervised machine learning, making it lightweight to use, operate and maintain the system. In fact, simply by collecting sample programs and running the proposed system, a student can get suitable sample programs to support his/her learning.

The rest of this paper is organized as follows. Section II describes the state-of-the-art research on the RSSEs. Section III overviews the proposed system. Section IV describes the implementation of the main functions of the proposed system. Section V shows the experimental results using typical Java programming techniques. Section VI discusses other implementation options and collaboration with a generative AI model. Section VII concludes the paper with our plans for future work.

II. State-of-the-Art Research

This section outlines recent studies concerning recommendation systems for software engineering. Technically, they can be broadly classified into clustering, pattern mining, and similarity. Many studies use multiple techniques.

A. Survey

Gasparic and Janes [5] survey 46 research and development articles on RSSE published between 2003 and 2013, and categorize them with respect to covered data and methods for recommendation. The most common type of covered data is source code with 21 papers, followed by help information to perform source code changes with 6 papers. As for the recommendation methods, list format is the most common with 33 papers, followed by document format with three papers, and table format with two papers.

Ko, Lee, Park, and Choi [6] discuss the recommendation system research trends from a macro perspective using top-ranking articles and conference papers electrically published between 2010 and 2021. The study analyzes how the recommendation models and technologies are utilized in seven main service fields including education service and academic information service. Smart education that accesses vast digital resources has stimulated a rapid increase of educational recommendation systems. The goal of the systems is to provide learners with personalized educational materials.

B. Clustering

Katirtzis, Diamantopoulos, and Sutton [7] discuss an algorithm that extracts API call sequences and then clusters them to create an API usage summary known as a source code snippet. Hierarchical clustering is performed by calculating the distance of extracted API call sequences using the longest common subsequence (LCS) algorithm [8]. Then, code slice techniques are applied to create a source code snippet.

Chen, Peng, Chen, Sun, Xing, Wang, and Zhao [9] propose an approach for API sequence recommendation with three strategies, i.e., heuristic search using a modified longest common subsequence algorithm, clustering API sequence using a hierarchical clustering algorithm, and summarizing API sequence recommendations. They use the clustering to make it easier for programmers to find similar API recommendations and to facilitate the API selection. Since they use a modified hierarchical clustering algorithm, each API is always hard clustered belonging to just one cluster.

C. Pattern Mining

Hsu and Lin [10] propose a recommendation system based on frequent patterns in source code. They originally define 17 syntax patterns and extract them from the source code under study. A sequence pattern extraction algorithm based on frequency known as *Prefix-Span* [11] is applied to generate recommended API usage patterns.

Chen, Gao, Ren, Peng, Xia, and Lyu [12] discuss a method to mine the usage patterns of low frequency APIs. Their method is based on three views, i.e., method-API relationship for local view, API-API co-occurrence for global view, and project structure for external view. With experiments of several hundreds of Java projects, their method is confirmed to achieve an increased rate for retrieving the low-frequency APIs.

D. Similarity

Diamantopoulos and Symeonidis [13] develop a system to recommend sample code stored in software repositories on the Internet, such as GitHub, GitLab and Bitbucket. The input to the system is a code fragment presented by a user, and the output is a set of sample codes similar to the code fragment. Similarities among source codes are calculated based on the vector space model and the Levenshtein distance [14].

Hora [15] discusses a source code recommendation system that analyzes source code contained in a particular project and creates ranked API usage examples on a web site. The system ranks the source code based on three quality measures, i.e., similarity, readability, and reusability. The similarity is calculated using the cosine similarity [4][16] in data analysis, while readability and reusability are calculated using indicators developed in software engineering studies.

Nguyen, Rocco, Sipio, Ruscio, and Penta [17] implement a system to present API usage in a timely manner during a coding process and discuss the evaluation of experimental results. The system calculates the similarity among similar projects by *tf-idf* and ranks API usage patterns using a collaborative filtering technique [18].

E. Approach of this Study

This study concerns a recommendation system for sample programs based on API call patterns, which is similar to many of the studies described in this section. The system first soft clusters sample programs based on a set of frequently occurring APIs. Next, the *tf-idf* model is used to calculate the recommendation of the programs belonging to each cluster. The significant difference from previous studies is the implementation of soft clustering that allows a single program to belong to multiple clusters. The study also automatically adjusts a clustering parameter to optimize the number of clusters. This implementation allows us to efficiently handle the hundreds of sample programs required in programming education.

III. OVERVIEW OF PROPOSED SYSTEM

This section describes the architecture of the proposed system from the functional point of view and outlines typical usage with an example from experiments performed in this study.

A. Architecture

Figure 1 depicts the architecture of the proposed system. The input for this system is a collection of sample programs stored in the *Sample code repository*. Currently, these sample programs are manually collected from the Internet, and stored in a specific project typically in *Eclipse*, an Integrated Development Environment (IDE) for Java [19]. In this study, we assume that all sample programs are correct and work properly.

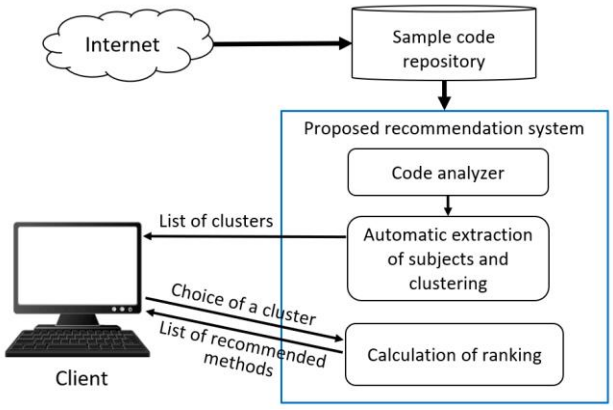


Figure 1. Overview of the proposed system.

Java programming techniques typically classified into several subjects, such as File I/O, collection, GUI, socket, and multithreading. This classification is widely accepted in programming education. The proposed system is designed to store Java sample programs in multiple packages or categories. Figure 2 shows the package structure used in this study, which is stored in a project of *Eclipse* named *Sample_Code*.

Programming education typically requires several to thirty Java sample programs in a package, though there is no limit to the number of Java files to include in each package. The *File_IO.Sample_1* and *File_IO.Sample_2* packages contain a set of sample programs for file IO, which is used for the experiments described in the previous paper [1].

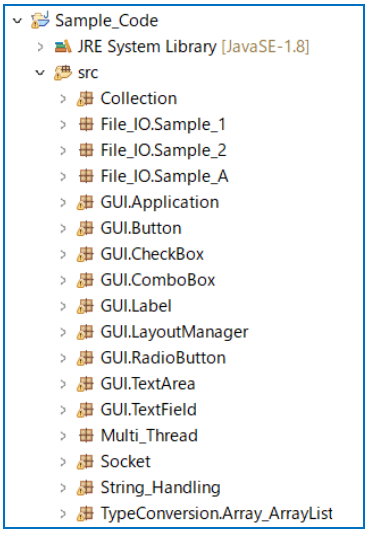


Figure 2. Package structure of *Sample_code*.

More than ten packages covering typical Java programming subjects are newly added. The *GUI.ComboBox* and *String_Handling* packages are used for the experiments described in the rest of this paper.

B. Starting Code Analyzer

The initial GUI screen of the proposed system contains only one *JComboBox* with the top directory of sample programs as an argument. The user of this system can view the package structure of the sample programs, and select one of the packages by pulling down the *JComboBox*. Figure 3 shows the screen dump that selects the *GUI.ComboBox* package.

