ICAS 2026

The Twenty-Second International Conference on Autonomic and Autonomous Systems

ISBN: 978-1-68558-356-9

March 8th –12th, 2026

Valencia, Spain

ICAS 2026 Editors

Roy Sterritt, School of Computing, Ulster University, Northern Ireland, UK

ICAS 2026

Foreword

The Twenty-Second International Conference on Autonomic and Autonomous Systems (ICAS 2026), held between March 8 - 12, 2026, was a multi-track event covering related topics on theory and practice on systems automation, autonomous systems and autonomic computing.

The main tracks referred to the general concepts of systems automation, and methodologies and techniques for designing, implementing and deploying autonomous systems. The next tracks developed around design and deployment of context-aware networks, services and applications, and the design and management of self-behavioral networks and services. We also considered monitoring, control, and management of autonomous self-aware and context-aware systems and topics dedicated to specific autonomous entities, namely, satellite systems, nomadic code systems, mobile networks, and robots. It has been recognized that modeling (in all forms this activity is known) is the fundamental for autonomous subsystems, as both managed and management entities must communicate and understand each other. Small-scale and large-scale virtualization and model-driven architecture, as well as management challenges in such architectures are considered. Autonomic features and autonomy requires a fundamental theory behind and solid control mechanisms. These topics gave credit to specific advanced practical and theoretical aspects that allow subsystem to expose complex behavior. We aimed to expose specific advancements on theory and tool in supporting advanced autonomous systems. Domain case studies (policy, mobility, survivability, privacy, etc.) and specific technology (wireless, wireline, optical, e-commerce, banking, etc.) case studies were targeted. A special track on mobile environments was indented to cover examples and aspects from mobile systems, networks, codes, and robotics.

Pervasive services and mobile computing are emerging as the next computing paradigm in which infrastructure and services are seamlessly available anywhere, anytime, and in any format. This move to a mobile and pervasive environment raises new opportunities and demands on the underlying systems. In particular, they need to be adaptive, self-adaptive, and context-aware.

Adaptive and self-management context-aware systems are difficult to create, they must be able to understand context information and dynamically change their behavior at runtime according to the context. Context information can include the user location, his preferences, his activities, the environmental conditions and the availability of computing and communication resources. Dynamic reconfiguration of the context-aware systems can generate inconsistencies as well as integrity problems, and combinatorial explosion of possible variants of these systems with a high degree of variability can introduce great complexity.

Traditionally, user interface design is a knowledge-intensive task complying with specific domains, yet being user friendly. Besides operational requirements, design recommendations refer to standards of the application domain or corporate guidelines.

Commonly, there is a set of general user interface guidelines; the challenge is due to a need for cross-team expertise. Required knowledge differs from one application domain to another, and the core knowledge is subject to constant changes and to individual perception and skills.

Passive approaches allow designers to initiate the search for information in a knowledge-database to make accessible the design information for designers during the design process. Active approaches, e.g., constraints and critics, have been also developed and tested. These mechanisms deliver information (critics) or restrict the design space (constraints) actively, according to the rules and guidelines. Active and passive approaches are usually combined to capture a useful user interface design.

We take here the opportunity to warmly thank all the members of the ICAS 2026 Technical Program Committee, as well as the numerous reviewers. The creation of such a high quality conference program would not have been possible without their involvement. We also kindly thank all the authors who dedicated much of their time and efforts to contribute to ICAS 2026. We truly believe that, thanks to all these efforts, the final conference program consisted of top quality contributions.

Also, this event could not have been a reality without the support of many individuals, organizations, and sponsors. We are grateful to the members of the ICAS 2026 organizing committee for their help in handling the logistics and for their work to make this professional meeting a success.

We hope that ICAS 2026 was a successful international forum for the exchange of ideas and results between academia and industry and for the promotion of progress in the fields of autonomic and autonomous systems.

We are convinced that the participants found the event useful and communications very open. We also hope that Valencia provided a pleasant environment during the conference and everyone saved some time for exploring this beautiful city.

ICAS 2026 Chairs:

ICAS 2026 Steering Committee

Roy Sterritt, Ulster University, UK
Mark J. Balas, Texas A&M University, USA
Karsten Böhm, Fachhochschule Kufstein, Austria
Jacques Malenfant, Sorbonne Université | LIP6 Lab, France
Claudius Stern, biozoom services GmbH - Kassel | FOM University of Applied Sciences – Essen, Germany
Philippe Martine, INRIA Sophia Antipolis, France

ICAS 2026 Publicity Chairs

Francisco Javier Díaz Blasco, Universitat Politècnica de València, Spain
Ali Ahmad, Universitat Politècnica de València, Spain
Sandra Viciano Tudela, Universitat Politecnica de Valencia, Spain
José Miguel Jiménez, Universitat Politecnica de Valencia, Spain

ICAS 2026

Committee

ICAS 2026 Steering Committee

Roy Sterritt, Ulster University, UK
Mark J. Balas, Texas A&M University, USA
Karsten Böhm, Fachhochschule Kufstein, Austria
Jacques Malenfant, Sorbonne Université | LIP6 Lab, France
Claudius Stern, biozoom services GmbH - Kassel | FOM University of Applied Sciences – Essen, Germany
Philippe Martine, INRIA Sophia Antipolis, France

ICAS 2026 Publicity Chairs

Francisco Javier Díaz Blasco, Universitat Politècnica de València, Spain
Ali Ahmad, Universitat Politècnica de València, Spain
Sandra Viciano Tudela, Universitat Politecnica de Valencia, Spain
José Miguel Jiménez, Universitat Politecnica de Valencia, Spain

ICAS 2026 Technical Program Committee

Faouzi Adjed, Institut de Recherche Technologique SystemX, France
Lounis Adouane, Université de Technologie de Compiègne (UTC), France
Waseem Akram, University of Calabria, Italy
Ghassan Al-Falouji, Kiel University, Germany
Getachew Alemu, Addis Ababa University, Ethiopia
Alba Amato, Institute for High-Performance Computing and Networking (ICAR), Napoli, Italy
Vijayan K. Asari, University of Dayton, USA
Mark Balas, Texas A&MUniversity, USA
Jagadish Bapanapally, University of Wyoming, USA
John S. Baras, University of Maryland, USA
Safya Belghith, National Engineering School of Tunis | University of Tunis El Manar, Tunisia
Noura Ben Moussa, National Engineering School of Gabes, Tunisia
Sylvain Bertrand, ONERA - The French Aerospace Lab - Centre de Palaiseau, France
Navneet Bhalla, University College London, UK
Estela Bicho, University of Minho / Centre Algoritmi / CAR group, Portugal
Karsten Böhm, Fachhochschule Kufstein, Austria
Assia Boukhamla, Badji Mokhtar University-Annaba, Algeria
Kamel Bouzgou, Université des sciences et de la technologie Oran USTO-MB, Algeria / Université Paris Saclay - Univ. Evry, France
Ziv Brand, Sami Shamoon College of Engineering, Israel
Estelle Bretagne, University of Picardie Jules Verne / lab MIS (modeling, information and systems), France
Ivan Buzurovic, Harvard University, Boston, USA
Alberto Castagna, Trinity College Dublin, Ireland
Rahul Chandan, Amazon Robotics, USA
Wen-Chung Chang, Taipei Tech, Taiwan

Miquel Kegeleirs, IRIDIA | Université Libre de Bruxelles, Belgium
Lial Khaluf, Independent Researcher, Germany
Hasan Ali Khattak, National University of Sciences and Technology (NUST), Islamabad, Pakistan
Igor Kotenko, SPIIRAS and ITMO University, Russia
Hak Keung Lam, King's College London, UK
Jisun Kim, Aston University, UK
Elhaouari Kobzili, National Polytechnic School, Algeria
Charles Lesire, ONERA/DTIS | University of Toulouse, France
Baoquan Li, Tiangong University, China
Guoyuan Li, Norwegian University of Science and Technology, Norway
Ji-Hong Li, Korea Institute of Robot and Convergence, Republic of Korea
Juncheng Li, East China Normal University, Shanghai, China
Sen Li, The Hong Kong University of Science and Technology, Hong Kong
Yangmin Li, The Hong Kong Polytechnic University, Hong Kong
Zhan Li, Swansea University, UK
Enjie Liu, University of Bedfordshire, UK
Eric Lucet, CEA | LIST | Interactive Robotics Laboratory, Palaiseau, France
Ren C. Luo, National Taiwan University, Taiwan
Nacer K M'Sirdi, Aix-Marseille Université, France
Jacques Malenfant, Sorbonne Université - CNRS, France
Giuseppe Carlo Marano, Politecnico di Torino, Italy
Guilhem Marcillaud, Paul Sabatier University, IRIT, France
Aurelian Marcu, Center for Advance Laser technology CETAL - National Institute for Laser Plasma and Radiation Physics, Romania
Raúl Marín Prades, Jaume-I University, Spain
Philippe Martinet, INRIA Sophia Antipolis, France
Enrique Mateos Melero, Universidad Carlos III de Madrid, Spain
Rajat Mehrotra, Teradata Inc., Santa Clara,, USA
René Meier, Hochschule Luzern, Switzerland
Jérôme Mendes, University of Coimbra, Portugal
Oliver Michler, Technical University Dresden, Germany
Sérgio Monteiro, Centro Algoritmi | University of Minho, Portugal
Naghmeh Moradpoor, Edinburgh Napier University, UK
Paulo Moura Oliveira, UTAD University, Vila Real, Portugal
Luca Muratore, Istituto Italiano di Tecnologia, Genova, Italy / The University of Manchester, UK
Taro Nakamura, Chuo University, Japan
Roberto Nardone, University of Reggio Calabria, Italy
Rafael Oliveira Vasconcelos, Federal University of Sergipe (UFS), Brazil
Flavio Oquendo, IRISA - University of South Brittany, France
Luigi Palmieri, Robert Bosch GmbH, Corporate Research, Germany
Eros Pasero, Politecnico di Turin, Italy
James Patten, University of Limerick, Ireland
Timothy Patten, University of Technology Sydney, Australia
Ling Pei, Shanghai Jiao Tong University, China
Damien Pellier, Université Grenoble Alpes, France
Van-Toan Pham, Vietnam Maritime University, Vietnam / National Taipei University of Technology, Taiwan
Agostino Poggi, DII - University of Parma, Italy

Radu-Emil Precup, Politehnica University of Timisoara, Romania
Md Muzakkir Quamar, King Fahd University of Petroleum and Minerals (KFUPM), KSA
Adrian Redder, Paderborn University, Germany
Leonīds Ribickis, Riga Technical University, Latvia
Douglas Rodrigues, Paulista University - UNIP, Brazil
Majid Roshanfar, Concordia University, Canada
Joerg Roth, Nuremberg Institute of Technology, Germany
Loris Roveda, Università della Svizzera italiana (USI), Switzerland
Spandan Roy, International Institute of Information Technology, Hyderabad, India
Fariba Sadri, Imperial College London, UK
Mohammad Safeea, Coimbra University, Portugal / ENSAM, Lille, France
Mehrdad Saif, University of Windsor, Canada
Lakhdar Sais, CNRS | Artois University, Lens, France
Michael A. Saliba, University of Malta, Malta
Ivan Samylovskiy, Lomonosov Moscow University, Russia
Ricardo Sanz, Universidad Politecnica de Madrid, Spain
Jagannathan Sarangapani, Missouri University of Science and Technology, USA
André Schneider de Oliveira, Federal University of Technology - Parana, Brazil
Ivan Serina, University of Brescia, Italy
Vesna Sesum-Cavic, TU Wien, Austria
Abdel-Nasser Sharkawy, South Valley University, Qena, Egypt
Inderjeet Singh, University of Texas at Arlington Research Institute (UTARI), USA
Edoardo Sinibaldi, Istituto Italiano di Tecnologia (IIT), Italy
J. Tanner Slagel, NASA Langley Research Center, USA
Mohammad Divband Soorati, University of Southampton, UK
Iderli Souza, Federal University of Pampa, Brazil
Bernd Steinbach, University of Mining and Technology, Freiberg, Germany
Claudius Stern, FOM University of Applied Sciences, Essen, Germany
Roy Sterritt, Ulster University, UK
Adrian-Mihail Stoica, National University of Science and Technology Politehnica Bucharest | Faculty of Aerospace Engineering, Romania
Yun-Hsuan Su, University of Washington, Seattle, USA
Saied Taheri, Virginia Tech, USA
Omar Tahri, PRISME | INSA Centre Val-de-Loire, France
Brahim Tamadazte, FEMTO-ST Institute / CNRS, France
Maryam Tebyani, University of California, Santa Cruz, USA
Francesco Tedesco, University of Calabria, Italy
Ravi Tejwani, MIT, USA
Giorgio Terracina, Università della Calabria, Italy
Carlos M. Travieso-González, Institute for Technological Development and Innovation in Communications (IDeTIC) | University of Las Palmas de Gran Canaria (ULPGC), Spain
Xuan-Tung Truong, Le Quy Don Technical University, Vietnam
Eddie Tunstel, Motiv Space Systems Inc. / Motiv Robotics, USA
Paulo Urbano, Universidade de Lisboa, Portugal
David Valiente García, Miguel Hernandez University, Spain
Vivek Shankar Varadharajan, École Polytechnique de Montréal, Canada
Ramon Vilanova, School of Engineering - UAB, Spain
Nikolaos (Nikos) Vitzilaios, University of South Carolina, USA

Copyright Information

Table of Contents

Extending MAPE-K with Data Augmentation to Mitigate Data Scarcity

Catherine Saunders, Roy Sterritt, Peter Nicholl, Ian McChesney
School of Computing and Mathematics
Ulster University
Belfast, Northern Ireland
email:{ce.saunders, r.sterritt, p.nicholl, ir.mcchesney}@ulster.ac.uk

***Abstract*—The deployment of autonomous swarms in remote or hazardous environments, such as space exploration, presents new challenges to data collection. The distance to earth necessitates full autonomous and autonomic operation without human direction. Autonomic Computing and the MAPE-K (Monitor, Analyse, Plan, Execute, with Knowledge) loop provide a control structure for self-management. An autonomic system processes internal and external data and uses it to make informed decisions. In the event that the data is scarce or missing, this reduces the reliability of the system. This paper addresses the issue of data scarcity by presenting and evaluating the effectiveness of using numeric interpolation to fill in missing data produced by a multi-robot swarm. The interpolated data is used to complete an existing dataset that is then passed through a data generation and evaluation pipeline. The results show that interpolation and generation can produce high quality synthetic data that could be used to mitigate data scarcity.**

Keywords-Autonomic Computing; mape-k loop; data generation; ctgan; synthetic data; data scarcity.

I. INTRODUCTION

In remote space exploration missions involving autonomous swarms, it is unrealistic for human operators to supervise every decision. The system must be capable of monitoring its own behaviour, analyse internal and external conditions, adjust in real-time to optimise performance and avoid collisions. Autonomic Computing's [1][2] MAPE-K (Monitor, Analyse, Plan, Execute, with Knowledge) [3][4] control loop consists of separate stages that make self-management possible. The output from each stage flows into the next; the quality of the data collected has a direct effect on the Analyse and Plan stages. The quality of adaptation depends on the quality of the data in the Knowledge Base. Datasets with missing or incomplete data make the analysis and planning stages less reliable.

In our previous work [5], we developed a swarm simulation consisting of two types of virtual robot and three direct-message collaboration protocols that varied the level of cohesiveness. These protocols enabled the swarm to communicate and collaborate on a searching task by sending messages requesting help. The sub-swarms were split into balanced (50%-50%), unbalanced (70%-30%), and extremely unbalanced (90%-10%). Throughout this paper, a combination of the operational parameters - sub-swarm split, signal reach, and type of collaboration protocol is collectively referred to as a swarm configuration. The simulation was inspired by future NASA concept missions such as the Autonomous Nano-Technology Swarm (ANTS) Prospecting Asteroid Mission (PAM) [6][7][8]. This mission would involve sending 1000 small craft to an asteroid belt, with each craft equipped with one of ten possible scientific instruments. A high failure rate and redundancy is expected due to the hazardous environment. This is one use case in which data scarcity could arise even in a system designed with a high degree of autonomy. Experiments showed that sub-swarm imbalance significantly affected performance and led to longer search times. The simulated tabular dataset was then used to test a data generation pipeline that could be embedded within the MAPE-K loop.

Testing physical swarms with all combinations of sub-swarm configuration could prove costly. This is especially true for remote space exploration missions involving swarms. It is not only expensive but also impractical and potentially hazardous to collect data that covers every possible cooperation protocol or sub-swarm split. In a real-world scenario involving a swarm of robots, the dataset collected may only include data for some swarm configurations while other configurations may have no data at all. Simulation could be used to fill the gaps, but if real data exists, then it is also possible to use the real data as a starting point to interpolate plausible values for untested configurations.

Enhancing the MAPE-K loop with data generation capabilities was explored in our previous work [9]. A Conditional Tabular Generative Adversarial Network (CTGAN) [10] [11] was used to increase the size of a dataset produced by a virtual swarm. We used simulation data to test a data generation pipeline embedded in the MAPE-K control loop. A CTGAN model was trained on the simulation output; the resulting synthetic data was evaluated to assess its quality and faithfulness to the original data. The evaluation results showed that a CTGAN can be used to increase the sample size and produce high-quality data. This paper extends the previous work by adding an additional capability that could be used to fill in missing configurations before applying data generation.

This work extends the previous data generation pipeline by adding interpolation of missing swarm configurations to the MAPE-K. We focus on solving the issue of data scarcity, specifically when no data exists for sub-swarm splits that have not been deployed. We reuse our validated simulation dataset to test interpolation before applying the data generation step. We use the numeric data for the existing sub-swarm splits and interpolate new data for non-existent splits 60-40 and 80-20. Two numeric-only interpolation approaches are used: direct row-scaling (Approach A), and means-based (Approach B).

The results are evaluated using an evaluation suite comprising of visual and statistical techniques. These include boxplots, kernel density estimates, binned distributions, principal component analysis, and Kolmogorov-Smirnov tests. The best performing interpolation method is then used to create training data for the new 60-40 and 80-20 splits. The CTGAN synthetic data is then compared to the interpolated dataset. We assess whether an interpolate then generate process can provide a practical way to reason untested sub-swarm configurations using only data that exists.

In Section II, we provide background information on the research area; Section III discusses the interpolation techniques; in Section IV, we present the experiments and results.

II. Background and related work

Autonomic computing was proposed as a way to manage large, complex systems without relying on human intervention [12][13]. The MAPE-K loop was devised so that individual components within a system could self-manage and be scalable. The modular structure of MAPE-K means it can be enhanced with new stages or capabilities. Using deep learning techniques within the MAPE-K is a relatively new research area. Most work looks at integrating a Large Language Model (LLM) to provide more advanced semantic reasoning and planning capabilities. Mitigation of data scarcity is usually discussed with the goal of producing more data for training an LLM, not increasing the size of collected data [14][15]. In [16], an LLM is added to the MAPE-K to improve multi-agent cooperation. In [17], they explore ways LLMs could be integrated into self-adaptive systems to monitor data and generate adaptation plans. They suggest that embedding an LLM into the Analysis and Planning phases of the MAPE-K would benefit performance. In [18], they propose enhancing the MAPE-K feedback loop with Generative AI in the form of a LLM. They suggest modifying the MAPE-K control loop by adding a GPT that performs the reasoning tasks.

However, the idea of adding data generation to MAPE-K to mitigate real-time data scarcity remains underexplored. CTGAN models can be used to augment small tabular datasets by learning the structure and relationships with the data and producing additional synthetic rows. This concept is increasingly promoted as a method for addressing issues such as limited sample sizes, privacy restrictions, or the cost associated with collecting new data. CTGAN generated data has also been used to produce training data for training LLMs when real data is scarce or sensitive [15]. A CTGAN can learn the distribution of the training data and generate new rows that preserve the statistical properties and relationships between features. In [9], we showed that a CTGAN trained on swarm simulation output can produce plausible synthetic data.

This paper builds on our previous work and proposes a method for creating unseen data from known distributions. This extends our data generation pipeline proposed in [9] by adding a precursor step that fills in gaps in a dataset before it is passed to a CTGAN to increase sample size. This work presents a modification to the MAPE-K so that it can make use of Generative AI to solve the problem of data scarcity.

III. Interpolation in the MAPE-K Loop

This section describes the two numeric interpolation approaches used for creating new data for untested swarm configurations. The two approaches are – Approach A: Direct Row Scaling, and Approach B: Means Interpolation.

The simulation produced output for a heterogeneous swarm consisting of two types of robot. The swarm was configured to be equally balanced, unequally balanced (70% one type, 30% the other) and extremely unbalanced (90%-10%). The interpolation techniques were used to interpolate the new swarm configurations 60-40 and 80-20. The interpolated data was then used to train a CTGAN. In our previous work [9], 20 CTGAN models were trained with varying epochs (100-2000) and batch sizes (50-500). Higher epochs combined with lower batch sizes produced the best results. For this work, 2000 epochs were used with a batch size of 500 to reduce training time. The previous work [9] showed that the 2000-epoch and 500-batch model performed well with only a marginal difference from the top performing model of 2000-epoch and 50-batch.

A visual and statistical analysis of the numeric values within these augmented datasets was performed. It compares the new interpolated swarm splits of both approaches and whether they fit logically within the distributions of the original simulated data.

Linear interpolation was used to fill in the gaps between adjacent known configurations. It was applied between two neighbouring configurations one at a time, making a linear estimate between two known anchor points a logical approach. Other methods such as Multiple Imputation by Chained Equations (MICE) are suitable for filling in missing fields within existing records and would not be suitable for generating entire rows for unseen configurations. Future work could explore comparing numeric interpolation to generative approaches such as Large Language Models (LLMs) or Generative Adversarial Networks (GANs).

For Approach A, we generated new samples for 60-40 and 80-20 splits by scaling numeric rows from the nearest known configuration (50-50 for 60-40, 90-10 for 80-20). The rows from the 50-50 split were multiplied by a scaling factor. The majority value of the interpolated split (60) was divided by the majority value of the reference split (50). The scaling factor was calculated by dividing 60 by 50 to get 1.2.

For Approach B, Scipy's 'interp1d' function was used to perform interpolation from the mean of the existing splits. Random Gaussian noise (10% of the interpolated value) was added to each generated row to introduce variability around the interpolated means. To interpolate the new 60-40 data, anchor reference points were created from the 50-50 and 70-30 data. To calculate the 80-20 split, the 70-30 and 90-10 splits were used as anchors. Linear interpolation was used to calculate new values from the mean of these anchors.