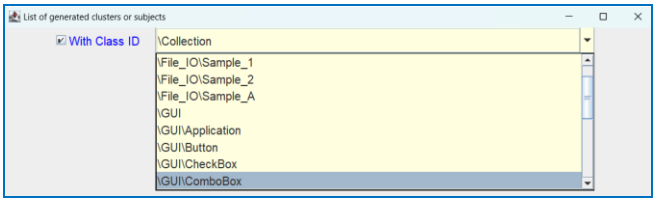


Figure 3. Screen dump for selecting the *GUI.ComboBox* package.

Then, the *code analyzer* in Figure 1 starts to extract declared method names and invoked method names from all Java files under the selected package or directory. A list of invoked method names is used for clustering the declared methods and ranking them.

Since Java allows a class to define its own methods, the same method name can be defined across multiple classes. In a Java program, a non-static method needs a reference variable to identify the class to which the method belongs. In this study, a new function to convert variable names to

class names is implemented in the *code analyzer* in order to uniquely distinguish non-static methods.

For example, Java has the *HashMap* and *TreeMap* classes. The both classes have the non-static *put* methods to insert an element to the map classes. The *code analyzer* generates *HashMap.put* and/or *TreeMap.put* by converting a reference variable to a class name. This conversion process is newly implemented in this study, and allows us to identify the difference between the *put* method in the *HashMap* class method and that in the *TreeMap* class.

C. Automatic Identification of Subjects and Clusters

Following code analysis, the *Apriori* algorithm [3] runs to identify the set of invoked methods that occur frequently. Programming subjects are automatically identified based on the frequent method set. Each subject corresponds to a cluster featured by the frequent method name set. Figure 4 shows an example of clustering with 17 identified clusters for the *GUI.ComboBox* package that includes 18 Java files and 45 declared methods.

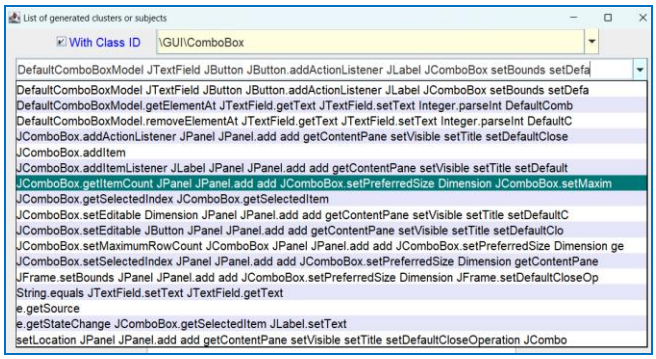


Figure 4. Identified programming subjects or clusters with class name.

Every non-static method name is preceded by a class name in Figure 4. Sometimes class names are so long that it is better to omit them for the purpose of a concise display. Unchecking the *With Class ID* checkbox at the top left corner of the initial GUI, the method names without class names are displayed as shown in Figure 5.

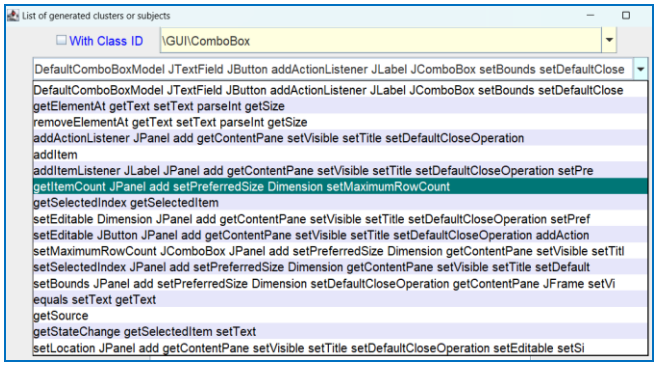


Figure 5. Identified programming subjects or clusters without class name.

Strictly, this study uses a soft clustering technique based on a maximal frequent itemset [20], i.e., a compact itemset that represents a frequent itemset. The method names displayed in Figures 4 and 5 are elements of a maximal frequent itemset. For example, "*getItemCount JPanel add setPreferredSize Dimension setMaximumRowCount"* suggests from the method names that the cluster is related to the programming techniques that specify the number of elements in a *JComboBox*, the size of a *JComboBox*, and the maximum number of rows that can be displayed.

D. Calculation of Recommended Ranking

Selecting an element in the *JComboBox* shown in Figures 4 and 5 causes to specify a cluster of methods, which starts calculations of recommendation values for each of the declared methods in the cluster. Figure 6 shows an example of a method recommendation. The values of recommendation for each declared method are normalized so that the maximum value is equal to one.

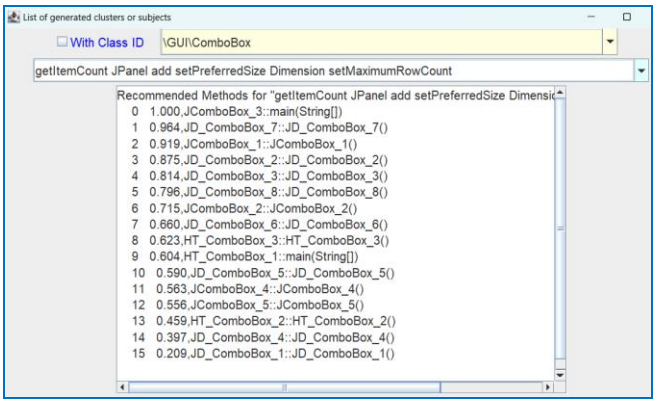


Figure 6. Sample of program recommendation.

Each method name is prefixed with a class name or a Java file name, so that a student can easily find out source code using an IDE, such as *Eclipse, NetBeans and IntelliJ IDEA.*

IV. IMPLEMENTATION

This section describes the implementation of three major steps for generating a recommendation. Those steps are code analysis, soft clustering, and ranking.

A. Code Analysis for Extracting Invoked Method Set

Functions necessary for system development are typically provided as runtime methods in Java. After learning the control structure of programs and object-oriented techniques, students and developers enhance their programming skills by learning how to use the runtime methods provided by Java communities. Therefore, the methods being invoked are closely related to the functionality of the programs under development. In this study, we assume that program similarity can be computed by the similarity of the method sets being invoked.

The *code analyzer* in Figure 1 extracts a declared method signature and a set of invoked method names. We implemented the *code analyzer* using the *Scanner* class [21], a tokenizer in Eclipse Java Development Tools (JDT) core.

This class provides the functionality to classify the tokens in a Java program into more than 100 types, and excludes comments for facilitating efficient analysis of executable statements. The *Scanner* class is also used in *Eclipse* [19] for navigating Java programs, including a class-method hierarchy and a list of field variables.