We show that interpolation could be added to the MAPE-K to enhance a dataset with missing data. An Autonomic Manager (AM) could interpolate uncollected swarm configuration data as a method to mitigate data scarcity. We show that non-existent data can be interpolated and proven to be logically consistent with known configurations. Adding

 ISBN: 978-1-68558-356-9

interpolation to the MAPE-K would provide an AM with more data to work from when performing analyses. To evaluate the accuracy of the two approaches, the following evaluation tests were used – Principal Component Analysis (PCA), Kolmogorov-Smirnov (KS), boxplots, and histogram comparison. To assess the utility of the interpolated data, we trained a CTGAN and evaluated the synthetic data against the interpolated data.

IV. EXPERIMENT RESULTS

The original simulation dataset was interpolated and the data from the best interpolation approach was used to train a CTGAN to produce synthetic data. Section A describes the evaluation of the interpolated data, and Section B describes the evaluation of the interpolated-synthetic data. The evaluation uses a mixture of visual analysis (boxplots, KDE plots, PCA) and statistical tests (KS, binned distributions, min-max comparison) to assess the interpolated and synthetic data from multiple perspectives.

A. *Interpolation Evaluation Metrics*

To evaluate the interpolated data, we assessed whether each approach preserves the numeric structure of the original dataset by using visual and statistical analysis.

- ***Boxplots***

A boxplot comparison was used to compare the new interpolated data's numeric feature 'Simulation Time' against the original simulation's running times for each swarm configuration. The 'Simulation Time' is the length of the time the simulation ran before all items were found and analysed by the swarm. Figure 1 shows the boxplots for Simulation Time (Y-axis); the X-axis gives the swarm composition split – original and interpolated splits. The new interpolated splits derived from Approaches A and B are displayed side by side in the X-axis '60-40' and '80-20' categories. Approach A displays more variability in the data compared to the compressed narrow range of Approach B boxplots. The results show that Approach A's technique produces data that is more consistent with the original data. It replicates the original data's tendency to exhibit more range variation as the sub-swarm configuration becomes more unequal.

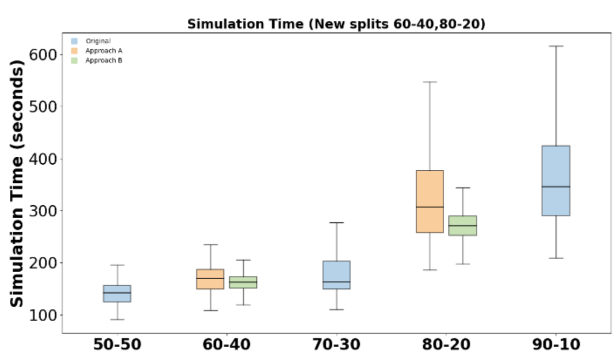


Figure 1. Simulation Time Boxplots showing new sub-swarm configurations compared to sub-swarm data in original dataset.

- ***KDE Plots***

A Kernel Density Estimation (KDE) plot was used to compare the distribution of the 'Simulation Time' feature. The KDE plot peaks represent where the data is most concentrated, the higher peaks the more data points are grouped in that range. In Figure 2, the original dataset is shown in blue; Approach A is orange;, Approach B is in green. The X-axis gives the simulation runtime in seconds, the Y-axis shows the data density.

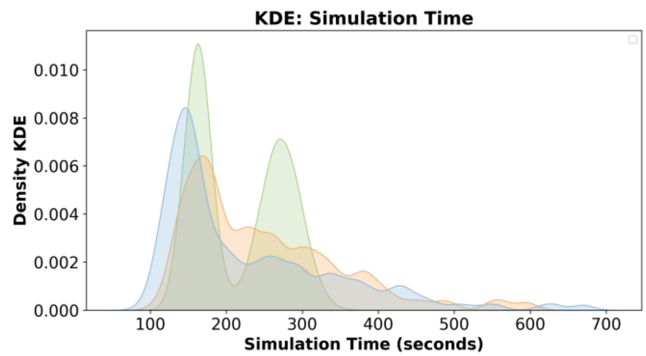


Figure 2. Simulation Time KDE plot, Approach A shown in orange, Approach B in green.

It was expected that the interpolated data should logically fall inside the extremes of the original data's curve. Approach A follows the shape of the original data's distribution closely. It consists of a high density peak at lower X-axis times but not as low as the original data which includes the fastest 50-50 configuration. Whereas Approach B shows that the new data is concentrated in two areas represented by high density peaks. It has concentrated the data in narrower time ranges and hasn't captured the general pattern of the original data.

- ***Binned Distribution Analysis***

The simulation time data was divided into 20 equally sized bins, with each bin representing a time interval of approximately 29 seconds. The choice of 20 bins was calculated using Sturges' Rule [19], the equation $k = [\log_2(n) + 1]$ gives the optimal number of bins for a histogram. The binned analysis shown in Figure 3 compares three sources of varying sample sizes. The original dataset contained n = 43,061 rows. The interpolated datasets each contained n = 20,000 rows. Sturges' rule suggests 16-17 bins for these sample sizes. A value of 20 was selected as it provides more granularity across the simulation time range (~90-672 seconds). This resulted in a bin width of approximately 29 seconds, which allowed for more information within each bin and reduced the noise.

We calculated the percentage of data that falls into each bin. Figure 3 shows the spread of the data across the bins and whether the approaches over or under concentrate values in certain bins. Approach A (orange) has values distributed across almost all bins. By contrast, Approach B (green) data is clustered in the first ten bins. No data exists in the higher

 ISBN: 978-1-68558-356-9

bins, indicating that the interpolation approach has produced a narrower spread of values. Approach A closely mirrors the original distribution. However, Approach B concentrates the data in a smaller number of bins. These results are consistent with the clustering behaviour observed in the KDE plots.

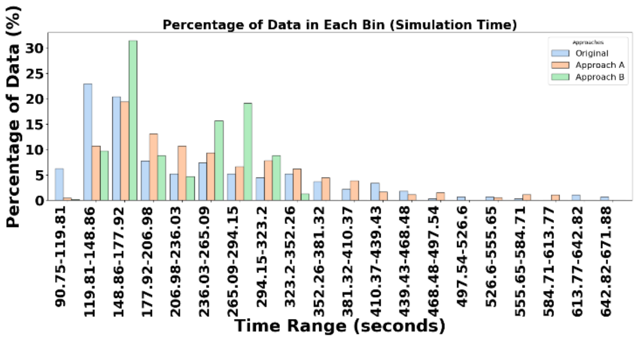


Figure 3. Distribution percentages of Original data and both Approaches.

- ***Min Max Values***

This test compared the *'Simulation Time'* feature's minimum, maximum, and mean values for all datasets. The results for both approaches are shown in Table I. The original dataset running times are provided for comparison. In the original data, the mean Simulation time increases as the sub-swarm becomes more unbalanced: a plausible 60-40 value would lie between 50-50 and 70-30; an 80-20 value should lie between 70-30 and 90-10.

The results show that Approach A's maximum values more closely capture the trend seen in the original data, whereas Approach B is contained within a smaller range. Approach B shows much narrower ranges and lower maximum values; the 60-40 maximum value is lower than the 50-50 maximum running time. This reinforces the KDE and histogram results that indicate that Approach B produces data clustered around the central anchor points and fails to capture the variability that exists in the real data. The standard deviation values in Table I also reflect this: Approach B has much lower standard deviations (16.25 and 27.43) than Approach A (36.81 and 92.07) and the original data (30.71 to 103.09), indicating that Approach B produces data with too little spread.

TABLE I. MIN, MAX, AND MEAN VALUES

Source	split	n	mean	std	min	max
App. A	60-40	10,000	175.18	36.81	108.9	315.32
	80-20	10,000	323.89	92.07	185.85	597.23
App. B	60-40	10,000	162.29	16.25	91.55	220.71
	80-20	10,000	271.5	27.43	172.73	378.76
Original	50-50	14,336	146.15	30.71	90.75	262.77
	70-30	14,376	178.19	43.81	109.95	330.64
	90-10	14,349	363.92	103.09	209.08	671.88

Approach A maintains the overall distribution structure and creates data that is more realistic and consistent with the data produced from the swarm simulation. For integration into the MAPE-K loop, this ability to maintain realistic minimum, mean, and maximum values is essential to creating high quality synthetic data. If the interpolated data used for CTGAN training has an artificially narrow range, the synthetic data will also be constrained. Approach A avoids this by preserving the full range of values seen in the original simulation.

- ***PCA***

PCA was used to examine the most important patterns to see if the two approaches look similar to the original. In Figure 4, each data point on the PCA scatter plot represents one row of the dataset. The dataset is reduced from 6 columns/features to 2 principal Components.

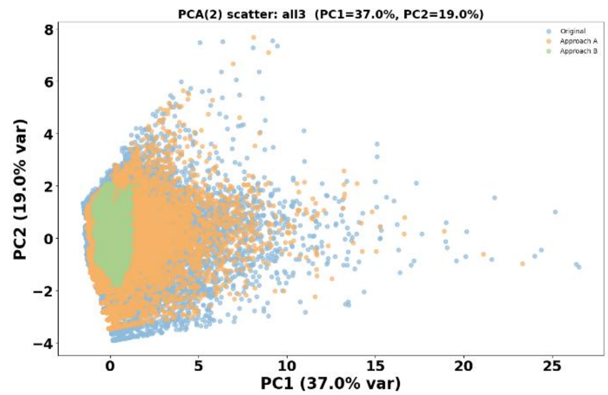


Figure 4. PCA comparison of the Original dataset, and interpolated Approach A and B datasets.

The dataset was standardized using Scikit-learn's 'StandardScaler', this scaled the features so that they were between 0 and 1. Scikit-learn's PCA function was used to transform the data into a 2D coordinate space. In Figure 4, the data point cloud shows considerable overlap between Approach A and the original data. It has accurately reproduced the structure and patterns of the original data. The green cloud represents Approach B, its data points are concentrated in a small area indicating that it has failed to capture the range and variability of the data.

- ***KS Test***

A Kolmogorov-Smirnov (KS) test was used to check that both interpolated approaches produce data that is statistically different from the original data's swarm configurations. This test measures the similarity between two distributions. It was expected that both approaches would be statistically significantly different. The test compared the 'Simulation Time' feature from each interpolated dataset to the original dataset. The original dataset was compared to the new interpolated sub-swarm split data (60-40, and 80-20).

The KS results shown in Table II confirm that both approaches produced data that is different from the original dataset. A lower KS statistic indicates a greater similarity. A result of 0.248 for Approach A confirms that the new dataset is clearly different from the original. The *p-values* for both approaches show that they are statistically significantly

 ISBN: 978-1-68558-356-9

different from the original simulation dataset. For the *p-value*, a result closer to 1 indicates that there is insufficient evidence to conclude that the datasets differ, whereas a value close to zero provides strong evidence that the distributions differ. As the *p-value* result for both approaches is $p < 0.001$, we have clear evidence that the interpolated datasets are different from the existing configurations in the original simulation dataset. The KS test confirmed that both interpolated datasets were statistically different from the original configurations, as expected.

TABLE II. KS RESULTS FOR INTERPOLATED VS ORIGINAL (SIMULATION TIME FEATURE)

Comparison	n (Orig.)	n (new)	KS Statistic	P-value	Result
Orig. vs A	43,061	20,000	0.2482	< 0.001	Differ
Orig. vs B	43,061	20,000	0.2171	< 0.001	Differ

B. CTGAN Evaluation Metrics

Based on the evaluation results in the previous section, Approach A was selected for further CTGAN training. The visual analysis, binned distributions, and min-max results demonstrated that Approach A better preserved the shape and variability of the original data. The KS statistic was used to check that the data was different from the original but it cannot determine which approach produces more realistic data, as it only measures difference not quality.

The interpolated dataset from Approach A was used to train a CTGAN model to produce further synthetic data. In our previous work [9], we developed a suite of metrics to evaluate the quality of CTGAN synthetic data. To evaluate the interpolated synthetic data, we applied several of the metrics from the previous work. The purpose was to assess whether interpolated data can be used as the training data for a CTGAN model.

The Approach A dataset contained 20,000 rows in total; this equated to 10,000 rows for each new scenario (60-40, 80-20). This was reduced to 2,341 rows (60-40) and 3,074 rows (80-20) so that the CTGAN was trained on a subset of the data. The subset consisted of only data for the lowest communication range. The simulation output data consisted of three communication ranges (low, medium, max).

The experiment involved training two CTGAN models – one per new swarm split. The interpolated data for the missing configurations was separated by Swarm Split (60-40 and 80-20) to allow the CTGAN to better learn the nuances of each configuration. The training process involved training for 2000 epochs with a batch size of 500. These training parameters were chosen based on the previous work [9], which showed that the 2000-epoch, 500-batch model provides an acceptable trade-off between performance and training times.

- ***Statistical Similarity (KS Test)***

This test compares the CTGAN synthesized data for 80-20 and 60-40 to the held-out interpolated test data's 80-20 and 60-40 values. It compares the two datasets and assesses their numerical distribution similarity for the Simulation Time feature.

The KS test results in Table III show that the synthetic data is similar to the held-out interpolated test data as expected. The interpolated test data vs synthetic data distribution comparison results (0.08-0.101) in Table III are much smaller than the interpolated KS results (0.217-0.248) from Table II. The results are closer to 0 than before as the same configuration is now being compared as opposed to new vs existing splits.

TABLE III. KS RESULTS FOR INTERPOLATED VS SYNTHETIC

Model	Epochs	Batch Size	KS Statistic	KS Complement
60-40	2000	500	0.101408	0.89859
80-20	2000	500	0.080510	0.91948

- ***KDE Plots***

The KDE plots show how well the numeric shape was reproduced in the synthetic data. For both splits, the general shape of the data aligns well with the training data's KDE. This confirms that the CTGAN Generator network has learned the overall shape of the distribution of the interpolated dataset.

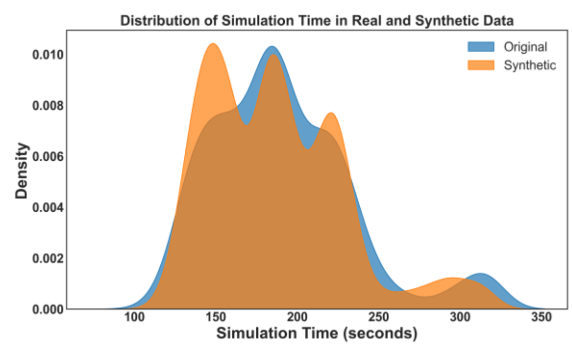


Figure 5. KDE of new 60-40 synthetic data compared to interpolated data.

Figure 5 shows the 60-40 plots; the orange KDE curve represents the synthetic data. It displays the same central peak as the original (blue) training data (interpolated dataset); however the synthetic data has added two unnecessary peaks. The new peaks do not significantly affect the overall shape, which remains similar to the Simulation Time curve.

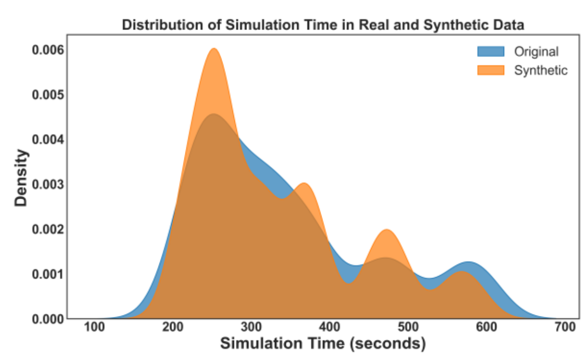


Figure 6. KDE of new 80-20 synthetic data compared to interpolated data.

Copyright (c) IARIA, 2026. ISBN: 978-1-68558-356-9

Figure 6 gives the results of the 80-20 comparison to the interpolated training dataset. The KDE lines overlap and follow the same general shape as the interpolated dataset; this shows that the CTGAN has learned the distribution shape.

V. Conclusion and Future Work

This work presented a comparison of two interpolation approaches for filling in the gaps in a simulated swarm output dataset. Approach A (Direct Row Scaling) outperformed Approach B (Means-based Interpolation). Noise was added to increase Approach B's spread; however, this still resulted in the data being confined to a narrow range around the mean.

Approach A preserved the shape and distribution of the data more accurately. The PCA analysis showed that Approach A overlapped well with the original dataset. The cloud pattern fits logically within the bounds of the original dataset. In data-scarce environments, having approximated but plausible numeric data for missing swarm configurations could prove useful. For example, an Autonomic Manager could use the data to make more informed decisions during the Analyse and Plan phases of the MAPE-K loop.

The CTGAN models trained on interpolated 60-40 and 80-20 configurations produced good quality synthetic data in terms of numerical distribution. The KS test shows a good match between real and synthetic data with regards to the Simulation Time feature. The KDE and histogram results demonstrated that the Generator was able to learn the distribution of the data.

These findings confirm that numeric interpolation, followed by generative modelling, is a viable technique for augmenting data where real world examples are missing. By combining row scaling and a CTGAN, the Autonomic Manager can simulate plausible data for previously untested swarm configurations. This would enable better analysis and planning under data-scarce conditions.

Future work could investigate the impact of the synthetic data on the MAPE-K loop's planning performance. This would involve assessing whether an Autonomic Manager performs better when trained on interpolated and synthetic data than an AM operating with incomplete data. As this work is based on a single simulated dataset and uses linear interpolation, further work could explore validating the approach across real-world swarm datasets. Deploying the interpolation and data generation approach within a live MAPE-K loop would allow for assessment of its impact on the Plan stage, providing evidence of practical benefit.

References

[1] P. Horn, "Autonomic Computing: IBM's Perspective on the State of Information Technology," in *AGENDA 2001*, Scottsdale, AZ., USA, 2001.

[2] IBM, "An Architectural Blueprint for Autonomic Computing," IBM, 2003.

[3] M. C. Huebscher and J. A. McCann, "A Survey of Autonomic Computing—Degrees, Models, and Applications," *ACM Comput. Surv.*, vol. 40, no. 3, pp. 1–28, Aug. 2008, doi: 10.1145/1380584.1380585.

[4] J. O. Kephart and D. M. Chess, "The vision of autonomic computing," *Computer (Long Beach, Calif.)*, vol. 36, no. 1, pp. 41–50, 2003.

[5] C. Saunders, R. Sterritt, and G. Wilkie, "Collective communication strategies for space exploration," *JBIS - J. Br. Interplanet. Soc.*, vol. 72, no. 12, pp. 416–430, 2019.

[6] P.E. Clark *et al.*, "PAM: Biologically Inspired Engineering and Exploration System Mission Concept, Components, and Requirements for an Asteroid Belt Population Survey," in *55th International Aeronautical Congress*, Vancouver, Oct. 2004, Paper No. IAC-04-Q5.07. doi: 10.2514/6.IAC-04-Q.5.07.

[7] E. Vassev, R. Sterritt, C. Rouff, and M. Hinchey, "Swarm Technology at NASA: Building Resilient Systems," *IEEE IT Pro*, vol. 14, no. 2, pp. 36–42, 2012.

[8] R. Sterritt and M. Hinchey, "Engineering Ultimate Self-Protection in Autonomic Agents for Space Exploration Missions," in *Proceedings of the 12th IEEE International Conference and Workshops on the Engineering of Computer-Based Systems*, IEEE, 2005, pp. 506–511. doi: 10.1109/ECBS.2005.36.

[9] C. Saunders, R. Sterritt, P. Nicholl, and I. McChesney, "Synthetic Data Generation for Autonomic Computing," in *ICAS 2025, The Twenty-First International Conference on Autonomic and Autonomous Systems*, Lisbon, Portugal, 2025, pp. 1–7.

[10] I. Goodfellow *et al.*, "Generative adversarial networks," *Commun. ACM*, vol. 63, no. 11, pp. 139–144, Oct. 2020, doi: 10.1145/3422622.

[11] L. Xu, M. Skoularidou, A. Cuesta-Infante, and K. Veeramachaneni, "Modeling tabular data using conditional GAN," *Adv. Neural Inf. Process. Syst.*, vol. 32, no. NeurIPS, 2019.

[12] R. Sterritt, M. Parashar, H. Tianfield, and R. Unland, "A Concise Introduction to Autonomic Computing," in *Advanced Engineering Informatics*, Jul. 2005, pp. 181–187. doi: 10.1016/j.aei.2005.05.012.

[13] A. G. Ganek and T. A. Corbi, "The Dawning of the Autonomic Computing Era," *IBM Syst. J.*, vol. 42, no. 1, pp. 5–18, 2003.

[14] A. Vaswani *et al.*, "Attention Is All You Need," in *Proceedings of the Advances in Neural Information Processing Systems*, vol. 30, Long Beach, CA, USA, Dec. 2017, pp. 5998–6008.

[15] C. Del Gobbo, "A Comparative Study of Open-Source Libraries for Synthetic Tabular Data Generation: SDV vs. SynthCity," arXiv preprint arXiv:2506.17847, Jun. 2025.

[16] C. Zhang *et al.*, "ProAgent: Building Proactive Cooperative Agents with Large Language Models," *Proc. AAAI Conf. Artif. Intell.*, vol. 38, no. 16, pp. 17591–17599, Mar. 2024, doi: 10.1609/aaai.v38i16.29710.

[17] J. Li *et al.*, "Exploring the Potential of Large Language Models in Self-adaptive Systems," *Proc. 19th Symp. Softw. Eng. Adapt. Self-Managing Syst. SEAMS 2024, pp. 77–83, 2024, doi: 10.1145/3643915.3644088.*

[18] N. Nascimento, P. Alencar, and D. Cowan, "Self-Adaptive Large Language Model (LLM)-Based Multiagent Systems," in *2023 IEEE International Conference on Autonomic Computing and Self-Organizing Systems Companion (ACSOS-C)*, IEEE, Sep. 2023, pp. 104–109. doi: 10.1109/ACSOS-C58168.2023.00048.

[19] H. A. Sturges, "The Choice of a Class Interval," *J. Am. Stat. Assoc.*, vol. 21, no. 153, pp. 65–66, Feb. 1926. [Online]. Available: http://www.jstor.org/stable/2965501.

Supervising Quality Environments with an Autonomic Ledger (SQuEAL)

Joel R. Bennett, Roy Sterritt
School of Computing, Ulster University, Belfast, N. Ireland
email: pixce@outlook.com | r.sterritt@ulster.ac.uk

Abstract— **Quality manufacturing environments must meet stringent regulatory requirements and implement an overarching Quality Management System (QMS), governing Good Manufacturing Practice (GxP), which engages in a process of Continuous Improvement (CI). This normally involves internal and external audits of the various quality systems, or when issues arise, identifying quality gaps in said systems and processes, followed by the implementation of correction(s). The QMS comprises a system of systems and can become extremely complex, requiring a high level of human effort, with multiple concurrent schedules, including system lifecycle, documentation and review. QMS supervises areas where products are held and includes logistical concerns, computerised systems, instrumentation requiring regular calibration, internal and external audits, inter-company certification of customer and vendor facilities, regulatory certification of facilities and products, printing and so on. The addition of human effort to attempt correction of human effort inevitably introduces more complexity and potential for new errors, even as improvements are being addressed. The problem of systems becoming more complex and difficult to manage is one directly addressed by the paradigm of Autonomic Computing (AC). The problem being addressed here is how we may take an autonomic computing approach to detecting quality gaps and corroborating quality systems integrity. This paper will consider the potential role that AC can play in improving QMS by augmenting the substantial human effort and corroborating the sequence of events through the implementation of a secure time-series ledger.**

Keywords- Autonomic Computing; GxP; QMS; quality; blockchain; ledger.

I. INTRODUCTION

This paper reports on research that considered the potential of a beneficial parallel relationship between the Quality Management System (QMS) and the paradigm of Autonomic Computing (AC) together with a proof of concept, which demonstrates how these benefits may be realised. This will specifically provide a corroborative time-series blockchain based ledger with autonomic features hereafter called simply an Autonomic Ledger (AL). Some non-exhaustive background information from these paradigms will be considered as briefly as possible, to provide an understanding of how a relationship between them might be described.

Many manufacturers work within a framework of regulation and guidance, which is collectively described as Good Manufacturing Practice (GMP). These practices extend to activities which support manufacturing, such as Good Laboratory Practice (GLP), Good Documentation Practice (GDP) and areas otherwise encapsulated by the term "GxP", or Good "anything" Practice [1].

The QMS is at its core a system of documentation which exists in some form within GxP environments and is designed to ensure manufacturers and the roles within them adhere to quality criteria, throughout the production lifecycle. We can refer to the ISO 9000 Quality Management System, which is a widely used set of global quality standards, which are primarily concerned with the quality of the documentation system that underlines the product [2]. ISO9001 is an original subset of ISO9000, which includes design, development, production, installation and servicing within the QMS cycle. The latest revision is ISO9001:2015 [3, pp. 6]. The essential form of ISO9001 standards is a set of documentation controls imposed upon an organisation to ensure quality permeates the planning and execution of all steps in any process. CI is something that should result from a properly functioning QMS according to ISO9001.

Automated and computerized systems have become a significant presence within the QMS, intended to increase product quality and enhance record keeping. Yet, they also introduced additional QMS documentation requirements. The specific go to standards document relating to automated and computerized systems is known as Good Automated Manufacturing Practice (GAMP) [4, pp. 4]. A proposed QMS approach to managing rising requirements of electronic records keeping, is the expectations of risk to quality and safety, versus "validate everything" [6]. This is precisely the approach now taken by GAMP revision 5, which is sub-titled "A risk based approach" [7].

AC is a paradigm that arose from recognition by individuals within IBM, of the inevitable rise of computing system complexity, which was coming, as far back as the 1990's [8]. It is inspired by observations within biological systems, in this case the unconscious Autonomic Nervous System (ANS) which is comprised of 2 sub-systems, the sympathetic (SyNS) influencing a heightened "fight or flight" response in the host and parasympathetic (PaNS) which provides a calming influence toward a "rest and digest" state [9]. The achievement of similar self-managing stateful systems was understood to be desirable, since complexity precludes conscious human supervision of every component [10]. As its biological counterpart implies, autonomic carries a meaning that is mainly concerned with reacting to "internal stimuli" [11].

The stated aim of AC is to have technology which manages technology, allowing the systems to keep working [12]. This concept requires an arrangement of self-properties, or we could say "unsupervised" properties, usually comprised of self- CHOP, that is, self-Configuring, self-Healing, self-Optimising, self-Protecting.

Fundamental implementations of AC are an effective loop comprised of Monitor, Analyse, Plan, Execute phases with results committed to a shared Knowledge base, resulting in a model known as MAPE-K (see Figure 1) [13].

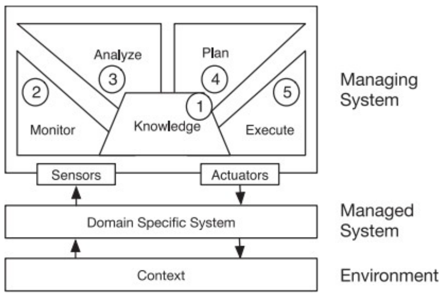


Figure 1. MAPE-K. Model [13, pp. 8].

An autonomic system should be composed of autonomic agents dedicated to specific elements of the system dubbed an Autonomic Manager (AM) [10, pp. 4]. The Monitoring phase imports the awareness of the system within its environment into the loop and updates the Knowledge. This is Analysed in the next phase to formulate updated policies. These are scheduled in the Planning phase and finally Executed in order to affect the system [13, pp. 7].

Blockchain, as the name implies superficially, is a database paradigm in which data is embedded within a series (or chain) of time stamped containers referred to as blocks. The structure that gives blockchain specialised properties is the addition, within each block, of a cryptographic hash derived from the hash of previous blocks. This ensures the chain of blocks cannot be broken without invalidating the rest of the chain, making modification impossible unless the whole chain is rewritten. In a public internet based permissionless ledger it is not adequate to leave this chain open to the possibility of being overwritten, but for a private ledger, with appropriate security, trust model requirements can be lowered and this form can be considered immutable in principle [24]. Blockchain demonstrates many autonomic features in its design. It is a self- securing, self-correcting, self-healing, distributed redundant database.

The purpose so far, is to suggest that QMS tends towards problems, which AC was designed to address in the domain of computing. The paper next addresses more detail on related works; then presents the proof of concept design and implementation; followed by testing and validation of such. A discussion and conclusion are then presented.

II. Related Works

There is work within the domain of AC, which shows that it has been converging with other areas of research and this certainly includes manufacturing and quality. It is particularly going to be of interest as many manufacturers begin transition into the Industry 4.0 context.

One proposal implementing multiple autonomic agents in differing roles, is the "Autonomic Manufacturing Execution System (@MES)". In a traditional Manufacturing Execution System (MES), there is a fairly rigid planning/scheduling layer and a final execution layer, similar to the tail of MAPE. @MES proposes an MES in which agents for Order Acceptance (OA) scheduling and Resource Allocation (RA) and for execution, receive feedback from the environment, accumulate shared knowledge, then model results to optimize the manufacturing process. Predicting possible states should also allow the autonomic agents to dynamically respond to orders within the system and allocate needed resources in an optimal proactive way, rather than in a constant reactive state [14].

"Autonomic computing in manufacturing process coordination in industry 4.0 context" [20] published in 2020 is a fairly detailed work, which proposes a framework of processes made up of several autonomic cycles, to enable manufacturing systems to operate autonomously and in a self- CHOP way, centred upon the idea of a "Smart Product" which configures the entire system of systems, from prioritizing incoming orders according to priority time-line, to orchestrating the request of raw materials required for the prioritized products, determining which components of the production line(s) are available and most suited to the given Smart Product and configuring the entire production line accordingly. As can be seen, the goals of industry 4.0 conceptually assume that the manufacturing systems are built to function from beginning to end, autonomically.

A very different and much narrower problem is considered in a 2014 paper entitled "Autonomic Software Systems - Developing for Self-Managing Legacy Systems" [21], in which a completely legacy web order system with several different components is essentially retro-fitted to interoperate with modern Amazon Web Services through XML translators. These legacy components included, a program written in COBOL, an FTP server, a SOAP server and a Web server. The legacy components were managed through a solution which comprised of several tasks, which were maintained through each task scheduling the next iteration of itself according to the given schedule, as its first activity. The schedule of tasks would be adjusted within given thresholds. This shows that a type of autonomicity can be achieved on systems that were not built to consider it, by allowing software to monitor and manage recovery that would normally require human interventions and effort to maintain. The newer approach to software has been successfully meshed with the old and this is also an objective in trying to build an autonomic QMS.

"Blockchain-Enabled Open Quality System for Smart Manufacturing: Applications and Challenges" [22] in 2022, is alongside the domain of research in this paper and provides a recognition of the role that blockchain technology may have in achieving a future trusted QMS which can operate in a "smart", or Industry 4.0 context, such as that discussed in [20]. The statement made that "very little attention has been paid to the incompatibility of the QMS with Industry 4.0 technologies that govern smart factory and the problems that may arise in the future owing to the

 ISBN: 978-1-68558-356-9

incompatibility" [22, pp. 10], is certainly recognized by the led author's industrial experience as well. [22] introduces this concept under the name "Open Quality". The term "open quality" appears to be relatively new. While this same terminology can also be found in a limited number of other prior works, those do not seem to be addressing the same solution, of incorporating blockchain into the quality system, but rather apply the term to particular models discussed in their respective works. More commonly, the integration of QMS with end-to- end smart manufacturing is dubbed "Quality 4.0".

It can be observed from the above that some work has been done to raise awareness of the need for smart systems to become autonomic. It is especially encouraging that the QMS is now beginning to be recognized as a system which requires consideration and development. The research into how QMS could benefit from blockchain technology, which is to this end, essentially an extremely robust, perhaps even immutable ledger would seem to make it a natural ally in the quality environment. However, none of the works so far considered appear to be directly addressing the same problem as that which is introduced in this paper. This work will attempt to demonstrate a proof of concept, for an autonomic system, incorporating endpoint computing nodes, aware of a subset of relevant quality activities within a QMS and utilising a blockchain ledger to record and corroborate activities in a virtually immutable way. Where [20] is concerned mainly with optimizing the manufacturing process through an autonomic cycle, our system will be focused upon supporting and optimizing the QMS activity which necessarily parallels the manufacturing process. As [21] attempted to bridge the gap between disparate technologies in the case of web services, our system will likewise introduce a layer of autonomicity to endpoints involved in a QMS and as [22] suggests blockchain can support quality systems, this work does also, through an AC lens.

III. Research Proof of Concept System Design and Implementation

The complexity rises in QMS as it rises in automation, yet as mentioned QMS remains essentially a manual document- based system. At the time of writing, as automation advances toward Industry 4.0 and, increasingly, the smart factory, it is clear that system complexity is set to rise exponentially. Taking the QMS standard manual approach to vetting data and understanding how it can be audited in a meaningful way, is a challenge that the quality industry is only just beginning to recognise.

The goal of this proof of concept is not just to provide corroboration of quality data events, but to provide a demonstration of an AC framework where the loops of the AM's synchronise a sub-set of quality system goals. The approach taken within the design will now be described and summarised.

To begin the design, initially a feature list of terms was compiled by taking a simplified Domain Driven Design (DDD) approach, considering the core domain questions that arise within the context of QMS and which of these could be addressed by the proposal [23]. The domain of application for the AC system, will be the management of computing end- point states. The QMS domain itself answers as already discussed above, with documentation, schedules and the accountability of roles. Since QMS includes a wide range of activities, for this purpose, the QMS documentation element is being considered in the supersets of the documentation, as major qualification states which are referenced in GAMP previously mentioned. These are Design Qualification (DQ), Install Qualification (IQ), Operational Qualification (OQ) and Performance Qualification (PQ). To this will be added the Standard Operating Procedure (SOP) representing the fully qualified training state and Non-Conformance (NC) fault state. The main scheduling requirement will be to manage end-point calibration activities. In order for this to be intelligible to the QMS, endpoints need to have both an individual identity and a QMS system classification. It is important that compliance is considered within the design and that endpoints out of compliance with policies are managed. To align with the principles of AC, the QMS will be aimed toward self- management, but with respect to the QMS it will be necessary to inform human roles to perform activities.

A. Nodes and Architecture

This design is conceived as AM roles with their inputs and outputs, together with a recording ledger that functions as long- term and trusted memory of events. Given the domain of QMS, with respect to computing endpoints, it was determined that system inputs would function based upon state values. Outputs would be feedback of the state data and responses as state updates. Basic QMS states can be understood in reflecting the qualification document phases above. Maintaining these major phase states addresses part of the problem. The subset of these phases can be considered in the particular working document identity issued to the endpoint, under which current quality activities are to be recorded. This will be maintained as a QMS documentation substate. As each QMS state advances, the current valid working document identifier is issued as a state value to the endpoint.

Since there can be any number of computing endpoints, in the opinion of the authors, a peer-to-peer (P2P) distributed architecture may evidently be considered a wholly legitimate approach to solving aspects of an AC problem and would allow the distribution of the ledger activities and redundancy of functionality, some aspects of which will be discussed later. However, for this design, primarily because a P2P model would require every agent to replicate and process all capabilities of the system on each node, whilst coordinating the distribution of data between nodes, the simplicity of a centralised orchestrating approach is being considered preferable. The justification is providing a clear separation of AM duties, maintaining the end-point and supervisory distinction to best demonstrate the above stated concept. Removing processing load of QMS decision making from the endpoints, as well as the storage requirement to commit and preserve ledger data is an additional consideration.

The end-point AM shall be designated an Autonomic End-Point (AEP). The AEP facilitates data gathering and where applicable, the ability to detect and interface with available operating system and software application features to provide the AM with monitoring values, including information most important to the QMS directly, namely the user and system identification. The AEP will both receive and maintain its system state and report events detected by function calls within its main loop. Further, the AEP agent makes some provision for potential human roles and feedback into the QMS system.

The central supervising AM will assume both the QMS and blockchain ledger recording activities simultaneously and will hereafter be referred to as Autonomic Ledger (AL), receiving feedback from the AEP's. The AL node activities will involve the detection and registration of AEP nodes, committing received events to the ledger, scheduling QMS activities, such as the previously mentioned calibrations and responding with appropriate commands and states, determined by a response policy function. The internode messaging shall be understood in the context of the ledger requirements.

B. Ledger Design

A time-series ledger is needed to provide a corroborative record of QMS endpoint activities by the AL This implementation proposes the particular suitability of blockchain technology for the previously mentioned reasons. Chief considerations are the security and virtual immutability of records so that they can be considered trustworthy from the perspective of the QMS, comprising a reliable, sequential long- term memory of events within the system. The data structure proposed is again derived from asking questions of the domain, specifically with reference to the Attributable, Legible, Contemporaneous, Original, Accurate (ALCOA) principle [5, pp. 5]. The proposed basic ledger storage structure adopts the so-called 5W questions.

- <u>w</u>ho is interacting with the endpoint, so the record is attributable.
- <u>w</u>here does the record come from, i.e. which endpoint.
- <u>w</u>hat information is being recorded.
- <u>w</u>hy was this information generated.
- <u>w</u>hen together with an independent index timestamp provides assurance that the record is contemporaneous and original.

This form has the advantage of being derived from a standard designed to be human readable, resulting in a legible and accurate record which captures concise information.

The standard definition of a blockchain referenced in the introduction, goes further than this implementation intends to. This ledger will be recording 5W records into blocks and the chain of blocks will be individually hashed to discount modification. In a private and closed system, this is being deemed sufficient for the implementation to exemplify a form of ALCOA compliance. The technical implementation of the ledger, is as a Python class, with persistent storage as a ZARR array. The class implements functions for the AL node to query the chain as required.

C. Inter-node messaging, command and policies framework

To facilitate internode communication, messaging framework options were evaluated, from using raw sockets, to messaging queues with exchange, such as RabbitMQ. ZeroMQ was chosen because it is lightweight, widely used, does not require the use of an exchange or broker and provides several messaging pattern options. The publish/subscribe pattern with polling seems to be most suitable and provides universal broadcasts but allows refinement to relevant endpoints by using topics. Although ZeroMQ provides a high level of assurance that message transmission and reception is robust, once again, to be viable within QMS, that assurance must be extended to the ideal that no GxP data record may ever be lost. A solution to this problem is buffering messages in a persistent state until they have been issued and consumed. The network message transmission and reception were therefore abstracted from the AM's, which instead process incoming and outgoing messages through having message buffers committed to persistent storage. This both guarantees that all messages are available and makes it possible to process incoming and outgoing messages rapidly from local storage on the AM loop. This design is comparable to the "Titanic" pattern described in the ZeroMQ documentation [25, pp. 6].

The formatting of messages to be sent to the AL from the AEP were naturally dictated by the ledger design decisions and assume the W5-part configuration already described. As well as allowing the capture of information in a format conducive to QMS requirements, this configuration was found to lend itself well to issue and reception of commands, where this was needed. In this further scenario, the reason, or why field, is adapted to become a command and the data, or what field embeds the sub detail, with when, who and concerning where being retained, thus preserving integrity.

Once the message is received, the required response is determined on the AL loop, by a response policy evaluating the incoming why field value and selecting the appropriate response. Broadly, this response will be one or more of these 4 steps:

- Commit the received data to the blockchain ledger.
- Respond with a message acknowledgement.
- Process a command on AL.
- Issue a command from AL to relevant AEP.

Special case commands allow for roles to update the QMS and directly issue new node additions and state changes.

Messages in the direction AL to AEP are instead in a simpler 2-part format, being the command and message, or supporting data. As an example, the AL communicates the current valid state regularly to the AEP. The AEP recognises the first part of this message as the state command and the second part provides the state itself. The AEP's own response policy writes this feedback to the AEP's own state information.

D. State Information

On the AL, state information is maintained in a QMS registration listing of AEP nodes. This information is comprised of unique system information, including the network IP address and system identity. This information is used during the network messaging phase by iterating through the valid nodes. This listing also contains the current valid state for each of the nodes based upon the inputs received to the present, which can be re-issued to the AEP's. The AEP meanwhile maintains its current state in its own configuration. This allows temporary states to be put in place on endpoints, whilst preserving the last valid QMS state on the AL, ready to be reissued when the reason for the temporary state has been addressed.

As with the messaging buffers, state information is preserved on persistent storage, so that interruptions to the node will not result in loss of configuration or QMS state.

E. Scheduling

The scheduler was naturally conceived as a complementary time series consisting of messages which would be updated with future events by the AL loop, as part of its planning activity. The scheduler time stamps are encountered inside a moving window during the AL loop. The in-scope messages are executed and the embedded activities carried out. This component of the loop is essential to managing the QMS and provides a means to notify affected parties and execute commands on supervised nodes at the appropriate times. The scheduler can be viewed as reflecting the AL policies, some of which may be fixed and others reviewed and updated when relevant events are available within the ledger.

Schedules needed will include required maintenance and calibrations, system life cycle, audits, backup and data retention. In addition, individual commands which need to be reissued by the AL at specific or repeated times.

F. The Autonomic Loops

With the structure as laid out above, the autonomic loops cycle on both agents to synchronise activities which will now be described. When the loop begins, it first checks for an existing configuration on persistent storage. If not found configuration is requested and then immediately committed to persistent storage. Likewise, the default behaviour for all functions involving tables in memory, is to check for and then generate, or attempt to load from persistent storage, update and commit back to persistent storage. This design allows the loop state to be preserved through interruption, such as power loss and resume operation from its previously held state values. Both the AEP and AL loop incorporate several functions which compile such tables functioning as short-term local memory. The loops are logically divided into the MAPE-K categories, although relevant activity may not strictly be confining that activity to the logical phase of the loop.

The most fundamental AEP configuration is the system role. While GxP systems will be configured, to engage the full autonomic loop, predefined human roles discard all but subscription to relevant messaging topics and allow responses, or manual commands to be issued to the AL. This is to mention how the human role is incorporated into the autonomic QMS.

On the AEP, this GxP role begins self-monitoring and gathers or updates the self-state including the filesystem, operating system information, currently logged on user and executable programs. During analysis functions, the policies of the AEP are invoked with detected changes and events. The purpose of planning is to reinforce analysis, by allowing values of interest to be preserved in timestamped checkpoints. This provides the loop with a simple universal means to compare past, present and future values. As an example, consider an endpoint is launching an executable on a schedule to perform an essential QMS function. The AEP records the timestamp when the executable is detected and then records the timestamp when the executable is no longer present. After several regular executions, the scheduler has preserved the valid past pattern and now may infer a forecasting pattern and schedule these as future events. This allows the loop to have self-knowledge of and immediately detect and self-manage deviations from norms, rather than reacting to breached values based purely upon rules or thresholds. The execution phase involves the incoming messages from the AL calling applicable functions and then sending the outgoing messages. Knowledge of the AEP intersects with the other phases.

The AL loop maintains the same form, with its own functions and incorporates the ledger in addition to messaging. A collection is initialised to capture ledger events throughout the loop. Monitoring updates the list of detected nodes from a network IP scanning function. The reception of messages would logically reside here, but is offloaded from the AL loop to the parallel messaging service. Having detected the current network state, the analysis phase compares detected nodes with the QMS node listing to determine if a new node needs to be registered in the state table. After this the incoming message buffer is processed, items committed to the ledger collection and then a response policy is called upon to determine which actions the AL will take. Planning also revolves around the scheduling which has already been outlined. It is concerned with overarching QMS activities rather than endpoint events, such as calibrations and audits. The scheduler stores these planned events as messages and recipients. After this outgoing message responses from either the response policies, or the schedulers are executed. Although knowledge touches all of the phases, the blockchain is called at the end of the loop within K, to record the loop events before finally performing some self-protecting activity in the form of blockchain integrity checks, including block length and tampering detection.

The working name for this system will be the Supervised Quality Environment Autonomic Ledger (SQuEAL), with reference to telling all about the QMS environment and taking action to synchronise and correct.

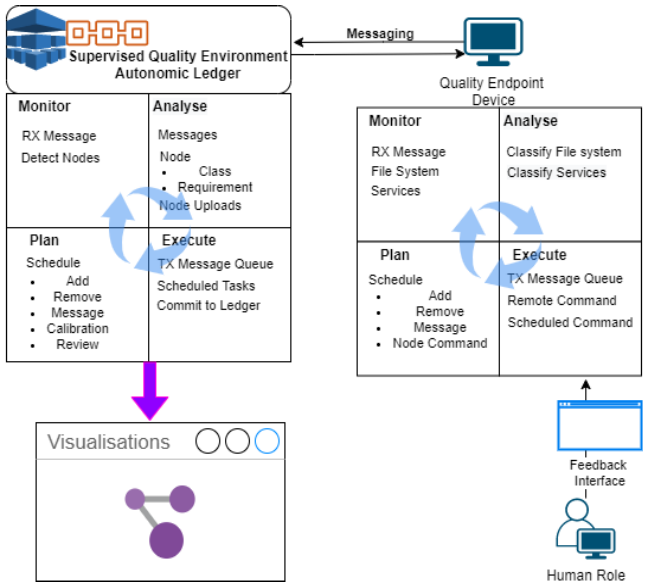


Figure 2. SQuEAL.