Figure 7 shows a sample of a Java program. Figure 8 shows the declared method signature and a list of invoked method names that are extracted from the Java program. A method or API with the same name is usually invoked multiple times in a declared method. Therefore, the *code analyzer* extracts the invoked method name and the number of times invoked, which are used for calculating cosine similarity [4][16]. For example, the *main* method in the *JComboBox_3* class in Figure 7 invokes the *Dimension* method twice, and *JComboBox.setPreferredSize* method twice, etc.

```
1  package GUI.ComboBox;
2  //  A sample code using "setMaximumRowCount()"
3  import javax.swing.*;
4  import java.awt.Dimension;
5
6  public class JComboBox_3 {
7      public static void main(String[] args){
8          String[] color= {"Light red", "Red", "Dark red",
9                  "Light blue", "Blue", "Dark blue",
10                 "Light green", "Green", "Dark green"};
11         JPanel p = new JPanel();
12         //  Show 8 elements by default
13         JComboBox<String> comboA = new JComboBox<>(color);
14         comboA.setPreferredSize(new Dimension(120,25));
15         p.add(comboA);
16         // Pull-down to display all elements
17         JComboBox<String> comboB = new JComboBox<>(color);
18         comboB.setMaximumRowCount(comboB.getItemCount());
19         comboB.setPreferredSize(new Dimension(120,25));
20         p.add(comboB);
21
22         JFrame fm= new JFrame();
23         fm.getContentPane().add(p);
24         fm.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
25         fm.setBounds(10, 10, 400, 200);
26         fm.setTitle("JComboBox_3");
27         fm.setVisible(true);
28         System.out.println("--- Finished ---");
29     }
```

Figure 7. Sample Java program named *JComboBox_3.java.*

```
JComboBox_3::main(String[])
        Dimension, 2
        JComboBox.getItemCount, 1
        JComboBox.setMaximumRowCount, 1
        JComboBox.setPreferredSize, 2
        JFrame, 1
        JFrame.getContentPane, 1
        JFrame.setBounds, 1
        JFrame.setDefaultCloseOperation, 1
        JFrame.setTitle, 1
        JFrame.setVisible, 1
        JPanel, 1
        JPanel.add, 2
        add, 1
```

Figure 8. Invoked method names and the number of invoked times.

It should be noted that the methods, such as *println()* and *printStackTrace()*, are intentionally excluded from the extraction process because they are often used to print data values for debugging purpose. They are considered to fail to characterize the function of a declared method.

B. Soft Clustering Based on Apriori Algorithm

1) Apriori algorithm and maximal frequent itemset

Apriori algorithm proposed by Agrawal and Srikant [3] starts by identifying the frequent individual items of length one, and extending them to larger itemset as long as those itemset frequently appear in the database under consideration.

Let us a database D be a set of transactions t, i.e., $D= \{t_1, t_2,\ldots, t_n\}$. Let us each transaction t_i be a nonempty set of itemset, i.e., $t_i = \{i_{i1}, i_{i2},\ldots, i_{im}\}$. The itemset is a nonempty set of items observed together.

The support value of an itemset is defined as the number of transactions in the database D. Using terms of the database D and transaction t_i, the support value of an itemset X is defined by the following formula:

$$Support(X)= |\{ t_i \in D : X \subseteq t_i \ \& \ 1 \le i \le n \}| \quad (1)$$

A set of items is called frequent if its support value is greater than a user-specified minimum support value, i.e., *minSup*.

Here, we cite the *Apriori* principle:

> *If an itemset is frequent, then all of its subsets are also frequent.*

This means that if a set is infrequent, then all of its supersets are infrequent. The *Apriori* algorithm works based on this principle, in which the frequent item sets of length k are utilized to identify frequent item sets of length $k+1$.

Since the frequent itemset generated by the *Apriori* algorithm tends to be very large, it is beneficial to identify a compact representation of all the frequent itemset. One such approach is to use a maximal frequent itemset [20].

Definition:

A maximal frequent itemset is a frequent itemset for which none of its immediate supersets are frequent.

Table I shows an example of a database consisting of five transactions of itemset.

TABLE I. EXAMPLE OF DATABASE

Transaction ID	Item set
1	A B C
2	A C D
3	A D
4	B C
5	B C D

Figure 9 illustrates an example of the maximal frequent itemset in a lattice structure where a node corresponds to an itemset and arcs correspond to the subset relation [20]. *MinSup* is set to 20% (= 1/5*100). Since the number of transactions in the database is 5, *minSup* 20% means if an

itemset appears once or more than once, it is frequent. In Figure 9, the nodes surrounded by solid lines indicate the frequent itemset, while the nodes with yellow backgrounds indicate the maximal frequent itemset. By definition, the maximal frequent itemset forms the boundary between frequent and infrequent itemset.

All frequent itemset can be derived from the set of maximal itemset. In Figure 9, the following three sets of itemset are generated form the maximal frequent itemset:

{{A, B, C} {A, B}, {A, C}, {B, C}, {A}, {B}, {C}}
{{A, C, D} {A, C}, {A, D}, {C, D}, {A}, {C}, {D}}
{{B, C, D} {B, C}, {B, D}, {C, D}, {B}, {C}, {D}}.

Each maximal frequent itemset defines a soft cluster of itemset where one element belongs to multiple clusters. For example, the itemset {A, C} belongs to the two clusters {A, B, C} and {A, C, D}.

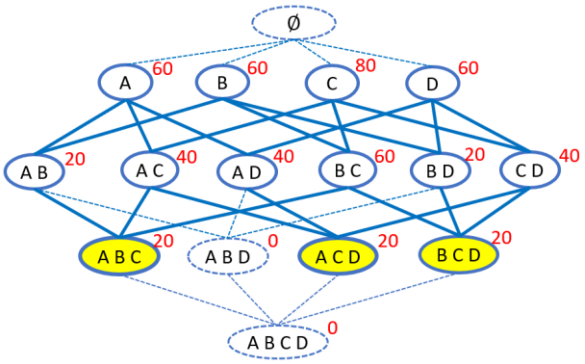


Figure 9. Maximal frequent itemset in lattice structure with 20% *minSup*.

Figure 10 illustrates an example of the maximal frequent itemset with *minSup* of 40%. In the case of Figure 10, the following four sets of itemset are derived:

{{A, C}, {A}, {C}}
{{A, D}, {A}, {D}}
{{B, C}, {B}, {C}}
{{C, D}, {C}, {D}}.

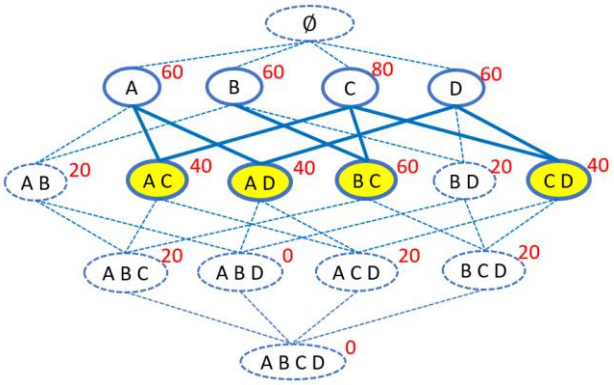


Figure 10. Maximal frequent itemset in lattice structure with 40% *minSup*.

It should be noted that the maximal frequent itemset, and thus the number of elements in the itemset, changes according to the value of *minSup*. In the proposed system, the value of *minSup* is varied by 1% to find the *minSup* that produces the maximal frequent itemset with the largest number of itemsets.

Table II shows some of the package names containing Java programs used in the experiment, the number of Java files, and the number of declared methods. Because *String* and *Collection* APIs are rather simple to use, only one declared method, i.e., *main*, is used in each Java file. Therefore, the number of Java files is equal to the number of declared methods. In contrast, the API related to *GUI* is complex to use, and multiple declared methods are used in a Java file. Therefore, the number of declared methods is larger than the number of Java files.