A summary representation of the system design is seen in Figure 2.

IV. TESTS AND EVALUATION

The complexity and integrity requirements of utilising a live QMS system in a manufacturing environment, or reconstructing one even in part, mean that as part of this proof of concept, it was decided instead to construct a simulation which would demonstrate aspects of the operation of the concept. This is a node based on the AEP autonomic loop, with some significant modification. The purpose of the simulator, is generating a dynamic narrative, feeding to the real AL node messages representing an environment of endpoints. The simulator accepts and records the AL responses and uses them to update simulated AEP state information. This is possible to accommodate, in part because the existing design of the AL is already based upon accepting messages, storing state values and integrates the special cases previously mentioned. The simulator node, as with the AEP and AL nodes retains the persistent state feature, allowing the simulation state to be preserved between sessions. A number of preset dictionaries are present and others are generated when the simulation begins. At the top-level is the definition of the valid QMS system classes. This defines the system names, together with the executables and services which make up the system operation on the simulated endpoint. Next, a dictionary of endpoint names are compiled together with randomly generated weight values, which allows the simulation to vary the importance of the simulated endpoint within the simulation narrative. Alongside this, a similar dictionary of user agents with weighted values are generated. The simulator first attempts to send any outstanding messages in the outgoing message buffer to the AL thereby clearing the session and getting ready to receive state updates.

Now the simulation loop moves onto generating actions, first by generating a random number of actions which will take place during the current cycle. A decision delay value sets the pace of simulation decisions in seconds. The action generation loop executes for the chosen number of actions with the purpose of building a completed 5W record to represent a detected action in the system. Each action decision will assign an agent (who) to an endpoint (where), with the current timestamp (when) and then choose an event (why) with relevant generated data (what). The events and associated data are produced in a decision function which takes the current agent and endpoint as input and chooses randomly from a valid list of action categories. Each category contains a subset of events including a none-event, which are again chosen randomly. If the event involves an executable or service, this is chosen from the system dictionary, matching the class of the current endpoint being considered. There are also a number of fault states. To vary the control of simulated events, user agent decisions to utilise specific endpoints are accompanied by a generated session length. For the duration of this session, the user agent is blocked from taking a further action and the endpoint being used is also blocked from subsequent actions until the session expires. Further, both regular and random opportunities are presented for introducing less frequent events. One such case is calibration messages which occur at intervals of n simulation loops. The next action taken by an agent at the end of a session provides a randomised opportunity to have the agent advance the current system state. This approximates that there are activities during qualification phases. The QMS system state for each endpoint is maintained in a table and updated when the advance system state function is called.

A completed message is now sent to the AL with a special case message response prefix identifying the simulator, to enable the AL to respond on the simulator topic. The AL response appends the identification of the simulated endpoint to the end of its reply message string. Since the simulator is listening to its own topic, this identifier is used to correctly record endpoint updates in the simulators state table. The end of the simulation loop processes any incoming messages last, thus updating the state values for the next iteration.

The evaluation of system performance is now derived from examination of the effects messaging inputs and outputs have upon system states and observing self-heal autonomic features of the AEP and AL providing stable operation.

Experiments were conducted on virtual machines representing the AEP simulator and the AL node with a private internal network. The AEP simulator accommodates parameters to vary the number of simulated endpoints (W) and agents (U) in the environment, as well as the number of decisions (D) per loop. These were set initially with small scale values of W=30, U=12 and D=8, followed by larger scale experimentation at W=300, U=150 and D=8 with a loop delay of 1 second maintained to moderate the system progress. In simulation mode the QMS schedules execute using seconds, rather than days. This allows the cycles to be observed in a tighter time frame and each experiment was executed for a number of hours to facilitate results and establish stability.

 ISBN: 978-1-68558-356-9

Viewing the progress of self-configuration, state updates and node classification is facilitated by visualisations which the AL outputs at regular loop intervals. Node visualisation was implemented in a graph network providing evidence of the registration of nodes and classification over time, which was observed to progress successfully throughout the duration. A correlation function was developed to query the blockchain and provide insights into potential associations. Examples correlate who with where and where with why event descriptors, which are composed in a correlation matrix and then displayed in a heatmap format. This facilitated observation of both agent and endpoint activity over time. A blockchain events viewing function maps recorded events to colours, allowing the activities to be visualised as a sequence of bars. An ASL_LOOPTIME_SECS event is committed to the ledger upon every iteration of the loop. This functions as a form of heartbeat for the data, so that the real time when events are being recorded can be observed. This is viewed alongside the QMS events, so as to not obscure them.

Finally, AL loop statistics are collected throughout the runtime, thus demonstrating the operation and time penalty of the various functions. The AL was suspended during experiment to observe the effect on loop timing and reliability of recovery.

V. DISCUSSION

In both small and larger scale experiments, the nodes performed as expected, including persistent storage handling of short-term memory in dataframes, long term blockchain memory and the message buffers. Maintaining system operation and handling all encountered events is fundamental to AC system design but was challenging to achieve in some respects. As an example, the frequency of persistent dataframe storage to meet the requirements of this implementation required building a shared function to handle pandas disk operations gracefully. This was extended to provide redundant copies of each dataframe and handle automatic recovery from the copy in the event of any error reading the primary copy. The blockchain functionality was similarly adapted. ZeroMQ messaging with persistent buffers proved to be reliable, but a backup is also provided to capture misunderstood and discarded messages in a log. This is an acceptable mitigation which ensures compliance with GxP requirements to preserve all records and allows for investigation and CI.

Loop statistics were gathered in both small and large cases and on host computers of differing specification. Comparing a lower specification host computer with a higher specification, during regular activities, the AL loop time would progress from typically 1-2 seconds vs. 5 seconds, up to a peak of 30 seconds vs. 60 seconds during which all events are both captured and analysed with visualisation outputs. The example of 300 simulated nodes is perhaps comparable to a Small-Medium Enterprise (SME). There are some timing aspects which cannot be moderated by computer performance, such as network polling timeout values, but the performance improvement provided by modest increases in computing power implies that an AL adopting this form can match or exceed the rate of QMS data generation in such an environment.

Maintenance of states during suspension of the AL node with parameters W=30 and A=12 was excellent. Suspending the AL for 20 minutes resulted in delayed messages being handled by the AL in the next available cycle, processing the incoming buffer, committing transactions and generating all responses within 35 seconds. On the accelerated time scale, this could be taken to represent many weeks of QMS activity and the AL both recovered and updated the system states successfully. The suspension of the AL loop and recovery can be seen in Figure 3. Note the events are recorded within the suspended period which can be observed in the lower ASL_LOOPTIME bar.

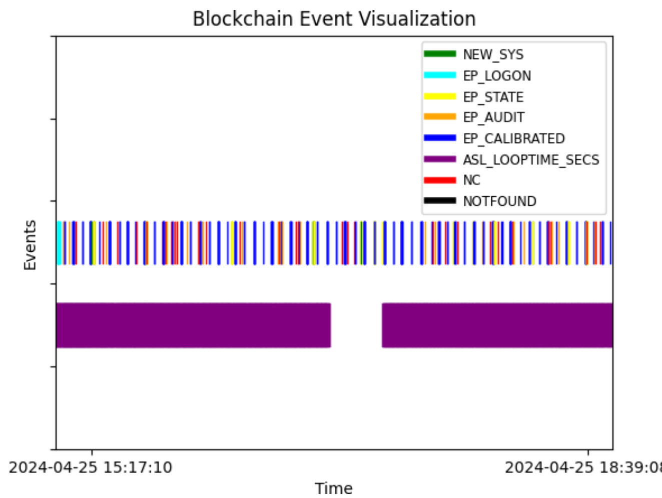


Figure 3. Blockchain event visualisation showing AL recovery.

Calibration activity events are correctly clustered and can be observed at intervals in the blockchain. As would be expected, given the randomised nature of the events data generation, correlation was observed most strongly in the early stages of the experiment and would gradually decompose as it progressed.

Having demonstrated a working concept of maintaining stateful QMS policies with respect to endpoints, there are definitely a considerable number of areas which would warrant further work. Extending the range of supported QMS states and policies, as well as the documentation issuance would provide the needed complexity to begin approaching the real world problem. The AL could be equipped to also generate, classify, issue and monitor QMS documentation, allowing document events to be treated as well as endpoints. With respect to Self- CHOP, the AEP could provide a subset of its self-knowledge to the AL, allowing similar systems to be classified automatically by the AL, even if the actual system class is currently unknown. Redundancy could be extended to include P2P technology. Sequence forecasting, such as through an LSTM (Long Short-Term Memory) RNN (Recurrent Neural Network) would provide a mechanism for improving self- optimisation in planning and schedule suggestions. The scheduler could conceivably be extended to facilitate messaging of external systems, allowing the autonomic QMS loop supervision of other systems QMS activities. For example, automated product sampling and testing schedules.

It's essential that any discussion of reducing human effort is accompanied by considering the potential ethical implications. With regards to QMS, the human factors affect the individuals working within QMS processes, but also involve customers

[26] and in this area can include critical product categories, such as food, medicine and medical devices. Product recalls can also be extremely costly and wasteful [27]. It has been suggested that organisational adoption of QMS resulting in improvement of outcomes, has a net positive impact upon customer and job satisfaction as well as environmental impact [28, pp. 8-11]. The goal of this concept as stated, is to augment those outcomes only by synchronising activities, not by replacing them. Further, a QMS primarily exists to effectively regulate human processes and activities. This proposal is focused upon addressing the rising complexity with regards to the non-human elements of the system and relieving the system of gaps as far as possible and reducing in that sense wasted human effort. It does not consider or include the separate ethical implications of automation proper and Industry 4.0.

VI. Conclusion

This research provided an opportunity to deepen understanding of both the QMS and AC and a relationship which may soon come to exist between them. The core of the operation demonstrated, is adhering to and attempting to maintain a complex set of policies, whilst building in some necessary accommodation of the human roles.

This paper proposes the SQuEAL AC concept as being a potentially suitable framework for integrating AC within the domain of QMS computing endpoints, although tentatively, as it is evident a lot of work remains to be done. Given more development, it would seem possible to incorporate some additional aforementioned features. An interesting case would be testing the suitability of AEP AM's providing touch points on a wider range of operating systems and devices, such as HMI's and PLC's in an automated equipment setting, bringing those inside the autonomic QMS.

The stability of the system in its current form provided validation of adhering to AC design principles and is extremely promising. Adopting the form of the MAPE-K loop with its sensors and effectors, enabled all the necessary activities to be considered and logically ordered. Building autonomically [29], is about building the level of trust into the system which GxP demands.

Acknowledgment

This paper was supported by Ulster University's Artificial Intelligence Research Group (AIRG) and the School of Computing.

References

[1] E. Henson, "The role of integrity in a GXP environment", Journal of GXP Compliance, vol. 13, no. 2, pp. 79-86, 2009.

[2] Mehmet Sıtkı İlkay and Emre Aslan, "The effect of the ISO 9001 quality management system on the performance of SMEs", International Journal of Quality & Reliability Management, ISSN: 0265-671X, July 2012.

[3] Milton P Dentch, "The ISO 9001:2015 Implementation Handbook: Using the Process Approach to Build a Quality Management System", ASQ Quality Press, Milwaukee, Wisconsin, 2017.

[4] P.I.C.O. Scheme, "Good Practices for Computerised Systems in Regulated "Gxp" Environments", Pharmaceutical Inspection Convention, PI, pp. 011-1, 2007.

[5] R. McDowall, "Data Integrity Focus, Part VIII: What is Good Documentation Practice (GDocP)?", LCGC North America, vol. 37, no. 9, pp.684-688, 2019.

[6] J. Avellanet, "FDA 21 CFR Part 11 Revisited", BioProcess International, 2009.

[7] K.C. Martin and A. Perez, "GAMP 5 quality risk management approach", Pharmaceutical Engineering, vol. 28, no. 3, p.24, 2008.

[8] F. Heylighen and C. Gershenson, "The meaning of self-organization in computing", IEEE Intelligent Systems, vol. 18, no. 4, 2003.

[9] Roy Sterritt, Manish Parashar, Huaglory Tianfield and Rainer Unland, "A concise introduction to autonomic computing", Advanced Engineering Informatics, vol. 19, no. 3, pp. 181-187, 2005, doi:10.1016/j.aei.2005.05.012.

[10] J. O. Kephart and D. M. Chess, "The vision of autonomic computing," Computer, vol. 36, no. 1, pp. 41-50, Jan. 2003, doi:10.1109/MC.2003.1160055.

[11] A. Williams, "Defining autonomy in systems: Challenges and solutions", Autonomous systems: issues for defence policymakers, pp. 27-64, 2015.

[12] B. Li, X. Zhang and F. Xie, "Modeling and analyzing of autonomic intrusion tolerant based on PEPA", International Conference on Computer and Information Application, Tianjin, China, pp. 244-247, 2010, doi: 10.1109/ICCIA.2010.6141582.

[13] Didac Gil De La Iglesia and Danny Weyns. "MAPE-K Formal Templates to Rigorously Design Behaviors for Self-Adaptive Systems", ACM Trans. Auton. Adapt. Syst. vol. 10, no. 3, pp. 1-31, 2015, doi: 10.1145/2724719.

[14] Milagros Rolón and Ernesto Martínez, "Agent-based modeling and simulation of an autonomic manufacturing execution system", Computers in Industry, 63(1), pp 53-78, ISSN 0166-3615, 2012, doi: 10.1016/j.compind.2011.10.005.

[15] P. Rehacek, "Costs of quality or quality costs", International Journal of Advanced and Applied Sciences, vol. 5, no. 2, pp. 8-13, 2018.

[16] Suresh Kumar Krishnan, Arawati Agus and Nooreha Husain, "Cost of quality: The hidden costs", Total Quality Management, vol. 11, no. 4-6, pp 844- 848, 2000, doi: 10.1080/09544120050008309.

[17] M. Juran, and A. Godfrey, "Juran's Quality Handbook", 5th Edition, McGraw-Hill Companies, Inc., Washington DC., 1998.

[18] Ida Gremyr, Jan Lenning, Mattias Elg and Jason Martin, "Increasing the value of quality management systems", International Journal of Quality and Service Sciences, 2021.

[19] M.M. Dietrich, and M.S. Popa, "Measurement loop, automatic tuning and product quality tracking", MATEC Web of Conferences vol. 178, EDP Sciences, 2018, doi: 10.1051/matecconf/201817808001.

[20] Manuel Sanchez, Ernesto Exposito and Jose Aguilar, "Autonomic computing in manufacturing process coordination in industry 4.0 context", Journal of Industrial Information Integration, vol. 19, 2020.

[21] J. J. Mulcahy and S. Huang, "Autonomic Software Systems: Developing for Self-Managing Legacy Systems," IEEE International Conference on Software Maintenance and Evolution, Victoria, BC, Canada, pp. 549-552, 2014, doi: 10.1109/ICSME.2014.92.

[22] S. Ali, W.S. Shin, and H. Song, “Blockchain-Enabled Open Quality System for Smart Manufacturing: Applications and Challenges”. Sustainability 2022, doi: 10.3390/su141811677.

[23] V. Vernon, “Domain-driven design distilled”, Addison-Wesley Professional, 2016.

[24] L. Ghiro, F. Restuccia, S. D'Oro, S. Basagni, T. Melodia, L. Maccari and R.L. Cigno, “What is a Blockchain? A Definition to Clarify the Role of the Blockchain in the Internet of Things”, 2021, arXiv preprint arXiv:2102.03750.

[25] Z. Meng, Z. Wu, C. Muvianto and J. Gray, “A Data-Oriented M2M Messaging Mechanism for Industrial IoT Applications”, IEEE Internet of Things Journal , vol. 4, no. 1, pp 236 – 246, 2017, doi: 10.1109/JIOT.2016.2646375.

[26] Władysław Mantura, "Human factors in quality management." Human Factors and Ergonomics in Manufacturing & Service Industries vol. 18, no. 5, pp 565-579, 2008.

[27] Howard Abbott, “Product Recall: The Cost of Putting the Distribution Chain into Reverse.” Journal of Brand Management, vol. 1, pp. 163–167, 1993, doi:10.1057/bm.1993.23.

[28] L. Zhao, J. Gu, J. Abbas, D. Kirikkaleli and X. G. Yue, “Does quality management system help organizations in achieving environmental innovation and sustainability goals? A structural analysis”, Economic Research-Ekonomska Istraživanja, vol. 36, no. 1, pp 2484–2507, 2023, doi: 10.1080/1331677X.2022.2100436.

[29] Roy Sterritt, “Autonomic computing”, Innovations Syst Softw Eng, vol. 1, pp 79–88, 2005, doi: 10.1007/s11334-005-0001-5.

 ISBN: 978-1-68558-356-9

A Reproducible Framework for Evaluating Autonomic Swarm Recruitment

Enhancing MegaSwarm with Systematic Metrics, Visualization, and Performance Analysis

Parvathy Preeya Santhosh Kumar, Liam McGuigan, Catherine Saunders, Roy Sterritt
School Of Computing, Faculty of Computing, Engineering and Built Environment
Ulster University
Belfast, Northern Ireland
email: {Preeya_Santhosh_Kumar-P | ce.saunders | r.sterritt} @ulster.ac.uk | lmcguigan@gmail.com

***Abstract*— Swarm robotics provides scalability in task allocation due to decentralized control but the performance drops when robots act in the absence of organized coordination, especially when the environment is clustered or unbalanced. The biologically inspired behaviors in SwarmSim2 and MegaSwarm simulators were pheromone trails, recruitment, autonomic pulse communications and fault-tolerant sensing. Although these studies were able to define significant concepts, they were mostly restricted to case-based illustrations and statistical observation of one-off situations. In this paper, a research quality experimental framework is built on top of MegaSwarm that allows systematic and reproducible evaluation of recruitment strategies. Automation of batch sweeps of recruitment parameters, standardized metrics logging, and reproducible visualization output in the form of heatmaps, pheromone fields and time series charts have been added. The four recruitment modes, Off, OneResponder, MultiResponders and Blackboard, are tested with different swarm sizes and spatial layouts. Baseline experiments performed partial completion of the tasks (12-18 of 20 sites, mean scores plateaued at 0.6-0.8), whereas recruitment strategies increased coverage and completion times dropped significantly with Blackboard showing near total completion. This framework will turn MegaSwarm into an analytical tool to quantitatively study trade-offs between time, energy and communication in swarm recruitment.**

Keywords—MegaSwarm; Swarm Robotics; Pheromone-Trails; Heatmap; Autonomic Pulse Communication; Recruitment Modes; Fault -Tolerant Sensing.

I. INTRODUCTION

Swarm robotics studies the possibility of large-scale cooperation of simple robots without centralized control to solve complex tasks [1]. Swarm behaviors are inspired by natural systems (e.g., ants and bees) and are desirable in tasks where scalability, robustness, and/or adaptability are required (e.g., disaster response, persistent environmental monitoring, or planetary exploration) [2][3]. But decentralized coordination also presents problems. Without orderly communication, robots can waste effort, clump together inefficiently or fail to explore important regions. This constrains overall efficiency, especially in landscapes with spatially unbalanced or clustered tasks. To overcome these inefficiencies, scientists have suggested some biologically inspired solutions which include the use of pheromone trails and recruitment. These strategies enable robots to impact others without having to intervene directly into their living places or directly seeking help at workplaces. Although earlier prototypes including SwarmSim2 and MegaSwarm [4] also embody these concepts, much prior work is presented as case-based demonstrations of particular autonomic behavior, such as adaptive communication and cooperation strategies [4]-[6]. These demonstrations were able to confirm important points but were not reproducible at the level of recruitment strategy and parameter range. To overcome this shortcoming, mechanisms that are biologically inspired like pheromone trails and recruitment strategies are used. Pheromone stigmergy can be used to make robots strengthen a profitable route in the landscape, whereas recruitment can be used to actively signal seeking assistance at a particular location. Some of the models that exist are; One-Responder, Multi-Responder, and Blackboard [5][6]. Such mechanisms can balance loads and speed up the completion of tasks, but prior research has been limited, usually studying single cases and not systematically comparing across parameter spaces.

In this paper, the gap is filled by turning MegaSwarm into a research-grade experimental framework, i.e., beyond a prototype demonstrator. We have made changes to: Automatic sweeps of the recruitment strategies (Off, One-Responder, Multi-Responder, Blackboard) across need, Time-To-Live (TTL) and clustering gain. Uniform measurements of completion time, cost of energy, volume of communication and coverage of the tasks. Outputs in the form of heatmaps, pheromone fields and time-series plot to facilitate direct scientific reporting. Preliminary results reveal the significance of recruitment: uncoordinated swarms of 100 robots visited only 12-18 of 20 sites, with mean site scores stagnating at around 0.6 and 0.8, whereas with recruitment approaches, in particular Blackboard-based approaches, full site completion was approached with rates approaching 100 percent and convergence time reduced. The results can be summarized in three contributions: (i) adding reproducible research capabilities to MegaSwarm, (ii) systematically comparing recruitment strategies with different swarm sizes and layouts, and (iii) offering an analysis methodology of time, energy and communication trade-offs in swarm robotics.

In Section II, we discuss related works; Section III describes the methodology; Section IV details the results; section V a contextual discussion including limitations and future work, and VI concludes the paper.

II. RELATED WORKS

Swarm robotics leverages biologically inspired mechanisms to allow a decentralized system of simple robots to solve a complex task collectively. Initial work focused on the application of stigmergy and recruitment to handle task distribution, being inspired by ants, bees and termites. The ability to leave pheromones that influence other robots, and to signal help explicitly at the location of the task, is achieved through recruitment. These mechanisms allow scalable cooperation to be used but their effectiveness is highly sensitive to environmental clustering, swarm density and communication strategies.

SwarmSim2 simulator was an early prototype of testing pheromone foraging and recruitment. MegaSwarm was its successor, a more modular environment that supports a bigger swarm population and the idea of Autonomic Computing [7]-[10]. McGuigan [4] also presented the concept of Autonomic Pulse Communications (APC) in MegaSwarm where adaptive broadcast ranges helped increase swarm connectivity in varying conditions such as loss of messages. Saunders et al. [5] examined autonomic cooperation strategies, including Ruler-based hierarchies and Message Board recruitment, and indicated the worth of defined leadership roles. McGuigan et al. [6] went further by suggesting adaptive communication strategies to enable coverage to be improved at a reduction in the energy costs. Collectively, these prototypes confirmed autonomic behaviours like self-configuration, self-healing and adaptive communication. Assessments were mostly restricted to demonstrations (e.g., leader failure, fault recovery, robustness message) and not sweeps over recruitment parameters.

Recent research has moved into the field of combining machine learning and optimization with bio-inspired concepts. Jiang and Guo [12] introduced guided policy search to complete decentralized swarm control, where robots learn scalable behaviors based on reinforcement cues, instead of fixed behaviors. Zheng et al. [11] proposed a visual attention based selective interaction model, in which only a few neighbors are communicated with by the robots resulting in emergent collective behaviors. Stigmergy was also found useful in coverage tasks in Wu et al. [14] who developed a pheromone-inspired swarm sensor network to monitor regions persistently. Self-reconfigurable hierarchical frameworks were presented by Zhang et al. [13] to extend swarm concepts to structural coordination problems.

These studies indicate the diversity of mechanisms presently available to swarm robotics. Na et al. [17] went on to show how Deep Reinforcement Learning (DRL) can allow bio-inspired collision avoidance, a combination of neural control with swarm-inspired approaches. The approach of Mohamed et al. [15] is to use swarm robotics in human-centered applications, in the form of a multirobot person-search system to track moving users. This shows the increasing interest in swarm methods in applied, safety-related fields. Also in another direction, Zhao and Li [16] came in with a two-stage multi-swarm Particle Swarm Optimizer (PSO) that enhanced global optimization. Although potent, such multi-swarm strategies are mostly algorithmic and fail to capture physical task allocation and communication trade-offs in real-world robot crowds.

Despite these developments, there is still a common weakness, which is the lack of systematic evaluation of recruitment strategies. The early prototypes defined significant autonomic mechanisms [4]-[6], whereas the recent works extended swarm intelligence to learning [11][12][17] and optimization [16]. Nonetheless, most of the works are either case-based or present algorithm innovations in isolation. Comparative, reproducible studies of recruitment strategies One-Responder, Multi-Responder, Blackboard across factorial parameter spaces are few. New important contributions are seen in the key factors including recruitment, Time-To-Live (TTL) and cluster join probability which have not been systematically investigated.

Initial works in the field of swarm robotics are also important background knowledge to the present paper. Barca and Sekercioglu [18] described the most important challenges and future trends of swarm robotics and found that many of them are still problems in today's systems, such as scalability, fault tolerance, and communication. One of the first conceptualisations of swarm robotics as a discipline was by Dorigo et al. [19], who characterised swarm robotics based on the idea of being inspired by social insects, with decentralisation, redundancy and emergence. Programmable self-assembly at this scale was demonstrated by Rubenstein et al. [20] who used over a thousand robots to show how local rules could produce collective behaviours on a large scale. This historical work demonstrates the historical interest in robustness and adaptability in swarms and now shows why autonomic extensions such as recruitment and stigmergy are needed to avoid stagnation and inefficiency in practical applications.

This paper fills that gap by developing MegaSwarm as a research-quality experimentation framework. In contrast to the previous prototypes [4]-[6] that demonstrated autonomic principles on case studies, we automate sweeps of recruitment tactics and parameters, with standardized measures (time to complete, distance, communication volume, coverage) and visual results that can be reproduced. This framework helps fill the gap between conceptual demonstrations and rigorous evaluation to allow recruitment trade-offs to be directly benchmarked. This would supplement the current state of the art learning and optimization research [11]-[17] by offering a controlled benchmark against which to compare and a repeatable methodology with which future research can be carried out.

III. Methodology

A. Simulator Basis:MegaSwarm and Extensions

MegaSwarm is the base of this work which is a modular simulator derived by extending SwarmSim2. MegaSwarm was built to illustrate autonomic computing concepts applied to swarm robotics, specifically: self-configuration, self-healing and adaptive communication [4]-[6]. The simulator at its core supports heterogeneous robots (Pambots), decentralized task sites, communication primitives and optional fault injections.

- Robot Roles (PambotType.cs): Robots may be Leaders, Messengers, Imagers or the Workers.
- Communication (Long RangeMessage.cs): Direct and long-range message passing.
- Task Sites (Site.cs, SiteInfo.cs): Sites are described by radius, completion score and optional special status.
- Configuration (SimConfig.cs): A centralized place of configuration of the experiment.
- Fault Injection (TimeBomb.cs): scripted failure is possible.
- Control Modes (ControlMode.cs): The modes are as follows- None, Centralised, Autonomic and Basic.

These factors provided MegaSwarm with versatility, however, it also left several features incomplete and underdeveloped. Recruitment was present as incomplete stubs (Message Board, Ruler), clustering was stored but not utilized and pheromone fields did not reinforce in a functional manner.

Redesigned MegaSwarm into an experimental framework suitable to be used in research and contributed to:

- Fully implemented recruitment strategies,
- Cluster-aware exploration,
- Zone-based delegation fallback,
- Pheromone reinforcement and gradient following,
- Automated batch sweeps across parameter spaces,
- Standardized metrics logging and visualization outputs.

Table I shows the development of the framework as compared to the original prototype.

TABLE I. COMPARISON OF MEGASWARM AND EXTENDED FRAMEWORK

Feature	***Original MegaSwarm Prototype***	***Extended Framework***
Recruitment strategies	Partial stubs (Message Board, Ruler)	Full implementation: Off, One-Responder, Multi-Responder, Blackboard
Recruitment parameters	Fixed TTL, no sweeps	Factorial sweeps across TTL, Need, cluster gains
Cluster awareness	Clustering metric logged (not used)	Probabilistic join model: distance, density, backlog
Zone delegation	Angular sector division only	Integrated fallback mechanism when recruitment fails
Pheromone field	Scaffold present (decay only)	Active deposits on move, work, entry; gradient following; visual PNG exports
Heatmap	Not exported	CSV + PNG heatmap with axes, site overlays
Metrics logging	Partial task stats only	Comprehensive: ticks, energy (distance), comms, coverage
Batch automation	Manual runs	Automated batch sweeps (BatchRecruitmentSweep.cs)
Visualization	Limited runtime visuals	Charts (time-series plots, normalized pheromone fields)

B. Agent Model:Pambot.cs

Each Pambot runs a sense-think-act cycle.

- Exploration: Robots explore incomplete sites using a site picking strategy: Nearest, Farthest or Balanced.
- Participation in recruitment: Recruitment posts are screened by robots through a cluster-aware probability model (see Eq. 1).
- Zone Delegation: In case no recruitment is appropriate, robots resort to the zone-based allocation (eight angular sectors).
- Interaction of pheromones: Robots lay pheromones as they move, enter a site, or when working and attract wandering by following the pheromone gradients.
- Work: Robots can use TryPerformWorkAt() when in a site radius to add to its score by Work (), Contribute () or Claim (). Individual metrics are also monitored by robots, including communication events, distance travel, analysis and imaging units, and task contributions.

C. Automated Experimentation:BatchRecruitmentSweep.cs

To provide reproducibility we created a batch runner to automate parameter sweeps. In each episode, a configuration is defined.

- Strategies: {Off, One, Multi, Blackboard}
- Need: {1, 2, 3}
- TTL: {200, 600, 900}
- Pull radius: {4.0, 6.0}
- Base: {0.10, 0.20}
- Gain: {0.30, 0.45}

Each configuration was run as an "episode," with metrics appended to metrics.csv and plots exported. This ensured reproducibility and scalability. Automated runner minimised the human error by eliminating the manual reconfiguration between the runs, as well as allowing the consistent output formats to be compared directly. What is more, the existence of timestamped exports guaranteed that the results of every run could be tracked and were not overlapping, which increases data integrity. BatchRecruitmentSweep allowed to explore trade-offs in communication, energy use, and task completion systematically, as factorial sweeps were supported across parameters.

Table II shows capabilities between Prototype and Extended work.

TABLE II. CAPABILITIES BETWEEN PROTOTYPE AND EXTENSION

Capability	***Prototype Available***	***Extended Work***	***Notes***
Heterogeneous roles	Yes	Yes (Workers + Messenger focus)	Recruitment tested with workers; Blackboard used messenger comms
Long-range messages	Yes	Yes	Extended for Blackboard recruitment posts
Fault injection (TimeBomb)	Yes	Not used	Left for resilience experiments
Site specialisation	Yes	Yes	Special sites included in metrics
Zone delegation	Stub only	Fully implemented	8-zone fallback when recruitment fails
Recruitment	Partial stubs	Full implemented	Factorial sweeps automated
Pheromone field	Scaffold only	Fully implemented	Deposits + gradient following + visualization

D. Equations for Extension

To transform MegaSwarm into a research-grade framework several extensions were introduced, 1.Recruitment-probability:

$$P_{\text{join}} = B + G \cdot D \cdot \rho \cdot \beta \quad (1)$$

- P_join: The probability that a robot will respond to a recruitment post. This governs whether robots join a site when help is requested.
- B (Base Join Probability): A constant baseline chance of joining, even if other conditions are not ideal. Ensures recruitment is never completely ignored.
- G (Cluster Gain): Scales the impact of distance, density, and backlog. Higher G values encourage more aggressive joining.
- D (Distance Term): A measure of how close the robot is to the recruitment site. Robots within the Cluster Pull Radius have D=1; further away, D decreases linearly toward 0 at twice the radius. This ensures nearby robots respond first.
- ρ (LocalDensityFactor): 1/ (1+local density) . If many robots are already nearby, ρ\Rho shrinks, discouraging further responders. This prevents over-saturation.
- β (Backlog Term): Adjusts probability based on unfinished sites. If many sites remain, β<1, reducing clustering at one location. If few remain, β≈1, allowing stronger cooperation.

Eq. (1) operationalises the biological principle of "selective recruitment" — robots balance *urgency* (distance), *utility* (density), and *global workload* (backlog) rather than blindly responding.

2.Clustering-index:

$$C = 1 - \min\left(1, \frac{\frac{1}{N}\sum_{i=1}^{N}\sqrt{(x_i - \bar{x})^2 + (y_i - \bar{y})^2}}{\text{NormRadius}}\right) \quad (2)$$

- C: A normalized clustering index, ranging 0–1.
- C→1: swarm is tightly clustered.
- C→0: swarm is dispersed.
- N: Number of active robots.
- (xi, yi): Position of the ith robot.
- (x¯, y¯): Centroid (average swarm position).
- Numerator (mean distance to centroid): Measures how spread-out robots are.
- Denominator (NormRadius): Normalization factor (typically half the map size), scaling distances into 0–1 range.
- Outerformula 1−min(...): Converts "spread" into "clustering." High spread lowers C, high density raises C.

This measure gives the simulator an autonomic awareness of swarm dispersion. It was logged in the prototype, but the extension makes it actionable.

3. Pheromone Field Update:

$$\Phi t{+}1(x,y) = (\Phi t(x,y) + \Delta\Phi(x,y)) \cdot \lambda \quad (3)$$

- Φt(x,y): Current pheromone intensity at grid cell (x,y)(x,y)(x,y).
- ΔΦ(x,y): New pheromone deposit at time ttt. Scaled by robot action:
  - Work deposit (largest reinforcement),
  - Enter deposit (medium reinforcement),
  - Move deposit (small breadcrumb trail).
- λ: Decay constant (set at 0.995). Each tick retains 99.5% of pheromone, simulating natural evaporation.
- Φt+1(x,y): Updated pheromone level for next timestep.

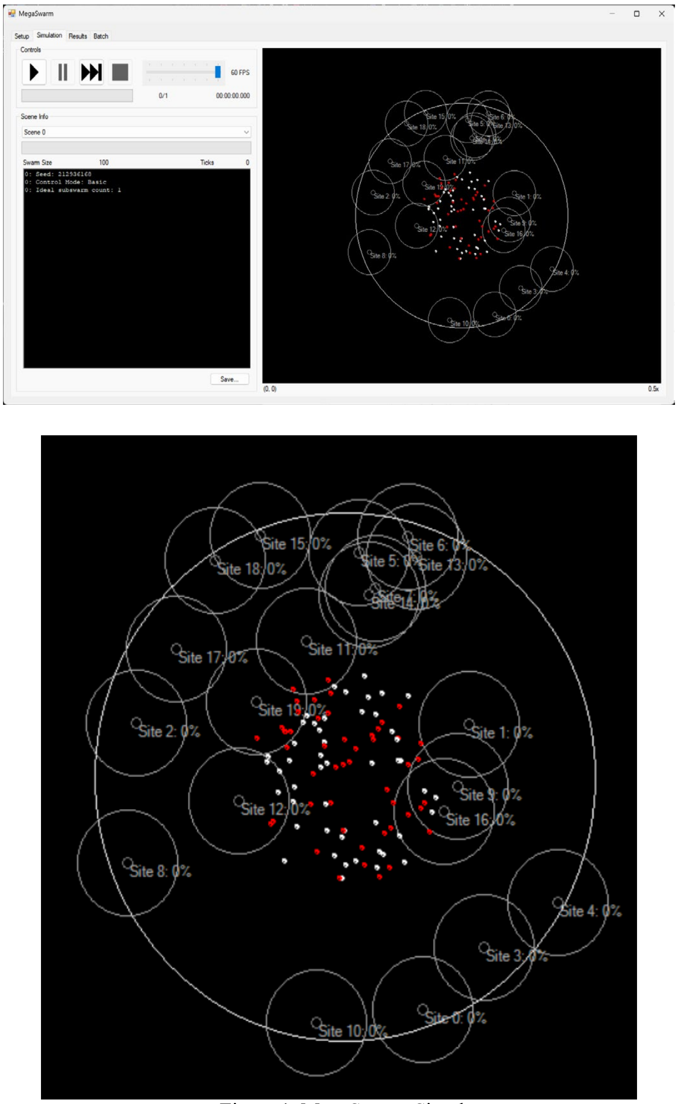


Figure 1. MegaSwarm Simulator

Courtesy of IARIA Board and IARIA Press. Original source: ThinkMind Digital Library https://www.thinkmind.org

 ISBN: 978-1-68558-356-9

Figure 1 shows the whole set up of the extended MegaSwarm Simulator.

E. Experimental Environment and Risk Management

The experimental environment for this project was based on the extended MegaSwarm simulator, a C# framework derived from SwarmSim2 and significantly modified to support recruitment strategies and reproducible evaluation. Simulations were executed on a Windows 11 system with Intel i7 processor, 16 GB RAM, and .NET runtime, providing sufficient resources to run swarms of 100 robots for up to 25,000 ticks. Each scenario involved 20 task sites arranged in balanced layouts, and recruitment modes were systematically varied across Off, One-Responder, Multi-Responder, and Blackboard strategies. The BatchRecruitmentSweep module enabled factorial sweeps of key parameters including Need, TTL, base join probability, and clustering gains, ensuring coverage of diverse experimental conditions. Outputs included CSV logs (sites completed, mean site score, communication pulses, communication units, and distance travelled), time-series charts (sites visited, active robots, mean site score), and spatial visualizations (heatmaps of robot visits and pheromone intensity maps). Random seeds were fixed for reproducibility, while export timestamps ensured traceability of every run. Together, this environment offered both quantitative metrics and qualitative insights into swarm behaviour, allowing robust comparisons across recruitment strategies.

Risk management was integral to ensuring reliability and reproducibility throughout experimentation. Software risks are from extending core files (SceneManager.cs, Pambot.cs, BatchRecruitmentSweep.cs), where minor errors could destabilize simulations; these were mitigated through incremental development, debugging with small test runs, and maintaining version-controlled backups. Data integrity risks were addressed by configuring the simulator to automatically export results into timestamped directories, preserving every run and preventing overwriting. This strategy also reduced the risk of selective reporting, as all metrics were consistently captured. Experimental risks related to limited scenario variation were acknowledged: only balanced site layouts and fixed swarm sizes were tested, which could bias conclusions. To mitigate this, the framework retains flexibility for clustered maps and larger swarms, to be explored in future work. Ethical risks were minimal given that experiments were fully simulation-based, but integrity was safeguarded by transparent logging and cross-verification of visual and numeric outputs. Collectively, these measures ensured that the results reported reflect true system behaviour, that no data was lost or corrupted, and that the experimental framework remains extensible for future research.

IV. RESULTS

A. Overview of Evaluation

The long version of MegaSwarm simulator was run with a swarm of 100 robots, a simulation length of 25 000 ticks and 20 task sites in a balanced set-up. The results are presented in four recruitment strategies, which include Off, One-Responder, Multi-Responder, and Blackboard. They are site coverage, site scores, swarm activity, spatial distribution, communication and energy expenditure.

B. Task Completion and Site Coverage

Figure 2 shows temporal task performances. The results in Off mode stabilized at 1012 sites with site scores averaged at 0.5 0.6. An improvement to coverage was seen by One-Responder and Multi-Responder that reached 12-16 sites compared to Blackboard that reached 16-18 sites. The scores on site were also of a similar nature with Blackboard coming to 0.9 as compared to Off mode, which was 0.5. Robot activity (Figure 2c) was kept at 100, since there were no failures, but many of the robots were under utilised.

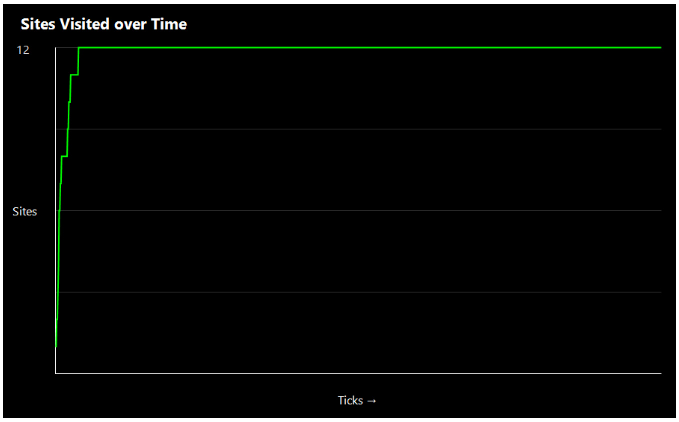


Figure 2(a). Sites Visited Over time

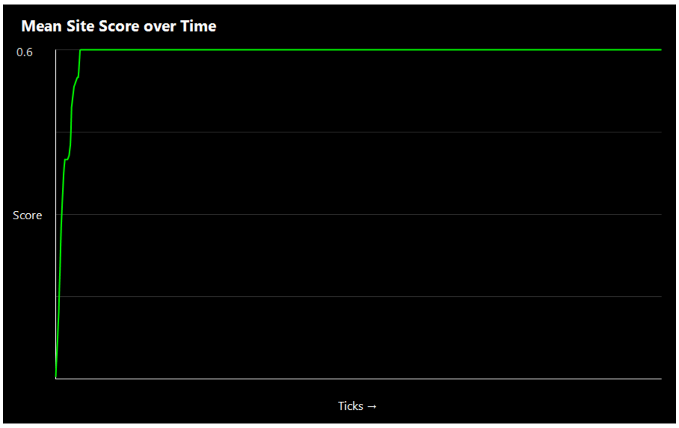


Figure 2(b). Mean Site Score Over time

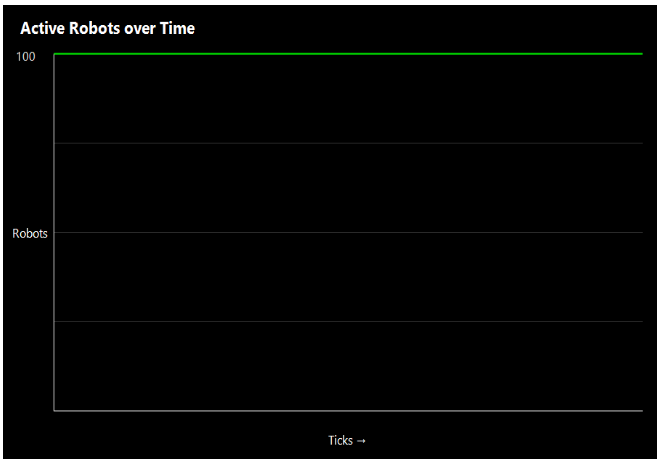


Figure 2(c). Active Robots Overtime

Copyright (c) IARIA, 2026. ISBN: 978-1-68558-356-9

C. Spatial Distribution of Exploration

Spatial exploration results are depicted in Figure 3. Heatmaps indicated that clustering was present in Off mode and areas that were not explored, Blackboard spread the exploration more evenly around the map. The structure of the fields of the intensity of the pheromones was less structured in Off mode, but stronger with better structure under Blackboard, indicating functioning stigmergy.

D. Communication and Energy Metrics.

Communication and energy statistics were taken out of metrics.csv. In Table III, average results are compared based on four strategies of recruitment. High movement and poor coverage were produced by Off mode. One-Responder performed slightly better but multi-responder performed evenly, and Blackboard performed the best in terms of coverage and efficiency at the expense of communication.

E. Efficiency Analysis

Efficiency was defined as sites completed per unit distance travelled. As shown in Table III, efficiency improved monotonically across strategies: Off delivered the lowest (≈2.0–2.4×10⁻⁶), while Blackboard achieved the highest (≈3.4–3.8×10⁻⁶). Recruitment strategies thus converted robot movement into more productive outcomes. This development indicates one of the most important discoveries: not only is coverage enhanced through recruitment, but energy usage is optimised. Without recruitment, robots moved a lot and left several sites unfinished which wasted the distance. In comparison, the recruitment strategies focused robots on active sites, which made the energy used in movement to play a direct role in site advancement. The Blackboard model especially increased efficiency through minimising redundancy and allowing a group to converge on incomplete tasks. In this way, efficiency analysis is a useful measure of differentiating between strategies, showing that the improvement in task completion was not brought about by increased energy use in isolation but by intelligent coordination of the robot activity.

F. Interpretation of Findings

From these results, several trends are clear:

- Recruitment improves task coverage and site completion
- Swarm utilization improves under cooperation, reducing idle robots.
- Spatial distribution becomes more uniform with recruitment, as shown in heatmaps.
- Communication costs increase progressively, with Blackboard requiring the most.
- Efficiency gains outweigh costs, as Blackboard yields more sites per unit energy despite higher comms.