TABLE II. NUMBER OF JAVA FILES AND METHODS

Package	No. of Java files	No. of methods
Collection	11	11
String_Handring	31	31
File_IO	30	40
GUI.Label	6	12
GUI.ComboBox	18	45
GUI	72	152

Figure 11 shows the value of *minSup* and the number of clusters or itemsets in the maximal frequent itemset for each package. The maximum number of clusters is reached when the *munSup* is between 4% and 6%.

In general, there is a certain trend between the number of declared methods and the number of clusters. The *Collection* and *GUI.Label* packages have the eleven declared methods and the twelve declared methods, respectively. The number of clusters is maximized when *minSup* is between 4% and about 9%. The *File_IO* and *GUI.Combobox* packages have the 40 methods and the 45 methods, respectively. The maximum number of clusters is observed when *minSup* is between 4% and 6%. The *GUI* package consists of nine sub-packages and contains 152 declared methods. Maximum number of clusters occurs when *minSup* is 4%.

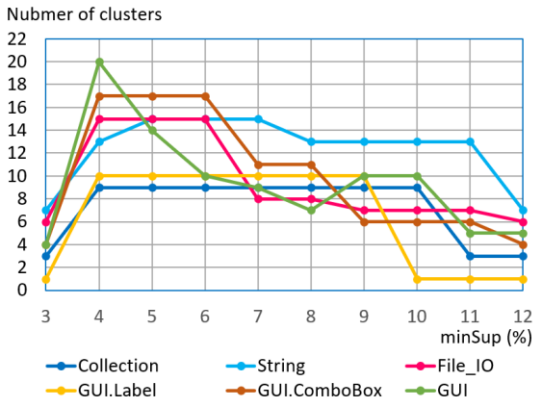


Figure 11. Values of *minSup* and the number of clusters.

In the current implementation, *minSup* is varied from 3% to 12% to count the number of generated clusters. Then, the

minSup that maximizes the number of clusters is determined. The lists of invoked methods shown in Figures 4 and 5 present clusters of a *GUI.ComboBox* with a *minSup* of 4%.

2) Soft clustering sample programs

More than ten binary programs that implement the *Apriori* algorithm are available on the web page maintained by Borgelt [22]. For the sake of openness and efficiency of implementation, this study uses *fpgrowth.exe* listed on the web page. Specifically, we implement a maximal-frequent-itemset generating function by calling *fpgrowth.exe* using *java.lang.Runtime.exec()* that executes the specified command and arguments as a separated process. The input data for this program is the set of invoked methods for each declared method, which is generated by the *code analyzer* ignoring the number of invoked methods citations. The result of running *fpgrowth.exe* is written to a file. Next, this file is read by the proposed recommendation system, which implements the linkage with the *Apriori* algorithm.

Figure 12 shows the maximal frequent itemset generated from the sample programs in *GUI.ComboBox* package shown in Figure 5, with a *minSup* of 4%. The maximal frequent itemset corresponds to the programming subjects. Figure 4 shows the complete set of method names preceded by the class name, which is actually used to calculate the recommended values.

```
0) DefaultComboBoxModel JTextField JButton addActionListener JLabel
   JComboBox setBounds setDefaultCloseOperation setTitle setVisible
   getContentPane Dimension setPreferredSize add JPanel
1) getElementAt getText setText parseInt getSize
2) removeElementAt getText setText parseInt getSize
3) addActionListener JPanel add getContentPane setVisible setTitle
   setDefaultCloseOperation
4) addItem
5) addItemListener JLabel JPanel add getContentPane setVisible setTitle
   setDefaultCloseOperation setPreferredSize Dimension setBounds
   JComboBox
6) getItemCount JPanel add setPreferredSize Dimension
   setMaximumRowCount
7) getSelectedIndex getSelectedItem
8) setEditable Dimension JPanel add getContentPane setVisible setTitle
   setDefaultCloseOperation setPreferredSize
9) setEditable JButton JPanel add getContentPane setVisible setTitle
   setDefaultCloseOperation addActionListener
10) setMaximumRowCount JComboBox JPanel add setPreferredSize
   Dimension getContentPane setVisible setTitle setDefaultCloseOperation
   setBounds
11) setSelectedIndex JPanel add setPreferredSize Dimension
   getContentPane setVisible setTitle setDefaultCloseOperation setBounds
   JComboBox
10) setMaximumRowCount JComboBox JPanel add setPreferredSize
   Dimension getContentPane setVisible setTitle setDefaultCloseOperation
   setBounds
11) setSelectedIndex JPanel add setPreferredSize Dimension
   getContentPane setVisible setTitle setDefaultCloseOperation setBounds
   JComboBox
12) setBounds JPanel add setPreferredSize Dimension
   setDefaultCloseOperation getContentPane JFrame setVisible
13) equals setText getText
14) getSource
15) getStateChange getSelectedItem setText
16) setLocation JPanel add getContentPane setVisible setTitle
   setDefaultCloseOperation setEditable setSize
```

Figure 12. Example of generated maximal frequent itemset.

Figure 13 shows a list of declared methods that contain at least one invoked method name that is included in a maximal frequent itemset. For example, clusters 1, 2, 4, and 7 are about *actionPerformed*, and *itemStateChanged*.

```
0)  DefaultComboBoxModel JTextField JButton addActionListener
    JLabel JComboBox setBounds setDefaultCloseOperation setTitle
    setVisible getContentPane Dimension setPreferredSize add JPanel
    <Intentionally omitted>
1)  getElementAt getText setText parseInt getSize
    JComboBox_2::actionPerformed(ActionEvent)
    JD_ComboBox_6::actionPerformed(ActionEvent)
    JD_ComboBox_7::actionPerformed(ActionEvent)
    JD_ComboBox_8::actionPerformed(ActionEvent)
2)  removeElementAt getText setText parseInt getSize
    JComboBox_2::actionPerformed(ActionEvent)
    JD_ComboBox_6::actionPerformed(ActionEvent)
    JD_ComboBox_7::actionPerformed(ActionEvent)
    JD_ComboBox_8::actionPerformed(ActionEvent)
3)  addActionListener add JPanel getContentPane setVisible setTitle
    setDefaultCloseOperation
    HT_ComboBox_1::main(String[])
    HT_ComboBox_2::HT_ComboBox_2()
    HT_ComboBox_3::HT_ComboBox_3()
    JComboBox_1::JComboBox_1()
    JComboBox_2::JComboBox_2()
    JComboBox_3::main(String[])
    JComboBox_4::JComboBox_4()
    JComboBox_5::JComboBox_5()
    JD_ComboBox_1::JD_ComboBox_1()
    JD_ComboBox_2::JD_ComboBox_2()
    JD_ComboBox_3::JD_ComboBox_3()
    JD_ComboBox_4::JD_ComboBox_4()
    JD_ComboBox_5::JD_ComboBox_5()
    JD_ComboBox_6::JD_ComboBox_6()
    JD_ComboBox_7::JD_ComboBox_7()
    JD_ComboBox_8::JD_ComboBox_8()
4)  addItem
    HT_ComboBox_3::actionPerformed(ActionEvent)
    JComboBox_1::JComboBox_1()
    JComboBox_1::actionPerformed(ActionEvent)
5)  addItemListener JLabel JPanel add getContentPane setVisible
    setTitle setDefaultCloseOperation setPreferredSize Dimension
    setBounds JComboBox
    <Intentionally omitted>
6)  getItemCount JPanel add setPreferredSize Dimension
    setMaximumRowCount
    HT_ComboBox_1::main(String[])
    HT_ComboBox_2::HT_ComboBox_2()
    HT_ComboBox_3::HT_ComboBox_3()
    JComboBox_1::JComboBox_1()
    JComboBox_2::JComboBox_2()
    JComboBox_3::main(String[])
    JComboBox_4::JComboBox_4()
    JComboBox_5::JComboBox_5()
    JD_ComboBox_1::JD_ComboBox_1()
    JD_ComboBox_2::JD_ComboBox_2()
    JD_ComboBox_3::JD_ComboBox_3()
    JD_ComboBox_4::JD_ComboBox_4()
    JD_ComboBox_5::JD_ComboBox_5()
    JD_ComboBox_6::JD_ComboBox_6()
    JD_ComboBox_7::JD_ComboBox_7()
    JD_ComboBox_8::JD_ComboBox_8()
7)  getSelectedIndex getSelectedItem
    HT_ComboBox_3::actionPerformed(ActionEvent)
    HT_ComboBox_3::itemStateChanged(ItemEvent)
    JComboBox_4::itemStateChanged(ItemEvent)
    JD_ComboBox_3::actionPerformed(ActionEvent)
    JD_ComboBox_5::itemStateChanged(ItemEvent)
8 to 16  <Intentionally omitted>
```