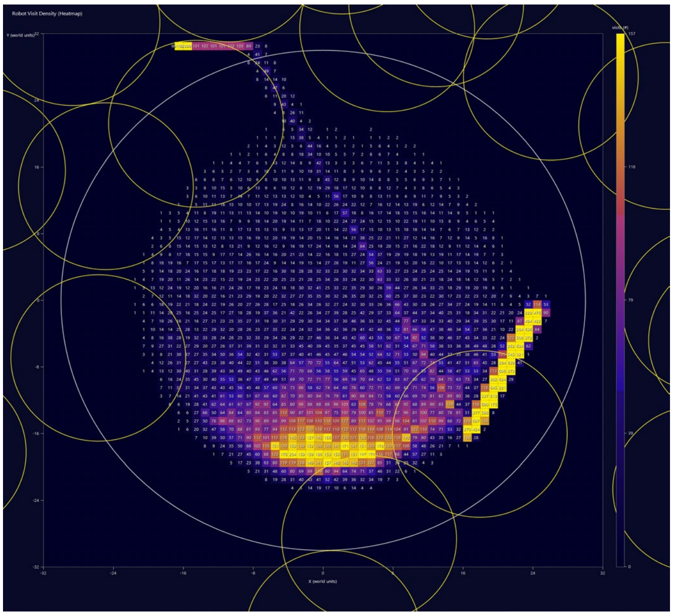


Figure 3(a). Heatmap of Robot Visits (Yellow being highest)

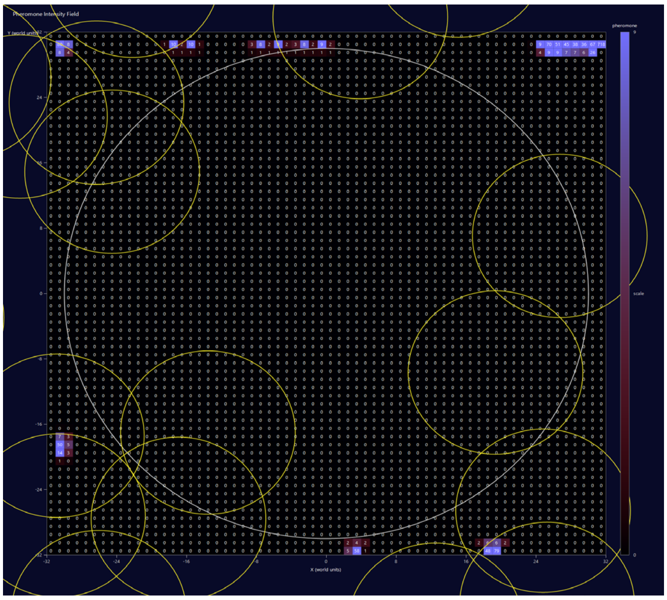


Figure 3(b). Pheromone Intensity Field

TABLE III. COMPARISON OF RECRUITMENT STRATEGIES

Strategy	Sites Comp.	Mean Site Score	Comm Units (scale)	Dist.	Efficiency (×10⁻⁶)
Off (No recruitment)	10–12 / 20	~0.5–0.6	~2.3×10⁸	~5.0M	2.0–2.4
One-Responder	12–14 / 20	~0.6–0.7	~5×10⁸	~4.9M	2.4–2.8
Multi-Responder	14–16 / 20	~0.7–0.8	~8×10⁸	~4.8M	2.9–3.3
Blackboard	16–18 / 20	~0.8–0.9	~1.3×10⁹	~4.7M	3.4–3.8

V. Discussion

The long MegaSwarm trials proved that the recruitment strategies differed. In Off condition, the number of sites visited by swarms plateaued at ~10-12 sites visited and the mean score of sites visited per swarm stabilized around 0.5. This is the classical issue of uncoordinated search, a lot of robots end up exploring the same few locations and leave others unsearched. And yet the general efficiency was low, although the activity was great.

One-Responder recruitment slightly increased the number of sites visited and efficiency. Its structure, however, in which only one robot could respond to a recruitment post, was restrictive, since complex sites typically needed multiple contributors.

Multi-Responder recruitment has a more scalable balance and allowed multiple helpers to be deployed at the same time. This increased site coverage to 14-16 sites, increased mean site scores and decreased wasted effort. The communication load however increased correspondingly.

Lastly, Blackboard recruitment generated the best performance with 1618 sites visited and sites mean scores of close to 0.9. By enabling the persistent site posts to all robots, Blackboard has made sure that sites which needed attention were always reinforced. This reduced redundancy, swarm usage and provided the highest efficiency per unit distance travelled. The price of this was a much greater communication burden, assessed in Comm Units.

The comparison brings out a key point in swarm robotics; recruitment leads to better performance at a cost of communication. The minimal comm load generated by the Off mode was at the expense of poor performance. Blackboard offered near-complete coverage, but it needed the most comm volume. Multi-Responder was developed as a compromise between efficiency and overhead.

This is consistent with previous autonomic swarm prototypes [4][5], where the cooperation mechanisms enhanced the completion of the tasks but clogged communication resources. In contrast to these previous studies, the current framework provided a quantitative measure of this trade-off, which allowed them to be compared reliably across factorial sweeps.

Several extended functions of MegaSwarm specifically made these insights possible. The probability recruitment model (Eq. The second ((avg-id-r)) parameter, which permitted robots to join sites selectively in a balance between urgency and backlog. The clustering index (Eq. Adaptive strategies were possible because swarm dispersion could be measured (within the scale of the swarm). Above all, the automated BatchRecruitmentSweep allowed reproducibility, systematically sweeping Need, TTL and clustering gains, instead of running a handful of anecdotal runs.

Charts and illustrations made reading easier. Clustering inefficiencies were indicated by heatmaps before Off mode, and the stigmergy was activated by pheromone maps during recruitment. In absence of such exports, inefficiencies would have been contained in raw logs. In this way, methods choices--equations, tables, sweeps, and visual outputs--immediately led to greater interpretation.

A. Limitations

Although Blackboard was able to perform better, there are drawbacks that still occur. First, site completion did not reach 100 per cent in the existing parameter settings. The plateaus in site score indicate that the swarm will see diminishing returns after filling all the recruitment posts. Second, pheromone trails were not strong even during the recruitment. This implies that the force of deposits or decay rates must be further calibrated to produce more robust stigmergic behaviour. Third, communication measures showed negative Comm Units on logs, which is an anomaly of the existing logging system. Relative comparisons are valid but absolute comm costs are to be refined.

The last restriction was the experiments were restricted to balanced site layouts and swarm size = 100. These are sources of consistency, but actual robust claims will have to investigate the clustering of maps and the varying swarm size. Additional sweeps are possible using the framework, but they are a topic of future work.

The results, notwithstanding some implications, can be highlighted. First, recruitment is necessary to scale swarms to non-trivial tasks environments. Lack of coordination means that robots flood and waste resources. Second, communication cannot be neglected: increasing the coverage means a penalty to the network load. Designers of actual robot swarms must trade off radio power, bandwidth and coordination. Third, reproducibility of the framework is important. This work shows how systematic technique can scale up swarm robotics to the point of publishable science by automating sweeps and exporting ready plots.

B. Future Work

In future, three areas should be worked on. First, parameter optimisation: It is possible that optimising pheromone decay and deposit rates could produce stronger stigmergy. Second, scenario diversity: the experiments will be extended to clustered maps and to different swarm sizes as a test of scalability. Third, Fault resilience: TimeBomb modules tolerate failure of individual robots; this will test whether recruitment also improves fault toleration.

In conclusion, the Discussion validates that the recruitment methods enhance the level of swarm performance significantly beyond uncoordinated searching and that Blackboard provides the greatest coverage at the expense of communication overhead. These results confirm the methodological extensions made in MegaSwarm including recruitment sweeps, clustering analysis and visualization. The systematic measurement of trade-offs fills these gaps and provides a reproducible method of autonomic swarm evaluation.

VI. Conclusion

This project aimed to develop a reproducible research framework that expands the MegaSwarm simulator beyond a prototype into a reproducible research framework on which recruitment strategies in swarm robotics can be systematically evaluated. Previous prototypes did not support complete recruitment capabilities, provided limited measures and the results were not repeatable in different scenarios. The extended framework filled these gaps by introducing four recruitment policies (Off, One-Responder, Multi-Responder, Blackboard),

 ISBN: 978-1-68558-356-9

offering cluster-aware exploration via probabilistic joining model, and making use of pheromone-based stigmergy by depositing pheromones on movement, entry and work. A BatchRecruitmentSweep module was automated to provide factorial parameter sweeping and ensure consistency and comparability across experiments. Moreover, visualization capabilities were introduced, such as heatmaps and maps of the robot visits, pheromone concentration, and time-series plots in addition to detailed CSV logs that track communication, distance, and task coverage. These extensions made MegaSwarm a transparent and research-ready appraisal tool.

Findings showed that recruitment did make a significant difference in swarm performance over uncoordinated exploration. Off mode flattened at 10-12 sites with an average of about 0.5, which indicates unproductive effort and premature stagnation. The One-Responder was slightly better, but the allocation of single helpers constrained its ability, whereas multi-responder was able to cover 14-16 sites with improved efficiency. Blackboard had consistently had the highest coverage (16-18 sites) and mean scores of close to 0.9 but it was also associated with the highest communication cost. These results demonstrate the trade-off that is inherent to communication and performance: the higher the cooperation, the better the coverage and efficiency but the more the communication overhead. Although there are gains to this, shortcomings persist, such as poor pheromone reinforcement, site completion stagnating below 100% and logging oddities in Comm Units. In addition, the experiments were restricted to uniform site layouts and swarm population of 100. Future research needs should be related to fine-tuning the pheromone parameters, to the application of clustered maps and larger swarms, and to the resilience of the robots to failures, using the TimeBomb module. In summary, expanded MegaSwarm system not only proves that recruitment is necessary to achieve scalable swarm performance, but also offers a reproducible way to quantify tradeoffs between communication, energy and coverage. This twofold contribution-methodological and empirical, is a step towards a more robust, autonomic multi-robot system.

ACKNOWLEDGMENT

This paper was supported by Ulster University's Artificial Intelligence Research Group (AIRG) and the School of Computing.

REFERENCES

[1] M. G. Hinchey, J. L. Rash, W. Truszkowski, C. Rouff, and R. Sterritt, "Autonomous and Autonomic Swarms", Paper presented at Autonomic & Autonomous Space Exploration Systems (A&A-SES-1) at International Conference on Software Engineering Research and Practice (SERP'05), pp 36-44, 2005.

[2] M. G. Hinchey, R. Sterritt, and C. Rouff, "Swarms and Swarm Intelligence", IEEE Computer, Vol. *40 No.* 4, pp 111-113, 2007. doi: 10.1109/MC.2007.144

[3] E. Vassev, R. Sterritt, C. Rouff and M. G. Hinchey, "Swarm Technology at NASA: Building Resilient Systems," in *IT Professional*, vol. 14, no. 2, pp. 36-42, March-April 2012, doi: 10.1109/MITP.2012.18.

[4] L. McGuigan, "Autonomic Pulse Communications for Adaptive Transmission Range and Data Transfer in Decentralised Robot Swarms", PhD Thesis, Ulster University, 2023.

[5] C. Saunders, R. Sterritt, and G. Wilkie, "Autonomic Cooperation Strategies for Robot Swarms," in Proc. IARIA International Conference on Adaptive Systems, pp. 20–27, 2016.

[6] L. McGuigan, R. Sterritt, G. Wilkie, & G. Hawe, "Decentralised Autonomic Self-Adaptation in a Foraging Robot Swarm", in International Journal On Advances in Intelligent Systems, vol. 15, no. 1&2, pp 12-23, 2022.

[7] P. Horn, "Autonomic computing: IBM perspective on the state of information technology", IBM T.J. Watson Labs, NY, 15th October 2001, Presented at AGENDA 2001, Scotsdale, AR.

[8] R. Sterritt, "Towards autonomic computing: effective event management," *27th Annual NASA Goddard/IEEE Software Engineering Workshop, 2002. Proceedings.*, Greenbelt, MD, USA, 2002, pp. 40-47, doi: 10.1109/SEW.2002.1199448.

[9] J. O. Kephart and D. M. Chess, "The vision of autonomic computing," in Computer, vol. 36, no. 1, pp. 41-50, Jan. 2003, doi: 10.1109/MC.2003.1160055.

[10] R. Sterritt, "Autonomic Computing", Innovations in Systems and Software Engineering, vol. 1, pp 79–88, 2005, doi: 10.1007/s11334-005-0001-5.

[11] Z. Zheng, Y. Zhou, Y. Xiang, X. Lei, and X. Peng, "Emergence of collective behaviors for swarm robotics through visual attention-based selective interaction," IEEE Robotics and Automation Letters, vol. 9, no. 11, pp. 9399–9406, Nov. 2024.

[12] C. Jiang and Y. Guo, "Multi-Robot Guided Policy Search for Learning Decentralized Swarm Control," in *IEEE Control Systems Letters*, vol. 5, no. 3, pp. 743-748, July 2021, doi: 10.1109/LCSYS.2020.3005441.

[13] Y. Zhang, et al., "Self-Reconfigurable Hierarchical Frameworks for Formation Control of Robot Swarms," in *IEEE Transactions on Cybernetics*, vol. 54, no. 1, pp. 87-100, Jan. 2024, doi: 10.1109/TCYB.2023.3237731.

[14] Y. Wu, M. Li, G. Li and Y. Savaria, "Persistence Region Monitor With a Pheromone-Inspired Robot Swarm Sensor Network," in *IEEE Internet of Things Journal*, vol. 9, no. 14, pp. 12093-12110, July 2022, doi: 10.1109/JIOT.2021.3133501.

[15] S. C. Mohamed, A. Fung and G. Nejat, "A Multirobot Person Search System for Finding Multiple Dynamic Users in Human-Centered Environments," in *IEEE Transactions on Cybernetics*, vol. 53, no. 1, pp. 628-640, Jan. 2023, doi: 10.1109/TCYB.2022.3166481.

[16] Q. Zhao and C. Li, "Two-Stage Multi-Swarm Particle Swarm Optimizer for Unconstrained and Constrained Global Optimization," in *IEEE Access*, vol. 8, pp. 124905-124927, 2020, doi: 10.1109/ACCESS.2020.3007743.

[17] S. Na, H. Niu, B. Lennox and F. Arvin, "Bio-Inspired Collision Avoidance in Swarm Systems via Deep Reinforcement Learning", in *IEEE Transactions on Vehicular Technology*, vol. 71, no. 3, pp. 2511-2526, March 2022, doi: 10.1109/TVT.2022.3145346.

[18] JC Barca, and YA Sekercioglu, "Swarm robotics reviewed", Robotica, vol. 31, no. 3, pp345-359. 2013, doi:10.1017/S026357471200032X.

[19] M. Dorigo, et al., "Evolving Self-Organizing Behaviors for a Swarm-Bot". Autonomous Robots vol. 17, pp 223–245, 2004, doi: 10.1023/B:AURO.0000033973.24945.f3

[20] M. Rubenstein, A. Cornejo, R. Nagpal, "Programmable self-assembly in a thousand-robot swarm." Science, vol. 345 no. 6198, pp 795-799, 2014, doi: 10.1126/science.1254295.

Agentic AI and the Autonomic Computing Vision: Continuity, Evolution, and Beyond

Roy Sterritt
School of Computing
Ulster University
Belfast, Northern Ireland.
e-mail: r.sterritt@ulster.ac.uk

***Abstract*— Autonomic Computing was introduced in the early 2000s to address the escalating complexity of software systems by promoting self-managing behaviors grounded in closed feedback loops such as Monitor–Analyze–Plan–Execute over a shared Knowledge base (MAPE-K). While influential, autonomic systems often failed to generalize beyond narrow domains due to brittle models, limited learning, and engineering challenges. Recent progress in machine learning — especially large language models (LLMs) and Agentic AI — offers renewed pathways to autonomy, with agents capable of goal-driven decision-making, planning, and tool integration. This paper argues that Agentic AI not only carries forward but extends the original vision of Autonomic Computing. Through comparative analysis spanning control loops, autonomy, adaptability, knowledge representation, governance, and human roles, we illustrate conceptual continuity and key differences. We conclude that Agentic AI realizes many autonomic aspirations while introducing new challenges in safety, alignment, and system governance. Our analysis serves as a roadmap for researchers at the intersection of systems engineering and modern AI.**

Keywords; Agentic AI, Autonomic Computing, Autonomic Systems, Self-Managing Systems, MAPE-K, Large Language Models, Autonomous Agents, Human-in-the-Loop Systems.

I. INTRODUCTION

Modern software systems — driven by cloud-native architectures, distributed services, and machine learning modules — face unprecedented complexity in configuration, adaptation, and operational reliability. Recognizing a similar systems-of-systems complexity challenge, Autonomic Computing was formulated to embed self-managing capabilities directly into system design, inspired by the biological autonomic nervous system's ability to regulate processes without conscious intervention. The seminal Autonomic Computing vision emphasized properties such as self-configuration, self-healing, self-optimization, and self-protection (self-CHOP), mediated through structured control loops and shared knowledge [1] [2].

Despite its conceptual appeal, autonomic systems often remained confined to narrow applications — for example, resource management or fault recovery — due to reliance on handcrafted rules and rigid models that could not easily adapt to open-ended operational environments [3].

In contrast, the rise of generative models and Agentic AI reflects a data-driven approach to autonomy. Agentic AI refers to systems composed of autonomous agents built on LLMs and supporting architectures that perceive, reason, plan, act, and adapt with minimal human intervention. These agents dynamically interpret goals, orchestrate tools, and adjust behavior based on feedback, suggesting a realization of autonomic aspirations through learned representations rather than static policies. Recent work defines Agentic AI as extending traditional AI with capabilities such as multi-step reasoning, context persistence, tool invocation, and collaborative multi-agent coordination [4].

Recent work has explicitly revisited the Autonomic Computing vision in light of large language models. Zhang et al. argue that LLMs may overcome several long-standing limitations of autonomic systems, including rigid modeling assumptions and limited semantic reasoning [5].

This paper addresses a gap in existing work by providing a comparative, architectural analysis that positions Agentic AI as a partial realization and extension of the Autonomic Computing vision, rather than as an independent paradigm. While prior research has examined autonomic systems and Agentic AI largely in isolation, this work systematically analyzes their conceptual continuity, architectural evolution, and governance implications. In doing so, we clarify the extent to which Agentic AI fulfills autonomic aspirations through learned representations and flexible reasoning, while also exposing new challenges.

The proposed perspective is not without limitations. The analysis is primarily conceptual and does not present empirical validation of Agentic AI–based autonomic systems. Moreover, although Agentic AI offers increased adaptability and expressiveness, it introduces trade-offs related to predictability, verification, operational cost, and governance that remain open research problems. Central to this discussion is the evolving role of humans — whether remaining in the loop or transitioning toward self-governing systems — raising critical questions of trust, safety, and accountability.

While this paper draws on prior autonomic computing research, including work by the author, the analysis deliberately integrates independent perspectives from recent Agentic AI and systems literature to avoid retrospective bias. The paper continues with section II on related work, then in separate sections on the Autonomic Computing vision (section III) and Agentic AI architectures and capabilities (section IV) before a comparative analysis in section V.

Challenges and open problems (section VI) are addressed before concluding the paper in section VII.

II. Background & Related Work

A. Autonomic Computing

The Autonomic Computing initiative, formulated by IBM and subsequent researchers, was motivated by the growing management burden of complex distributed systems. Its philosophy draws on the biological autonomic nervous system's self-regulatory mechanisms, advocating for software that manages itself under dynamic conditions with minimal human oversight. The MAPE-K reference architecture embodies this philosophy: monitoring exposes system state; analysis interprets data; planning formulates corrective actions; execution enacts changes; and all components share knowledge through a common repository [2]. Moreover, Sterritt and colleagues elaborated on self-managing systems, emphasizing biological metaphors and distributed agent technologies as enablers of autonomicity in computing [6].

Critiques of Autonomic Computing point to the difficulty of specifying exhaustive rules and models prior to deployment, leading to brittle behavior in the face of unforeseen inputs. Sterritt's work on event management and biologically-inspired self-protection mechanisms illustrates both the promise and the complexity of realizing autonomicity in practice [7].

Autonomic Computing has been extensively surveyed by Huebscher and McCann, who systematized autonomic architectures, control mechanisms, and application domains while also identifying limitations related to model accuracy, policy specification, and scalability [8].

B. Intelligent and Multi-Agent Systems

Research on intelligent agents and multi-agent systems predates modern AI agents, exploring goal-driven autonomous components with beliefs, desires, and intentions. These systems contribute foundational concepts such as decentralized control, negotiation protocols, and coordination strategies, often in symbolic representations. However, their scalability and adaptability were limited by expressiveness and computational complexity.

Early autonomic system designs increasingly adopted agent-based approaches. Tesauro et al. proposed a multi-agent architecture for autonomic computing in which distributed agents coordinate through learning and policy optimization to achieve self-managing behavior [9].

C. Large Language Models and Agentic AI

Recent work distinguishes between traditional AI agents and Agentic AI systems. In this paper, "Agentic AI" is used in the sense articulated by recent taxonomies [10], emphasizing persistent autonomy, tool use, and goal-directed reasoning.

Recent advancements in deep learning have produced foundation models capable of sophisticated reasoning, natural language understanding, and action planning when coupled with external tools. Agentic AI — characterized by persistent context, multi-step planning, and tool use — builds on this foundation. Frameworks and taxonomies for Agentic AI explore architectural designs, communication protocols, memory management, and safety guardrails [11]. Parallel work in autonomous system development demonstrates the potential of agentic approaches in domains such as computer vision and smart manufacturing [12].

The emergence of large-scale transformer-based language models demonstrated surprising few-shot generalization capabilities, enabling task adaptation without explicit retraining and fundamentally altering assumptions about machine reasoning [13].

The intersection of Autonomic Computing and modern foundation models has begun to attract attention. Zhang et al. examine whether LLMs can operationalize autonomic principles by enabling richer analysis, planning, and knowledge representation within self-managing systems [5].

III. The Autonomic Computing Vision

A. MAPE-K Control Loop

At the heart of Autonomic Computing is the MAPE-K loop (Figure 1), which formalizes the feedback mechanism enabling self-management. Systems continuously monitor internal and external conditions, analyze operational data for anomalies or optimization opportunities, plan appropriate responses based on policies, and execute adjustments — all coordinated through an evolving knowledge base [2]. Huebscher and McCann provide one of the cleanest formal explanations of MAPE-K variants and architectural trade-offs [8]. Distributed realizations of the MAPE-K loop were explored early on, with agent-based autonomic managers coordinating monitoring, analysis, and action through decentralized learning mechanisms [9].

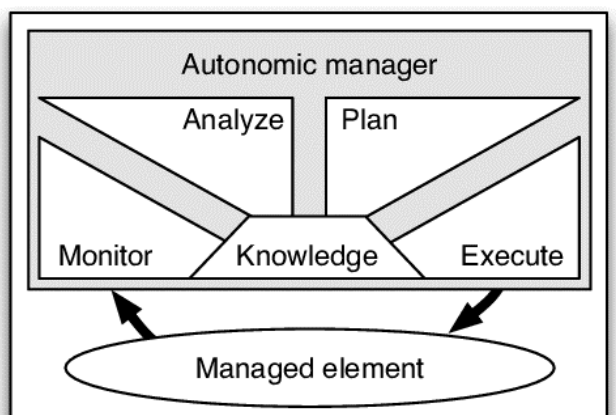


Figure 1. Autonomic Computing's *Autonomic Manager* with *MAPE-K* control loop [1].

B. Self-* Properties

Autonomic systems aspire to a set of self-properties — self-configuration, self-healing, self-optimization, self-protection — that collectively reduce dependence on manual intervention. These properties align with goals of increased

 ISBN: 978-1-68558-356-9

reliability, performance, and resilience. Sterritt al al.'s concise introduction highlights that achieving these properties necessitates intelligent interaction and richer models of system behavior [3]

Beyond self-configuration and self-healing, autonomic systems were envisioned to include self-protective behaviors inspired by biological processes. Sterritt and Hinchey introduced the notion of apoptosis and self-destruct mechanisms for autonomic agents, arguing that deliberate, pre-programmed, termination of compromised components can be a rational and protective system behavior [14][15] (aka Apoptotic Computing [16]).

Autonomic Computing research explored both distributed agent-based control [9] and biologically inspired safety mechanisms [6][7] underscoring that autonomy and governance were jointly considered from the outset.

C. Human Roles in Autonomic Systems

In Autonomic Computing, humans traditionally specify policies, provide oversight, and intervene when systems cannot resolve complex or ambiguous situations. The original vision aimed for "Human out-of- the-loop" but this evolved to acceptance of "Human-on-the-loop" (as opposed to heavy human involvement of Human-in-the-loop). The envisioned reduction in human workload was balanced with the recognition that expert knowledge remains essential for governance and high-level decision-making.

IV. Agentic AI Architectures and Capabilities

Conceptual taxonomies emphasize that Agentic AI extends beyond single-task agents by incorporating planning loops, memory mechanisms, and coordinated tool invocation within persistent goal structures [10]. According to an industry tutorial on agentic frameworks, modern agentic systems are often conceptualized as layered architectures with distinct perception, reasoning, and execution modules [17].

Where non–peer-reviewed sources are cited, they are used solely to illustrate practitioner perspectives and emerging architectural patterns rather than to substantiate technical claims.

A. Core Components of Agentic AI

Agentic AI systems involve autonomous agents that perceive environmental states, reason with context, plan action sequences, and act via integrated tools or APIs. Large language models serve as the core reasoning engine, offering flexible interpretation of goals and environment states. Memory and state mechanisms enable context persistence across iterative operational cycles [4].

B. Comparison with Autonomic Control Loops

While Autonomic Computing emphasizes explicitly engineered control loops, Agentic AI implements implicit feedback loops within reasoning and planning architectures. Agents repeatedly sense, reason, act, and update internal context, resulting in an adaptive cycle that mirrors MAPE-K functions but through learned interactions rather than static policies. Note though in Sharma's view, MAPE is explicitly within the Agentic AI architecture (Figure 2) [18].

C. Adaptation and Learning

Agentic systems boast broader adaptability, driven by data and probabilistic reasoning. Instead of pre-specified rules, agents learn representations and refine behavior through iterative interactions and feedback, enabling handling of unforeseen scenarios more effectively.

These capabilities build upon the few-shot and in-context learning properties first demonstrated at scale in transformer-based language models [13], which allow agents to adapt behavior based on prompt-level conditioning rather than static policy rules.

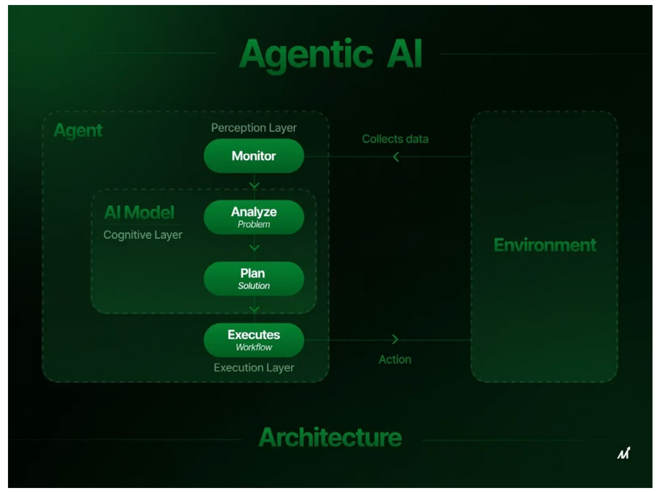


Figure 2. Agentic AI Architecture [18].

V. Comparative Analysis

A. Control Mechanisms

Autonomic systems rely on structured MAPE-K loops with predefined models and policies; Agentic AI uses dynamic reasoning loops where agents synthesize monitoring, analysis, planning, and execution in a unified cognitive cycle. Both enable self-management, but differ in flexibility and implementation. Agentic loops can adjust goal pursuit strategies on the fly.

Agentic AI can be viewed as a generalization of earlier agent-based autonomic systems. While Tesauro et al. employed multiple learning agents to realize autonomic behavior within well-defined objectives [9], contemporary agentic systems operate over far broader goal spaces and rely on foundation models rather than task-specific learning.

Unlike autonomic control loops, which operate over relatively stable and predictable time scales, Agentic AI introduces cognitive reasoning cycles whose latency and duration vary significantly, complicating coordination with low-level system controllers and human decision processes.

Importantly, this analogy should not be interpreted as functional equivalence: Agentic AI replaces explicit control-loop formalism with implicit, learned reasoning processes that sacrifice predictability for flexibility.

B. *Autonomy and Adaptability*

Autonomic Computing traditionally constrains autonomy within engineered policy bounds. Agentic AI offers broader autonomy, capable of flexible goal decomposition, multi-step planning, and coordination across agents, albeit with potential unpredictability. As noted in recent taxonomies [10], Agentic AI occupies a conceptual space between traditional rule-based agents and fully autonomous systems, complicating governance and accountability models.

C. *Knowledge Representation and Cognitive Control*

Autonomic systems depend on structured, curated knowledge bases; Agentic AI leverages distributed, learned representations derived from large datasets, offering rich semantic grounding but reduced interpretability.

By embedding semantic and contextual knowledge within learned representations, LLMs may address knowledge engineering challenges long identified in autonomic systems [5].

The Knowledge Plane proposed in autonomic and self-managing systems can be viewed as an early architectural precursor to the cognitive layers now instantiated by large language model–based Agentic AI.

In autonomic computing and autonomic communications, the Knowledge Plane was proposed as a logical layer responsible for global reasoning, policy interpretation, and contextual decision-making, distinct from low-level control mechanisms [19]-[21]. Research in this area explored both distributed agent-based control [9] and biologically inspired safety and self-protection mechanisms [14][15], reflecting an early recognition that autonomy and governance must be co-designed.

In contrast, contemporary Agentic AI systems embed reasoning, planning, and world modeling within a cognitive layer realized by large language models, effectively internalizing what was previously an explicit Knowledge Plane [13][10]. Recent work has explicitly argued that such LLM-based agents may finally operationalize aspects of the original autonomic vision, albeit with different epistemic assumptions and governance challenges [5][22].

D. *Governance and Safety*

Policy compliance is central in Autonomic Computing, with governance mechanisms enforcing boundaries. Agentic AI introduces complex safety and alignment challenges, requiring new frameworks for verification, trust, and accountability.

Early autonomic research explicitly addressed governance through controlled failure. The concept of autonomic agent apoptosis proposed by Sterritt and Hinchey treats self-destruction as a safety mechanism rather than a fault, a level of explicit governance largely absent from contemporary Agentic AI systems [14][15].

E. *Human in the Loop vs Self-Governing Systems*

Both paradigms aim to minimize manual intervention. Autonomic systems position humans as policy designers and overseers, while Agentic AI shifts humans toward goal specification, auditing, and high-level oversight, raising questions on trust, ethical constraints, and accountability.

While Agentic AI enables more flexible interpretation of high-level human intent, it also amplifies the risk of intent drift, where evolving goals, ambiguous prompts, or contextual shifts lead to behavior misaligned with original system objectives.

The comparative analyze of Autonomic Computing and Agentic AI is summarized in Table 1.

TABLE I. COMPARISON OF AUTONOMIC COMPUTING AND AGENTIC AI.

Dimension	Autonomic Computing	Agentic AI
Control Mechanism	Explicit MAPE-K loops	Implicit reasoning-action cycles
Knowledge Representation	Handcrafted models	Learned representations
Adaptability	Limited rule-based	Broad data-driven
Autonomy	Policy-bounded	Flexible and context-driven
Human Role	Policy author & overseer	Goal setter & auditor
Governance Challenges	Verification of rules	Safety, alignment, unpredictability

VI. CHALLENGES AND OPEN PROBLEMS

Key challenges include establishing safety guarantees for learned agent behavior, integrating formal verification with data-driven autonomy, designing scalable governance frameworks, and balancing agentic independence with human trust. Further research must bridge systems engineering rigor with AI interpretability and accountability.

This paper is intentionally conceptual and architectural in nature; systematic empirical evaluation of Agentic AI–based autonomic systems remains an important direction for future work.

A fundamental challenge introduced by Agentic AI is the shift from verifiable, policy-driven behavior to probabilistic reasoning that resists formal verification, thereby shifting assurance toward empirical validation and human oversight.

A further challenge arises from the reliance on LLM-based communication and goal processing within Agentic AI systems. Goals are typically expressed in natural language and interpreted probabilistically, which can lead to ambiguity, goal drift, or the emergence of conflicting objectives across agents or between system components. Unlike traditional autonomic systems, where policies and objectives are explicitly encoded and conflict resolution mechanisms can be formally specified, Agentic AI often lacks clear, verifiable procedures for detecting and resolving such conflicts.

In multi-agent settings, this challenge is amplified by indirect communication mediated through shared context or generated language, increasing the risk of misalignment and unintended coordination dynamics. While human oversight can mitigate some of these risks, it does not scale naturally to fully autonomous or high-speed environments. Addressing goal conflict detection, prioritization, and resolution in LLM-driven systems remains an open research problem central to the development of governable Agentic AI.

Revisiting autonomic concepts such as agent apoptosis [14][15] may provide valuable guidance for designing safe termination, rollback, or containment mechanisms for Agentic AI, particularly in long-running or self-directed systems.

In contrast to autonomic managers designed for continuous, low-overhead operation, Agentic AI systems incur non-trivial inference, energy, and economic costs, raising practical concerns about scalability and sustainability in always-on systems.

VII. Conclusion

Agentic AI realizes many core aspirations of Autonomic Computing by embedding adaptive reasoning cycles, learned representations, and flexible planning into system architectures. The deployment landscape for agentic systems is broad and rapidly evolving. Major technology companies — including Google, Microsoft, and Salesforce — are deploying agentic capabilities in production enterprise environments, from autonomous customer service and IT workflow agents to AI-driven coding and research assistants. Simultaneously, computer-use agents capable of interacting with open-ended GUI environments, and multi-agent orchestration frameworks enabling agent-to-agent collaboration, are introducing new categories of governance and safety challenges — particularly around unbounded action spaces, inter-agent monitoring, and the scalability concerns noted in Section VI.

Recent reflections by Sterritt reaffirm that autonomic computing remains a relevant conceptual foundation in the age of AI, while emphasizing the need to reconcile increased autonomy with robust governance and self-protection mechanisms [22], even framing such as "Autonomic (Apoptotic) Agentic AI" [23]. By framing Agentic AI as both a continuation and extension of the autonomic vision, this paper charts a path for future research that harmonizes the strengths of each paradigm.

Acknowledgment

Fitting with the Agentic AI spirit, the author acknowledges the use of large language model–based tools (specifically ChatGPT by OpenAI) to assist with early-stage drafting, structural organization, and language refinement of this manuscript. All technical content, interpretations, claims, and final revisions were performed and verified by the author, who takes full responsibility for the work.

References

[1] J. O. Kephart and D. M. Chess, "The Vision of Autonomic Computing," IEEE Computer, pp 41-50, 2003.

[2] R. Sterritt and M. Hinchey, "Autonomic computing - panacea or poppycock?," 12th IEEE International Conference and Workshops on the Engineering of Computer-Based Systems (ECBS'05), Greenbelt, MD, USA, 2005, pp. 535-539, doi: 10.1109/ECBS.2005.22.

[3] R. Sterritt, M. Parashar, H. Tianfield, and R. Unland, "A Concise Introduction to Autonomic Computing," Adv. Eng. Informatics, vol. 19, no. 3, pp. 181-187, 2005, doi: 10.1016/j.aei.2005.05.012.

[4] L. Ilves, M. Kilian, S. M. Parazzoli, T. C. Peixoto, and O. Velsberg, "The Agentic State: Rethinking Government for the Era of Agentic AI", Oct. 2025, https://agenticstate.org/paper.html (last access 27/02/2026).

[5] Z. Zhang et al., "The Vision of Autonomic Computing: Can LLMs Make It a Reality?" Microsoft Research, 2024.

[6] R. Sterritt and M. G. Hinchey, "From Here to Autonomicity: Self-Managing Agents and the Biological Metaphors that Inspire Them", Integrated Design & Process Technology Symposium (IDPT 2005), Beijing, China, pp. 143-150, 2005.

[7] R. Sterritt, "Towards autonomic computing: effective event management," 27th Annual NASA Goddard/IEEE Software Engineering Workshop, 2002. Proceedings., Greenbelt, MD, USA, pp. 40-47, 2002, doi: 10.1109/SEW.2002.1199448.

[8] M. C. Huebscher and J. A. McCann, "A Survey of Autonomic Computing," ACM Comput. Surveys, pp. 1-28, 2008.

[9] G. Tesauro et al., "A Multi-Agent Systems Approach to Autonomic Computing," AAMAS, pp. 464-471, 2004.

[10] R. Sapkota et al., "AI Agents vs. Agentic AI: A Conceptual Taxonomy, Applications and Challenges," arXiv, 2025.

[11] H. Derouiche, Z. Brahmi, and H. Mazeni, "Agentic AI Frameworks: Architectures, Protocols, and Design Challenges," arXiv, 2025.

[12] J. Kim et al., "Autonomous Computer Vision Development with Agentic AI," arXiv, 2025.

[13] T. Brown et al., "Language Models Are Few-Shot Learners," arXiv:2005.14165, 2020.

[14] R. Sterritt and M. Hinchey, "Apoptosis and Self-Destruct: A Contribution to Autonomic Agents?", In Formal Approaches to Agent-Based Systems, vol. LNCS 3228, pp 262-270, Springer, 2004, doi:10.1007/978-3-540-30960-4_18.

[15] R. Sterritt and M. Hinchey, "Biologically-Inspired Concepts for Autonomic Self-Protection in Multiagent Systems", In Safety and Security in Multiagent Systems vol. LNAI 4324, pp. 330-341, Springer, 2009, doi:10.1007/978-3-642-04879-1_22.

[16] R. Sterritt, "Apoptotic Computing: Programmed Death by Default for Computer-Based Systems", Computer, vol. 44, no. 1, pp 59-65, 2011, doi: 10.1109/MC.2011.5.

[17] P. Dini, "The Anatomy of an Agentic Framework," Tutorial presented at NetWare 2025, IARIA, 2025. https://www.iaria.org/conferences2025/filesNetWare25/Tutorial_PetreDini_TheAnatomyOfAgenticFramew.pdf (last access 28/2/2026)

[18] R. Sharma, "Agentic AI Architecture: A Deep Dive", Markovate, Nov. 2025, https://markovate.com/blog/agentic-ai-architecture/ (last access 27/02/2026).

[19] D. Clark, C. Partridge, J. C. Ramming, and J. T. Wroclawski, "A knowledge plane for the internet", Proc. Applications, technologies, architectures, and protocols for computer communication, Karlsruhe, ACM SIGCOMM, pp 3-10, 2003.

[20] R. Sterritt, M. Mulvenna, and A. Lawrynowicz, "Dynamic and contextualised behavioural knowledge in autonomic communications", In M. Smirnov (Ed.), Autonomic Communication, vol. LNCS 3457, pp. 217-228, Springer, 2004.

[21] R. Sterritt, D. W. Bustard, D. Gunning, and P. Henning, "Autonomic Communications and the Reflex Unified Fault Management Architecture", Advanced Engineering Informatics, vol. 19, no. 3, pp 189-198, 2005, doi:10.1016/j.aei.2005.05.015.

[22] R. Sterritt, "AI and Autonomic Computing", Editorial at The Twenty First International Conference on Autonomic and Autonomous Systems, Lisbon, Portugal, 2025, https://pure.ulster.ac.uk/en/publications/ai-and-autonomic-computing/ (last accessed 28/02/2026).

[23] R. Sterritt, "Autonomic Computing to Agentic AI", Presentation at Secure AI in Healthcare – Sandpit, 26th Nov. 2025, https://pure.ulster.ac.uk/en/activities/autonomic-computing-to-agentic-ai/ (last accessed 28/02/2026).

A Ground-Aware LiDAR Intensity Filter for Reliable SLAM in Challenging Reflection Environments

Federico Rollo[1,2], Valentina Pericu[1], Marco Roveri[2], Arash Ajoudani[3] and Navvab Kashiri[1]
[1] Robotics Laboratory, Innovation Hub & Intellectual Property, Leonardo S.p.A., Genoa, Italy, email: name.surname@leonardo.com
[2] Industrial Innovation, DISI, Università di Trento, Trento, Italy, email: name.surname@unitn.it
[3] HRI[2], Istituto Italiano di Tecnologia, Genoa, Italy, email: name.surname@iit.it

Abstract*—LiDAR intensity data has become increasingly important in robots odometry and Simultaneous Localization and Mapping (SLAM) systems, as it provides semantic information about the environment. It measures the returned energy level of the emitted rays, revealing material properties such as reflectance. However, the reliability of intensity values is compromised by various factors, including surface roughness, distance from the sensor, ray incidence angles, and variations in the transmitted energy from the Light Detection and Ranging (LiDAR) sensors. Additionally, glass surfaces such as windows and highly reflective objects, e.g., mirrors, can produce erroneous point detections, which can mislead Simultaneous Localization and Mapping (SLAM) algorithms. A common challenge arises with low incidence angles between the LiDAR sensor and the incidence surface, such as floors in indoor or semi-structured outdoor settings. These surfaces often produce points with low-intensity values regardless of their actual reflectivity, making them difficult to differentiate from other noise or incorrect measurements. We propose two implementations of a ground-aware LiDAR intensity filter to address these challenges. This filter is designed to mitigate the effects of misleading intensity values by filtering out incorrect detections from reflective surfaces while preserving points belonging to the ground. Experimental validation using state-of-the-art LiDAR odometry in reflective and complex environments demonstrated the necessity and effectiveness of our ground-aware intensity filter. The results indicate that the proposed filtering approach improves the robustness and reliability of odometry and SLAM, enabling them to operate successfully even in scenarios that would otherwise lead to failures or inaccurate mapping.

Keywords: ***LiDAR filtering, Intesity filtering, SLAM, Odometry***

I. Introduction

LiDAR intensity data has gained attention as a valuable resource for improving LiDAR odometry and SLAM systems. This intensity measurement, which represents the returned energy level of the emitted laser beams, provides additional information about the reflectance properties of objects' materials [1] (see Figure 1). For example, metallic surfaces typically produce high-intensity returns, whereas concrete surfaces exhibit medium- to low-intensity returns. These intensity measurements add semantic information to the point cloud data, complementing geometric features and enabling more robust data interpretation. For instance, Wang et al. [2] proposed the Intensity Scan Context (ISC), an extension of the Scan Context approach in [3] that integrates intensity information to enhance place recognition capabilities. Similarly, He et al. [4] incorporated intensity information into the Iterative Closest Point (ICP) algorithm in [5], achieving significant performance improvements in LiDAR odometry systems. Other works that

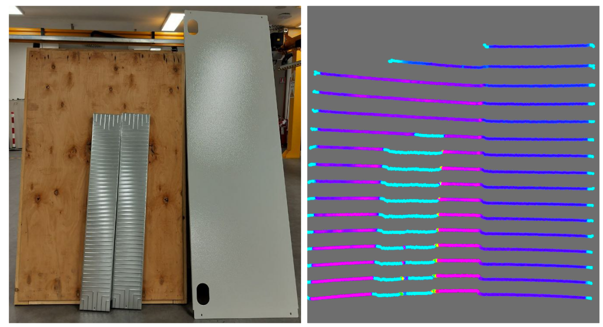


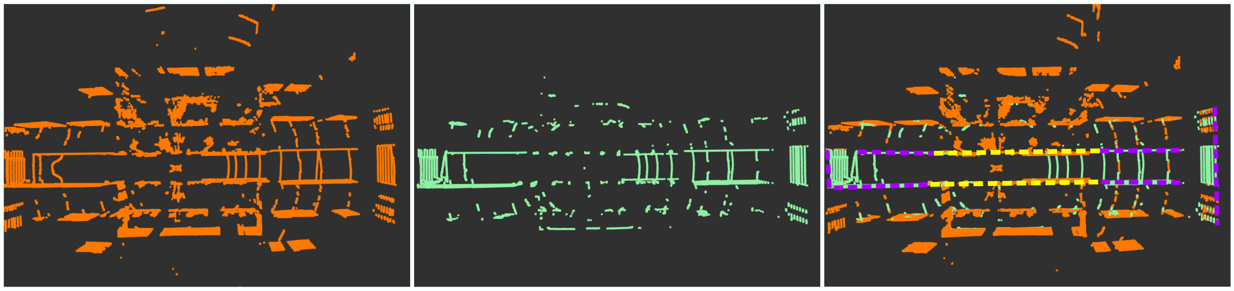
Figure 1. Comparison of LiDAR-perceived intensities across diverse materials. The intensity ranges from the lower values of shiny metallic surfaces, represented in light blue, through the medium values of opaque metallic surfaces in violet, to the higher values of the wooden panel in pink.

heavily rely on LiDAR intensity include those proposed by Guadagnino et al. [6] for sparse LiDAR odometry and by Wang et al. [7] for large-scale environment mapping. In both approaches, low-intensity points are either filtered as outliers or thresholded based on their magnitude. However, as elaborated later in this study, this practice can remove points of interest with low incidence angles θ, such as those on the ground (see the left image in Figure 4).