Figure 13. Declared methods belonging to each cluster.

Cluster 3 is about how to create a GUI containing a *ComboBox*. Cluster 6 is related to the *JComboBox* property settings. Since the *JComboBox* needs to be placed in a screen frame, the API for *JFrame*, e.g., lines 22-27 of Figure 7, are commonly included. The declared methods of sample programs of clusters 3 and 6 are overlapping as soft clustering is employed in this study. Since this stage is before the recommended ranks are calculated, only the declared method names belonging to each cluster are listed.

Due to space constraints, clusters 0, 5, 8 through 16 are intentionally omitted. Many of the clusters consist of the same 16 declared methods as listed in clusters 3 and 6. In this implementation, if a declared method includes one or more invoked methods that comprise a maximal frequent itemset, then it is treated as an element of the cluster corresponding to that maximal frequent itemset. Therefore, the same set of methods appears in many clusters. The implemented condition seems to be most appropriate. However, if a user wants to reduce the number of elements belonging to each cluster, it can be easily implemented by setting the number of methods included in a maximal frequent itemset to two or more.

C. Calculation of Recommendation Ranking

1) Definition of tf-idf

The Term Frequency-Inverse Document Frequency *(tf-idf)* weight is a statistical measure that is commonly used in information retrieval [4]. In the context of our study, the *tf-idf* can be rephrased as follows:

Tf (term frequency) means the frequency of an invoked method name in a sample program,

Idf (inverse document frequency) indicates a numerical value used for measuring the importance of an invoked method name in a set of sample programs.

Among several options to calculate the *tf* and *idf*, we adopt the following definitions.

Tf_i is defined as the number of occurrences of an invoked method *i* in declared method.

Idf_i is defined as $log(N/DF_i)$, where *N* is the total number of declared methods that occur in a package of sample programs, and DF_i is the number of declared methods where an invoked method *i* appears at least once. It should be noted that idf_i of an invoked method *i* that appears in all declared methods is equal to $log(N/N)$, which is equal to 0.

2) Calculating Tf-idf for Sample Program Recommendation

As mentioned earlier, the maximal frequent itemset consists of a set of method names that suggest programming subjects. Examples of the maximal frequent itemset is displayed on the *JCombobox* in the GUI as shown in Figures 4 and 5. The proposed system identifies a set of declared methods related to the maximal frequent itemset when a user selects a cell on the *JCombobox*. Then, the proposed system starts to compute *tf* and *idf* for each of invoked methods that are defined in the set of declared methods.

Table III lists the *tf* and *idf* values of the invoked method names relating to the maximal frequent itemset *{getItemCount, JPanel, add, setPreferredSize, Dimension, setMaximumRowCount}* that is shown on the seventh line from the top in Figure 5. There are 35 invoked methods in the 16 declared methods in cluster 6 that concerns the maximal frequent itemset.

TABLE III. *TF* AND *IDF* VALUES FOR INVOKED METHOD NAMES

No.	Invoked method name	Tf	Idf
0	DefaultComboBoxModel	2	0.903
1	Dimension	14	0.058
2	Font	1	1.204
3	JButton	6	0.426
4	JButton.addActionListener	6	0.426
5	JButton.setFont	1	1.204
6	JComboBox	9	0.25
7	JComboBox.addActionListener	2	0.903
8	JComboBox.addItemListener	3	0.727
9	JComboBox.getItemCount	2	0.903
10	JComboBox.setEditable	3	0.727
11	JComboBox.setEnabled	1	1.204
12	JComboBox.setMaximumRowCount	3	0.727
13	JComboBox.setPreferredSize	14	0.058
14	JComboBox.setSelectedIndex	2	0.903
15	JComboBox.setSelectedItem	1	1.204
16	JFrame	2	0.903
17	JFrame.getContentPane	2	0.903
18	JFrame.pack	1	1.204
19	JFrame.setBounds	2	0.903
20	JFrame.setDefaultCloseOperation	2	0.903
21	JFrame.setTitle	1	1.204
22	JFrame.setVisible	2	0.903
23	JLabel	8	0.301
24	JPanel	16	0
25	JPanel.add	16	0
26	JTextField	4	0.602
27	add	16	0
28	getContentPane	14	0.058
29	setBounds	11	0.163
30	setDefaultCloseOperation	13	0.09
31	setLocation	2	0.903
32	setSize	2	0.903
33	setTitle	13	0.09
34	setVisible	13	0.09

Since the proposed system uses soft clustering based on a maximal frequent itemset, the method names that are included in the maximal frequent itemset should be considered to characterize the sample programs more strongly than the others. In this study, the weights of the invoked method names are adjusted using the following formula.

Let us *MFI* be the Maximal Frequent Itemset specified by a user and idf_{max} be the maximum of *idf* values.

$$\begin{aligned} Adjusted\ idf_j &= idf_j + idf_{max} && \text{if } j \in MFI \\ &= idf_j && \text{if } j \notin MFI \end{aligned} \quad (2)$$

Table IV shows the adjusted *idf* values for the maximal frequent itemset *{getItemCount, JPanel, add, setPreferredSize, Dimension, setMaximumRowCount}*. The

add methods are defined in both of the *JPanel* and *JFrame* classes. They are distinguished in the internal processing. Because the *add* method of the *JFrame* class is called via the *getContentPane* method, as shown in line 23 of Figure 7, it is simply denoted by *add*.

TABLE IV. ADJUSTED *IDF* VALUES

Invoked method name	Idf	Adjusted idf
Dimension	0.058	1.262
JComboBox.getItemCount	0.903	2.107
JComboBox.setMaximumRowCount	0.727	1.931
JComboBox.setPreferredSize	0.058	1.262
JPanel	0	1.204
JPanel.add	0	1.204
add	0	1.204

The degree of recommendation $DegR_i$ for a declared method i is calculated as:

$$DegR_i = \sum_{k=0}^{k=M-1} tf_{ik} * (Adjusted\ idf_k) \quad (3)$$

where tf_{ik} is the number of occurrences of the invoked method k in the declared method i, and idf_k is the inverse document frequency of the invoked method k.

Table V shows the degrees of recommendation for the declared method regarding the maximal frequent itemset *{getItemCount, JPanel, add, setPreferredSize, Dimension, setMaximumRowCount}*.

TABLE V. DEGREES OF RECOMMENDATION FOR SAMPLE PROGRAM

No.	DegR	Name of sample program
0	19.623	JComboBox_3::main(String[])
1	18.926	JD_ComboBox_7::JD_ComboBox_7()
2	18.038	JComboBox_1::JComboBox_1()
3	17.178	JD_ComboBox_2::JD_ComboBox_2()
4	15.967	JD_ComboBox_3::JD_ComboBox_3()
5	15.614	JD_ComboBox_8::JD_ComboBox_8()
6	14.028	JComboBox_2::JComboBox_2()
7	12.956	JD_ComboBox_6::JD_ComboBox_6()
8	12.216	HT_ComboBox_3::HT_ComboBox_3()
9	11.856	HT_ComboBox_1::main(String[])
10	11.576	JD_ComboBox_5::JD_ComboBox_5()
11	11.041	JComboBox_4::JComboBox_4()
12	10.916	JComboBox_5::JComboBox_5()
13	8.998	HT_ComboBox_2::HT_ComboBox_2()
14	7.781	JD_ComboBox_4::JD_ComboBox_4()
15	4.104	JD_ComboBox_1::JD_ComboBox_1()

The maximal degree of recommendation is normalized to be 1.000 and displayed in the text area of the GUI. For the lists in Table V, the normalized degrees of recommendation are obtained by dividing all the degrees by 19.623. This calculation generates the final list of recommendations shown in Figure 6.

V. EXPERIMENTAL RESULTS

This section describes two sets of experimental results. The first set of experimental results is about declared methods or sample programs relating to the *GUI.Combobox* package. Because GUI components in Java are typically embedded in a screen frame called *JFrame*, the declared methods for the *GUI.Combobox* package inevitably accompany *JFrame* APIs. Consequently, they are often complicated. The other set of experimental results concerns the *String_Handring* package. APIs for the *String* class tend to be called alone. Therefore, the declared methods for the *String_Handring* package are often concise.

A. Experiment on Programs in GUI.Combobox Package

Figure 14 shows the sample Java files included in the *GUI.Combobox* package that is listed on the 10th line from the bottom in Figure 2. The number of Java files is 18, and the number of declared method is 45 as shown in Table II.

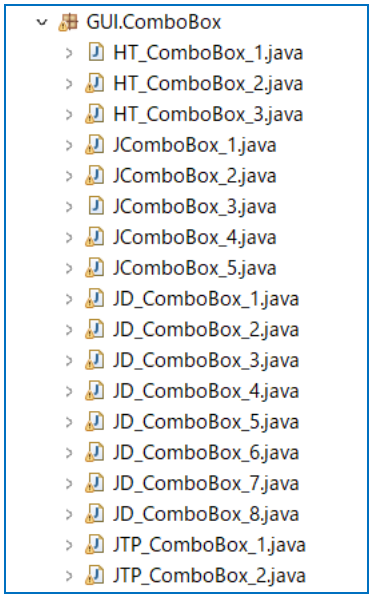


Figure 14. Sample Java files included in *GUI.Combobox* package.

Let us the programming subject be "*getItemCount JPanel add setPreferredSize Dimension setMaximumRowCount*" as listed on the seventh line from the top in Figure 5. The generated recommendation list is shown in Figure 6. Figure 7 shows the source program of the declared method named *JComboBox_3.java::main(String[])* with the normalized recommendation value of 1.000. This method has the top recommended rank because it contains all the invoked method names or APIs that make up the programming subject.

Figure 15 shows the declared method of eighth recommended rank with the normalized recommendation value of 0.660 named *JD_ComboBox6()*. This method fails to include two APIs of programming subjects, i.e., *getItemCount* and *setMaximumRowCount*. Instead, it includes APIs, such as *JTextField* and *JButton*.

```
22      JD_ComboBox_6(){
23            String[] combodata = {"Swing", "Java2D", "Java3D", "JavaMail"};
24            model = new DefaultComboBoxModel<String>(combodata);
25            JComboBox<String> combo = new JComboBox<String>(model);
26            combo.setPreferredSize(new Dimension(80, 30));
27            JPanel p = new JPanel();
28            p.add(combo);
29
30            JPanel controlPanel = new JPanel();
31            text = new JTextField(10);
32            button = new JButton("add");
33            button.addActionListener(this);
34            controlPanel.add(text);
35            controlPanel.add(button);
36
37            getContentPane().add(p, BorderLayout.CENTER);
38            getContentPane().add(controlPanel, BorderLayout.PAGE_END);
39
40            setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
41            setBounds(10, 10, 300, 200);
42            setTitle("JD_ComboBox_6");
43            setVisible(true);
44      }
```

Figure 15. Sample program with normalized recommendation value of 0.660.

Figure 16 shows the declared method named *JD_ComboBox_1()* that is ranked at the end of recommendation list with the normalized recommendation value of 0.209. The method is a basic program for the usage of *JComboBox* and its integration into *JFrame*.

```
19      JD_ComboBox_1(){
20          String[] combodata =
21              {"Swing", "Java2D", "Java3D", "JavaMail"};
22          JComboBox<String> combo = new JComboBox<String>(combodata);
23          JPanel p = new JPanel();
24          p.add(combo);
25
26          getContentPane().add(p, BorderLayout.CENTER);
27          setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
28          setBounds(10, 10, 300, 200);
29          setTitle("JD ComboBox 1");
30          setVisible(true);
31      }
32  }
```

Figure 16. Sample program with normalized recommendation value of 0.209.

Because of the recommended value calculation formula, a declared method that is closely related to a programming subject is ranked high. In general, a lower-ranked declared method is concise and better suited for beginners in learning programming because it contains fewer APIs or invoked methods. A declared method containing many APIs and less related to the programming subject tends to be ranked in the middle of the recommendation list.

B. Experiment on Programs in String_Handring Package

In Java programming language, the *String* class provides various APIs that can be used to handle string data. It includes APIs like *length*, *charAt*, *equals*, *indexOf*, *substring*, *toUpperCase*, *toLowerCase*, etc. These APIs facilitate string processing.

Figure 17 shows the sample Java files included in the *String_Handring* package that is located on the second line from the bottom in Figure 2. The number of Java files is 31, which is the same as the number of the declared method named *main* as shown in Table II. Since sample programs on the *String* class are rather simple, only one declared method is defined in each Java file.

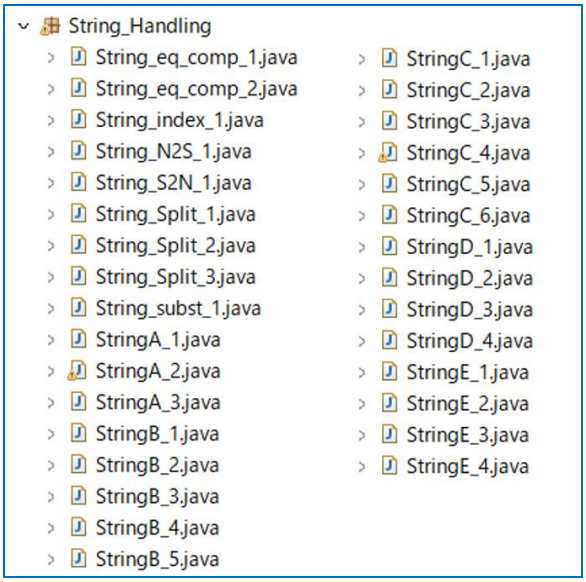


Figure 17. Sample Java files included in *String_Handring* package.

Figure 18 shows 15 identified programming subjects or clusters of the *String_Handring* package. Figure 18 reveals some commonly used APIs for string processing, such as *equals*, *indexOf*, and *substring*. Since this system uses soft clustering, there are APIs common to multiple clusters. For example, the *equals* API appears in four clusters, the *compareTo* API in three clusters as shown in Figure 18.

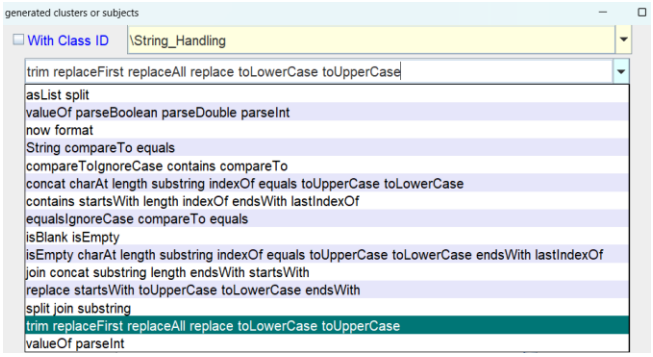


Figure 18. Identified programming subjects of *String_Handring* package.

Figure 19 shows a list of recommended declared methods for the programing subject "*trim replaceFirst replaceAll replace toLowerCase toUpperCase.*" The proposed system lists five declared methods with recommended values from 1.000 to 0.275.

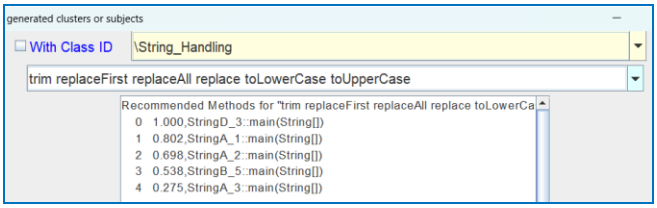


Figure 19. List of recommended declared methods.

Figure 20 shows the top-ranked sample program with the normalized recommendation value of 1.000. This sample program contains all the method names that constitute the programming subject.

```
package String_Handling;

public class StringD_3 {
    public static void main(String[] args) {
        String a = "Java 4 all ??";
        System.out.println(a.replace("?", "!"));            // Java 4 all !!
        System.out.println(a.replaceAll("[a-z]+", "_"));    // J_ 4 _ ??
        System.out.println(a.replaceFirst("[a-z]+", "_"));  // J_ 4 all ??
        System.out.println(a.toUpperCase());                // JAVA 4 ALL ??
        System.out.println(a.toLowerCase());                // java 4 all ??
        //  to remove whitespace from both ends of a string.
        String b = "       Java 4 Every Person !!    ";
        System.out.println(b.trim());              //Java 4 Every Person !!
    }
}
```

Figure 20. Sample program with normalized recommendation value of 1.000.

Figure 21 shows the sample program with a normalized recommendation value of 0.802. This sample program only contains two method names that constitute the programming subject, i.e., *toLowerCase* and *toUpperCase*. However, this program has a high recommended value because it contains many methods related to *String* class, such as *equlas*, *indexOf*, and *substring*.

```
package String_Handling;
//  http://www.btechsmartclass.com/java/java-string-handling.html
//  Java String Handling

public class StringA_1 {

    public static void main(String[] args) {
        String title = "Java Tutorials";
        String siteName = "www.btechsmartclass.com";
        System.out.println("Length of title: " + title.length());
        System.out.println("Char at index 3: " + title.charAt(3));
        System.out.println("Index of 'T': " + title.indexOf('T'));
        System.out.println("Last index of 'a': " + title.lastIndexOf('a'));
        System.out.println("Empty: " + title.isEmpty());
        System.out.println("Ends with '.com': " + siteName.endsWith(".com"));
        System.out.println("Equals: " + siteName.equals(title));
        System.out.println("Sub-string: " + siteName.substring(9, 14));
        System.out.println("Upper case: " + siteName.toUpperCase());
        System.out.println("Lower case: " + siteName.toLowerCase());
    }
}
```

Figure 21. Sample program with normalized recommendation value of 0.802.

Figure 22 shows the sample program with the normalized recommendation value of 0.275. This program only includes the *replace* method twice, causing to a low recommendation.

```
package String_Handling;

public class StringA_3 {
    public static void main(String[] args) {
        String s1="your name is java, isn't it?";
        String str=s1.replace('a','w'); //replaces all of 'a' to 'w'
        System.out.println(str);    // your nwme is jwvw, isn't it?

        str=s1.replace("is","was"); // replaces all of "is" to "was"
        System.out.println(str);    // your name was java, wasn't it?
    }
}
```

Figure 22. Sample program with normalized recommendation value of 0.275.

In the recommendation calculation proposed in this study, sample programs with fewer method types generally rank lower than those with richer in method types. However, the simpler program can be useful for beginners in programming because of its conciseness.

VI. Discussion

A. Syntax Analysis

In this study, the *Scanner* [21] class is used for parsing sample programs mainly because it reduces development effort. There are several options of parsing tools, including *JavaParser* [23] and *ANTLR* [24], both of which generate an Abstract Syntax Tree (AST). An AST is an intermediate representation of a source program represented by a tree structure. A few hundred lines of programming for traversing an AST allow an application to perform more complex operations than a mere method name extraction. ANTLR can parse formal languages including Java. All parsing tools work independent of IDEs and can parse sample programs stored in arbitrary directories.

B. Generative AI

ChatGPT is a chat-based generative AI released by OpenAI in Nov 2022 [25]. The *ChatGPT August 3* version allows users to get Java sample programs for *JComboBox* successfully. Since Java programs are generally characterized by APIs they call, *ChatGPT* precisely generates a report that contains a targeted sample program using a prompt including those APIs.

For example, the following prompt generates a report with a sample program that sets the number of elements to be displayed in a *JComboBox*'s dropped-down list using the *setMaximumRowCount* method:

> Would you show me a sample Java program about *JComboBox* using *setMaximumRowCount* method?

The proposed system and *ChatGPT* can be used to support each other. The proposed recommendation system automatically generates a list of APIs, which is helpful for writing prompts to *ChatGPT*.

For example, the list of APIs identified by the proposed system facilitates writing the following prompt:

> Would you show me a programming subject using the following Java APIs: "*getItemCount JPanel add setPreferredSize Dimension setMaximumRowCount*"?

Figure 23 shows a gist of *ChatGPT*'s response to this prompt. The response briefly states the subject and also suggests areas for further study.

> **Subject:** Creating a *JComboBox* within a *JPanel*
>
> **Objective:** Build a GUI application that contains a *JComboBox* inside a *JPanel*, allowing users to select options from the dropdown list. < *Intentionally omitted* >
>
> This subject will allow you to explore GUI customization and layout management in Java Swing while using the mentioned Java APIs to create a visually appealing and interactive user interface.

Figure 23. Gist of response from *ChatGPT*.

The proposed recommended system has a lot of potential to improve the programming learning environment by working together with generative AI models, such as *ChatGPT*, *Bing*, *Bard* and *Claude* [26].

VII. Conclusion and Future Work

This study deals with a recommendation system of Java sample programs using unsupervised machine learning. The proposed system soft clusters the sample program based on the set of invoked method names that are frequently observed. The clustering that corresponds to a programming subject is performed automatically using the *Apriori* algorithm. The recommended ranking of the sample programs in a cluster is calculated based on an adjusted *tf-idf* model that takes the method name and the number of times it is invoked.

This study is an extension of a previously published study [1]. The system described in this paper has been significantly enhanced in its functionality to perform source program parsing and soft clustering. Enhancements in parsing have made it possible to accurately parse complex sample programs, which allows the proposed system to handle sample programs on a variety of Java programming subjects. The functionality to optimize the value of *minSup*, i.e., a parameter of the *Apriori* algorithm, has been introduced to automatically perform optimal soft clustering.

It is confirmed through experiments using sample programs in the *File_IO*, *GUI*, and *String_Handring* packages, etc. that the sample programs containing APIs related to a programming subject are ranked high on a produced recommendation list. In addition, the set of APIs automatically identified by the proposed recommendation system is helpful for writing successful prompts for generative AI models including *ChatGPT*. The combination of the proposed system and the generative AIs offers significant potential to provide an unprecedented programming education environment.

Manual sample program acquisition from the Internet is time consuming and is a subject for future research. Additional experiments with larger number of sample programs are planned for a programming class room. Experiments in cooperation with generative AI are also planned.

References

[1] Y. Udagawa, “Lightweight Sample Code Recommendation System to Support Programming Education,” The Ninth International Conference on Advances and Trends in Software Engineering (SOFTENG 2023), IARIA, Apr. 2023, pp. 1-7, ISSN: 2519-8394, ISBN: 978-1-68558-042-1.

[2] M. P. Robillard, W. Maalej, R. J. Walker, and T. Zimmermann, “Recommendation systems for software engineering,” IEEE Software 27, pp. 80-86, Jul. 2010, DOI: 10.1109/MS.2009.161

[3] R. Agrawal and R. Srikant, “Mining sequential patterns,” The 11th IEEE International Conference on Data Engineering (ICDE), pp. 3-14, 1995, DOI: 10.1109/ICDE.1995.380415

[4] G. Sidorov. “Vector Space Model for Texts and the tf-idf Measure,” In Syntactic n-grams in Computational Linguistics, pp. 5-14, Apr. 2019, Springer, Cham, ISBN: 978-3-030-14770-9.

[5] M. Gasparic and A. Janes, “What Recommendation Systems for Software Engineering Recommend: A Systematic Literature Review,” Journal of Systems and Software 113, pp. 101-113, Mar. 2016, DOI: 10.1016/j.jss.2015.11.036

[6] H. Ko, S. Lee, Y. Park, and A. Choi, “A Survey of Recommendation Systems: Recommendation Models, Techniques, and Application Fields,” Electronics vol. 11, pp. 141-188, Jan. 2022, DOI: 10.3390/electronics11010141

[7] N. Katirtzis, T. Diamantopoulos, and C. Sutton, “Summarizing Software API Usage Examples Using Clustering Techniques,” The 21st International Conference on Fundamental Approaches to Software Engineering, vol. 10802, Springer, pp. 189-206, Apr. 2018, DOI: 10.1007/978-3-319-89363-1_11

[8] “Longest Common Subsequence,” Available from: https://www.tutorialspoint.com/design_and_analysis_of_algorithms/design_and_analysis_of_algorithms_longest_common_subsequence.htm

[9] C. Chen, X. Peng, B. Chen, J. Sun, Z. Xing, X. Wang, and W. Zhao, “More Than Deep Learning: Post-processing for API Sequence Recommendation,” Empirical Software Engineering. vol.27, pp. 1-32, Oct. 2021, Available from: https://ink.library.smu.edu.sg/sis_research/6580

[10] S.-K. Hsu and S.-J. Lin, “Mining Source Codes to Guide Software Development,” Asian Conference on Intelligent Information and Database Systems (ACIIDS 2010), pp. 445-454, Mar. 2010, DOI: 10.1007/978-3-642-12145-6_46

[11] J. Pei, J. Han, B. Mortazavi-Asl, H. Pinto, Q. Chen, U. Dayal, and M. Hsu, “Prefixspan: Mining Sequential Patterns Efficiently by Prefix-Projected Pattern Growth,” The 17th international conference on data engineering, pp. 215-224, Apr. 2001.

[12] Y. Chen, C. Gao, X. Ren, Y. Peng, X. Xia, and M. R. Lyu, “API Usage Recommendation Via Multi-View Heterogeneous Graph Representation Learning,” IEEE Transactions on Software Engineering, vol. 49, pp. 3289-3304, May 2023, DOI: 10.1109/TSE.2023.3252259

[13] T. Diamantopoulos and A. Symeonidis, “Mining Source Code for Component Reuse,” Mining Software Engineering Data for Software Reuse, Advanced Information and Knowledge Processing, Springer, pp. 133-174, Mar. 2020, DOI: 10.1007/978-3-030-30106-4_6

[14] “Levenshtein Distance,” Wikipedia, Nov. 2023, Available from: https://en.wikipedia.org/wiki/Levenshtein_distance

[15] A. Hora, “APISonar: Mining API usage examples,” Wiley Online Library, Software: Practice and Experience, vol. 51, issue 2, pp. 319-352, Oct. 2[4]020, DOI: 10.1002/spe.2906

[16] “Cosine Similarity,” Wikipedia, Oct. 2023, Available from: https://en.wikipedia.org/wiki/Cosine_similarity

[17] P. T. Nguyen, J. D. Rocco, C. D. Sipio, D. D. Ruscio, and M. D. Penta, “Recommending API Function Calls and Code Snippets to Support Software Development,” IEEE Transactions on Software Engineering, vol. 48, issue 7, pp. 2417-2438, Jul. 2022, DOI: 10.1109/TSE.2021.3059907

[18] A. Roy, “Introduction to Recommender Systems-1: Content-Based Filtering and Collaborative Filtering,” Jul. 29, 2020, Available from: https://towardsdatascience.com/introduction-to-recommender-systems-1-971bd274f421

[19] Eclipse foundation, “Download Eclipse Technology that is right for you,” Nov. 2023, Available from: https://www.eclipse.org/downloads/

[20] J. Rousu, “582364 Data mining, 4 cu Lecture 4: Finding frequent itemsets - concepts and algorithms,” University of Helsinki, Apr. 2010, Available from: https://www.cs.helsinki.fi/group/bioinfo/teaching/dami_s10/dami_lecture4.pdf

[21] Eclipse documentation "Interface IScanner," in org.eclipse.jdt.core.compiler, Dec. 2023, Available from: https://help.eclipse.org/latest/index.jsp?topic=%2Forg.eclipse.jdt.doc.isv%2Freference%2Fapi%2Forg%2Feclipse%2Fjdt%2Fcore%2Fcompiler%2Fpackage-summary.html

[22] "Christian Borgelt's Web Pages,", Nov. 2022, Available from: https://borgelt.net/fpgrowth.html

[23] JavaParser.org, "Tools for your Java code," 2019, Available from: https://javaparser.org

[24] T. Parr, "Download ANTLR", Sept. 2023, Available from: https:// www.antlr.org/download.html

[25] OpenAI, "Introducing ChatGPT," Nov. 2022, Available from: https://openai.com/blog/chatgpt

[26] J. Horsey "ChatGPT vs Bing vs Bard vs Claude comparison which ones right for you?" Aug. 2023, Available from: https://www.geeky-gadgets.com/chatgpt-vs-bing-vs-bard-vs-claude/