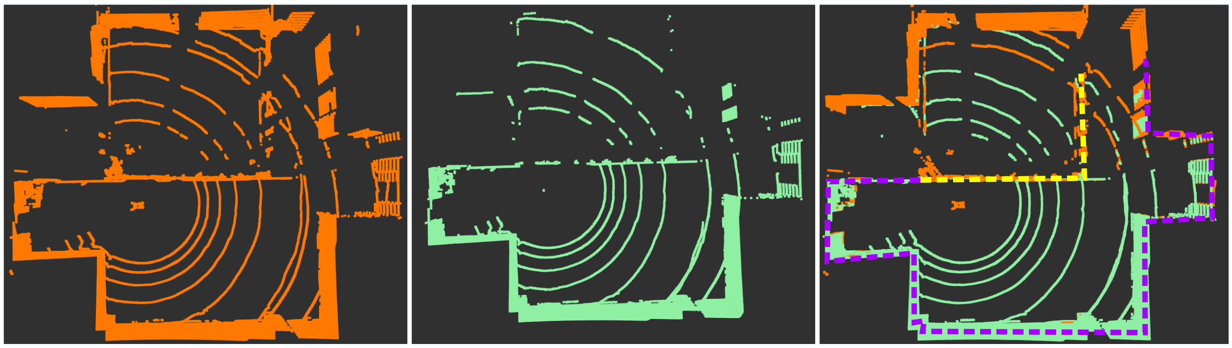
LiDAR intensity depends on the target surface's reflectance and roughness, as well as on factors such as distance, incidence angle, and the sensor's transmitted energy [1]. For example, metallic surfaces exhibit lower intensity values than wooden panels, as shown in Figure 1, which renders medium-high intensity. These differences are mainly due to variations in reflectance (opaque or lucid), surface roughness, and the orientation of each surface relative to the LiDAR scanner. Highly reflective surfaces (such as mirrors) and transparent surfaces (such as windows) are common in indoor environments. They often lead to erroneous detections in LiDAR point clouds, which can pose significant challenges for odometry and SLAM systems.

In indoor environments with windows, reflections create significant complications that cannot be ignored. As illustrated in Figure 2, the windows positioned along the yellow lines cause erroneous point detections in the original (orange) point cloud due to reflections. The ground truth map is approximately indicated by the purple line, while the filtered point cloud, generated using the method proposed in Section II, is shown in light green. In Figure 2a, the robot is navigating a narrow

 ISBN: 978-1-68558-356-9



(a) First filtering example with windows on both sides, in a narrow corridor.



(b) Second filtering example with windows only on one side, in a wide room.

Figure 2. Examples of the filtering effect are shown in two scenarios that show numerous erroneous reflections caused by the presence of windows. The original point cloud, collected from the robotic platform, is shown in orange, while the point cloud obtained using the ground-aware intensity filter appears in light green. The purple lines in the right images represent the structure of the environment/scenario, and the yellow lines indicate the positions of the windows. The overlap in the right image demonstrates that our filter successfully removes almost all incorrect reflections while preserving ground points.

corridor with windows on both sides. In Figure 2b, windows on only one side of the room create a mirroring effect that projects the room's reflection onto both sides of the robot. In these scenarios, using intensity information can be beneficial not only as an additional source of semantic data but also to filter out erroneous measurements caused by reflections and refractions.

Intensity-based filtering methods have been investigated in autonomous driving applications to remove noisy reflections from LiDAR point clouds caused by snow or rain, thereby enhancing detection accuracy and driving safety. One of the earliest examples is the work by Hui et al. [8], who used an intensity threshold to filter out noisy points originating from raindrops and snowflakes. More recent and sophisticated methods include Low-Intensity Outlier Removal (LIOR) by Park et al. [9], Dynamic Distance–Intensity Outlier Removal (DDIOR) by Wang et al. [10], and the latest Low-Intensity Dynamic Statistical Outlier Removal (LIDSOR) developed by Huang et al. [11].

These methods are specifically designed to address random noise caused by raindrops and snowflakes in autonomous driving scenarios. Although they rely on intensity measurements similar to our approach, they focus on the sporadic nature of weather-related noise. Consequently, when reflections are consistent, as in our case (see Figure 2), these filters fail to distinguish incorrect reflections from real environmental points. Moreover, they do not account for scenarios where erroneous reflections arise from other factors, such as the incidence angle between the LiDAR ray and reflective surfaces (see Figure 3). In indoor environments, for instance, ground points may exhibit low-intensity values due to their material reflectance and low incidence angles, which these methods do not address.

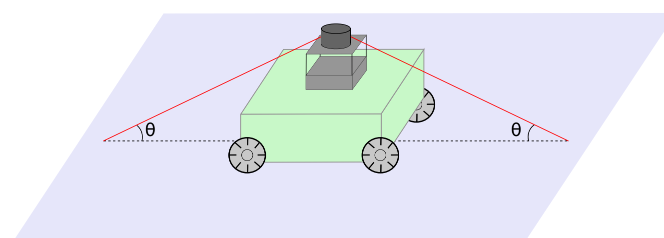


Figure 3. The incidence angle θ between the LiDAR beam and the reflective surface significantly impacts the returned energy level, often reducing the point intensity. Points affected in this way would typically be removed by standard threshold filtering methods.

Our solution addresses two key challenges: consistent erroneous reflections and low-intensity ground points. To tackle these issues, we present an efficient ground-aware intensity filter that enhances LiDAR-based systems while preserving the crucial information required for the final task.

Our main contributions are:

- A simple yet effective intensity-based filter that accounts for the robot's height to retain ground points with low reflectance (see Section II).

 ISBN: 978-1-68558-356-9

- A comprehensive evaluation and comparison of the filter's effectiveness using state-of-the-art odometry systems and a custom indoor dataset—featuring numerous glass doors—tested on a quadruped robot (see Sect. III).

This study underlines the need for integrating a lightweight yet effective intensity filter into LiDAR odometry or SLAM systems. Erroneous reflections frequently occur in both indoor (mirrors and windows) and outdoor urban (windows and raindrops) environments, adversely affecting LiDAR-based systems. The filter has been integrated into our LEO-SLAM [12] pipeline, proving its usefulness in highly reflective environments (see Figure 2). In Section II, we detail our approach and introduce two implementations of a lightweight, ground-aware intensity filter. Section III provides qualitative and quantitative evaluations, along with a detailed analysis of the results and their implications for LiDAR odometry and SLAM systems. Finally, Section IV presents our concluding remarks.

II. Ground-Aware Intensity filter

As mentioned in Section I, we propose two lightweight, ground-aware intensity filter implementations. The first, named the *naive intensity filter* in Section II-A, is simpler and faster, while the second implementation, referred to as the *normal intensity filter* in Section II-B, is more complex.

Both filters iterate through all the points ${}^l\mathbf{p}_i$ of the point cloud ${}^l\mathbf{P} = \{{}^l\mathbf{p}_1, {}^l\mathbf{p}_2, \ldots, {}^l\mathbf{p}_n\}$, expressed in the LiDAR frame l, and check whether their intensity ψ_i falls outside the specified boundary $\psi^* = [\psi^*_{min}, \psi^*_{max}]$, as expressed in (1). This equation defines the intensity range we aim to filter out:

$$\psi_i \notin \psi^* \equiv (\psi_i < \psi^*_{min}) \; \& \; (\psi_i > \psi^*_{max}) \quad (1)$$

However, as explained in Section I, points belonging to the ground often return low reflected energy due to the low incidence angle θ between the LiDAR ray and the ground surface. These ground points are typically filtered out by a standard thresholding filter, as seen in the left image of Figure 4, which can cause problems for LiDAR odometry and SLAM systems. Figure 4 demonstrates the benefits of incorporating ground awareness through a comparison between the standard intensity thresholding filter and one enhanced with ground awareness.

A. Naive Intensity Filter

As the name suggests, the first implementation, also known as the *naive intensity filter*, is a straightforward application of a bound filter with basic ground awareness.

To address the issue of ground filtering, we decided to retain all points located below the robot's footprint, i.e., the projection of the robot's base link onto the ground. This approach ensures that ground points are preserved, which provides valuable environmental information. However, it also retains incorrect reflections detected below the robot's height. While we argue that such false detections are minimal and insignificant compared to the other points being filtered out, it is important to acknowledge their presence.

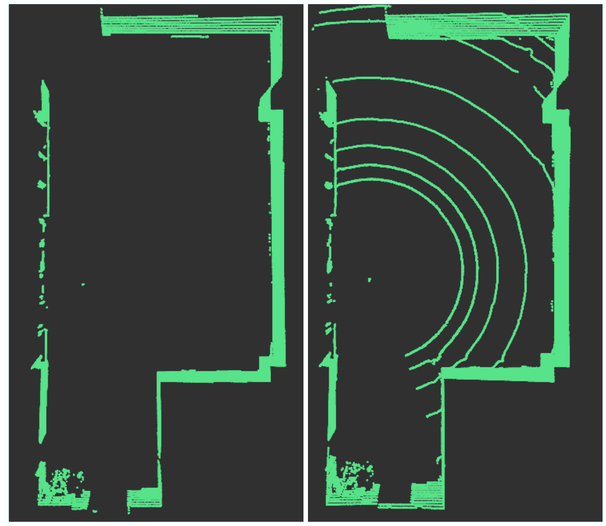


Figure 4. Comparison of intensity filtering: on the left, using only intensity thresholding; on the right, incorporating our ground awareness into the filtering process.

Considering ${}^l\mathbf{t}_f$ as the translation vector between the LiDAR frame l and the footprint f, we can use the third component, ${}^l z_f$, which translates from the LiDAR frame l to the footprint frame f along the z-axis, to check whether the points ${}^l\mathbf{p}_i$ in the LiDAR frame l lie below the footprint frame f.

Note that for fixed-wheeled robots, ${}^l\mathbf{t}_f$ is static and does not change during navigation. In our experimental setup, however, we used a quadruped robot, where ${}^l\mathbf{t}f$ can be computed using the positions of the feet, ${}^l\mathbf{t}f_i$, with f_i representing the pose of the feet in contact with the ground. Another important consideration is that we assume the LiDAR xy-plane is parallel to the base footprint xy-plane, meaning that only LiDAR-to-footprint translations are necessary. If this assumption is not valid, such as when the quadruped robot is tilted or the LiDAR is positioned with pitch or roll angles, the full homogeneous transformation ${}^l\mathbf{H}_f$ should be used to roto-translate the LiDAR points into the footprint frame before checking their height. The details of these platform-dependent factors are beyond the scope of this paper and are left to the reader.

Taking into account the above considerations and assumptions, the final *naive intensity filter* equation is:

$${}^l\mathbf{P}^* = \{{}^l\mathbf{p}_i \in {}^l\mathbf{P} \mid (\psi_i \notin \psi^*) \; \& \; ({}^l z_{p_i} < {}^l z_f)\} \quad (2)$$

where ${}^l\mathbf{P}^*$ is the ground-aware intensity filtered point cloud, ${}^l z_{p_i}$ is the z component of point ${}^l\mathbf{p}_i$, "|" and "&" denote "*such that*" and "*conditional and*" respectively, and $\psi_i \notin \psi^*$ is defined in (1).

B. Normal Intensity Filter

The second proposed solution is more sophisticated than the previous one; however, as reported in Section III, its benefits do not outweigh the issues it introduces. Similar to the previous approach, the normal filter checks whether the point intensity ψ_i of ${}^l\mathbf{p}i$ lies outside the boundaries

Copyright (c) IARIA, 2026. ISBN: 978-1-68558-356-9

$\psi^* = [\psi^*_{min}, \psi^*_{max}]$, and whether ${}^l\mathbf{p}_i$ belongs to the ground. However, the ground check is now more advanced, involving the computation of the normal vector ${}^l\boldsymbol{\eta}_i$ for each cloud point ${}^l\mathbf{p}_i$.

The *Normal Intensity Filter* consists of two steps:

1) Compute the normal vectors ${}^l\boldsymbol{\eta}_i$ for the points ${}^l\mathbf{p}_i$.
2) Check if the point's normal vector ${}^l\boldsymbol{n}_i$ is parallel to the z-axis of the robot's footprint frame f.

Norm computation: The normal vectors of the points are computed using the technique presented by Rusu et al. [13], with the implementation provided by the PCL library [14].

Parallel check: To check whether a normal vector ${}^l\boldsymbol{\eta}_i$ associated with a point ${}^l\mathbf{p}_i$ is parallel to the z-axis of the robot's footprint frame f, we first need to apply a roto-translation of the vector into the footprint frame as follows:

$$ {}^f\boldsymbol{\eta}_i = {}^l\mathbf{H}_f * {}^l\boldsymbol{\eta}_i \tag{3} $$

where ${}^l\mathbf{H}_f$ is the homogeneous transformation between the LiDAR and the footprint frame, and ${}^f\boldsymbol{\eta}_i$ is the normal vector expressed in the footprint frame.

Once we have expressed the normal vector in the footprint frame, we can calculate the angle between the vector ${}^f\boldsymbol{\eta}_i$ and the z-axis of the f frame, defined as ${}^f\mathbf{z} = [\mathbf{0},\ \mathbf{0},\ \mathbf{1}]^\mathbf{T}$. To compute these angles, we use the vector cosine similarity measure, θ_{η_i}, which is given by:

$$ \theta_{\eta_i} = \frac{{}^f\boldsymbol{\eta}_i \ \cdot \ {}^f\mathbf{z}}{\|{}^f\boldsymbol{\eta}_i\|\|{}^f\mathbf{z}\|} \tag{4} $$

where "·" denotes the dot product between ${}^f\boldsymbol{\eta}_i$ and ${}^f\mathbf{z}$, and $\|\boldsymbol{x}\|$ represents the vector *L2*-norm.

Using the cosine similarity θ_{η_i} in (4), we estimate whether the point ${}^l\mathbf{p}_i$, associated with its normal vector ${}^l\boldsymbol{\eta}_i$, lies on a surface parallel to the robot's base xy-plane. When ${}^f\boldsymbol{\eta}_i$ is parallel to ${}^f\mathbf{z}$, then $\theta_{\eta_i} = 0$. However, due to noisy measurements and small delays in sensor data collection, it is good practice to apply a minimal threshold ϵ_η and check whether θ_{η_i} falls below that value.

Given the above explanations, the final filter equation can be expressed as:

$$ {}^l\mathbf{P}^* = \{{}^l\mathbf{p}_i \ \in \ {}^l\mathbf{P} \mid \theta_{\eta_i} < \epsilon_\eta \} \tag{5} $$

III. Experiments

To validate our ground-aware intensity filter implementation, we used state-of-the-art LiDAR odometry algorithms such as KISS-ICP [15] and DLO [16]. Due to the point cloud sparsity in our scenario, we decided to remove one of the two downsampling filters used in KISS-ICP before performing scan matching, as proposed by the authors, in order to obtain better results.

We created a custom dataset in an indoor scenario with a high presence of windows, which led to numerous erroneous reflections detected by LiDAR. Our experimental setup consists of a quadruped robot, the Unitree B1, equipped with a Velodyne VLP-16 LiDAR. To filter out low-intensity points, we selected a filtered range $\psi^* = [0, 15]$ within the full intensity range $\psi = [0, 255]$. The terrain in the environment is mostly flat, with some ramps along the paths. We opted to create a custom dataset because, to the best of the authors' knowledge, commonly used public datasets such as the KITTI dataset [17] do not feature a significant number of erroneous reflections caused by intensity variations. As a result, these datasets are unsuitable for evaluating our proposed filtering methods.

Additionally, our dataset does not include a ground-truth trajectory that would allow for the calculation of Absolute Trajectory Error (ATE) or Relative Pose Error (RPE). However, for each path, we aligned the starting and finishing poses so that we could compute the final odometry error with minimal uncertainty. Using the initial and final robot poses, we evaluated the following three types of errors:

- Translation Error (TE) [m]: The Euclidean distance between the origins of the frames.
- Orientation Error (OE) [rad]: The rotation angle between the orientations of the frames, expressed in axis-angle form.
- Translation Vector Error (TVE) [m]: The translation error along the x, y, and z axes.

To validate the performance, we computed the Root Mean Square Error (RMSE) of the errors and, for completeness, compared the results across the following three scenarios:

1) Original LiDAR output, containing incorrect reflections.
2) Filtered LiDAR output, obtained by removing points with intensity values outside the threshold interval ψ^*.
3) Filtered LiDAR output with intensity filtering within ψ^*, augmented with additional ground awareness as described in Section II.

The RMSE results obtained in our experimental scenarios are reported in Table I. Specifically, we computed the Translation Vector Error (TVE) by subtracting the initial position ${}^B\mathbf{t}_i$ from the final position ${}^B\mathbf{t}_f$. The Translation Error (TE) is the Euclidean norm of TVE, while the Orientation Error (OE) was calculated by determining the relative quaternion orientation ${}^Bq_r = {}^B q_i^{-1} * {}^B q_f$ and converting the resulting angle into axis-angle form. Finally, we computed the RMSE using these error values for each scenario. It is important to note that we only used the *naive intensity filter* presented in Section II-A since the alternative solution proved to be unusable due to the high point cloud sparsity.

A. Discussion

The proposed intensity filter solution has proven to be robust and stable, offering improvements in state-of-the-art odometry systems. Below, we evaluate the experimental results (Section III-A1) and compare the two implementations (Section III-A2) to provide insights into which solution is more suitable.

1) Ground-Aware Filter Evaluation: The results reported in Table I demonstrate that our filtering method is beneficial and essential for the first framework evaluated, i.e., KISS-ICP [15]. In contrast, the second framework, i.e., DLO [16], proved to be robust enough to handle erroneous reflections, yielding comparable results in both unfiltered and ground-aware filtered scenarios.

TABLE I. THE ROOT MEAN SQUARE ERROR (RMSE) RESULTS FROM THE EXPERIMENTS ON OUR CUSTOM DATASET USING THE SELECTED FRAMEWORKS ARE PRESENTED, CONSIDERING THE FOLLOWING THREE SCENARIOS: (I) NO POINT CLOUD FILTER APPLIED, (II) APPLYING ONLY THE INTENSITY THRESHOLDING FILTER, AND (III) APPLYING THE INTENSITY THRESHOLDING FILTER WITH ADDITIONAL GROUND AWARENESS.

Framework	Filtering Mode	TE [m]	OE [rad]	TVE [m] (x y z)
Kiss-ICP [15]	No Filter	22.529	0.270	18.848 12.129 2.298
	Intensity	2.585	0.100	**0.278** 1.152 2.297
	Ground + Intensity	**1.112**	**0.075**	0.478 **0.986** **0.187**
DLO [16]	No Filter	**0.410**	0.068	0.164 **0.109** 0.359
	Intensity	2.395	0.098	0.179 0.408 1.979
	Ground + Intensity	0.428	**0.0637**	**0.156** 0.397 **0.031**

We observed that KISS-ICP was sensitive to erroneous reflections, which consistently impacted its performance. While introducing a standard intensity filter significantly improved the results, it still produced significant errors along the z-axis. This is because, as previously stated, the intensity filter removes ground points, which are crucial for the ICP algorithm to accurately recover errors along the z-axis.

An interesting observation is that in KISS-ICP, while the ground-aware filter performed better in almost all metrics, the x-axis component of the Translation Vector error (TVE) showed slightly higher errors compared to the solution using only intensity filtering. We found that this is due to ICP matching errors between ground-level range points in the current scan and those in the previous point cloud. LiDAR sensors emit infrared rays in rows (or channels), with each channel projecting a ring onto the ground, as shown in the right image of Figure 4. Since the point cloud is LiDAR-centric, these projected rings move along with the LiDAR, making them difficult to distinguish in consecutive scans, as the robot remains centred within them. As the robot typically moves along the x-axis of the LiDAR, the ICP algorithm tends to match the rings of two consecutive point clouds, introducing small errors along the x-axis. These errors are partially compensated by matching points on walls and obstacles, but the remaining errors accumulate over time, leading to slightly worse results when the ground points are preserved.

DLO, on the other hand, proved to be robust to erroneous reflections, yielding similar results in both the unfiltered and ground-aware scenarios. However, an interesting consideration arises: if odometry is used to create a point cloud map, the presence of erroneous reflections will affect the final results, resulting in an erroneous map. This is clearly demonstrated in Figure 2, where the comparison between the filtered and unfiltered point clouds in high-reflection environments shows that the unfiltered point cloud becomes essentially unusable. Additionally, when a standard intensity filter is applied to refine the map, the absence of ground points in the filtered point cloud negatively impacts the accuracy of ICP matching along the LiDAR z-axis, as evidenced in Table I, resulting in degraded performance in the final application. This highlights that, in mapping scenarios, our ground-aware filter is essential for producing an accurate final map without compromising odometry performance.

Lastly, an interesting observation made during the experiments was that DLO experienced odometry errors in the unfiltered scenario, whereas its performance was smoother in the ground-aware filtering scenario. DLO seems to be able to recover these accumulated errors, at least partially, by performing scan matching between the most recent point cloud scan and a piece of the entire accumulated map when places are already seen and revisited. This hypothesis could be further supported by using more advanced metrics, such as the ATE and RPE, throughout the entire trajectory. However, due to the absence of ground truth in our experiments (caused by the lack of motion capture systems), the authors are unable to fully validate this hypothesis and leave it open for future studies and deeper analysis.

2) Solutions Comparison: Although only one solution was tested, we provide a conceptual comparison of both proposed implementations, highlighting their respective pros and cons. The first solution (see Section II-A), referred to as the *naive solution* hereafter, is straightforward to implement. However, the main issue with this approach is that it retains all the erroneously reflected points below the robot's height. While many of these points may still correspond to the ground—facilitating their merging with actual ground points—this is not always the case.

The second solution (see Section II-B), known as the *normal solution*, is more advanced and can distinguish and retain additional surfaces by comparing the LiDAR's homogeneous transformation with the LiDAR norms. However, this comes at the cost of increased computational time and complexity. Furthermore, it still faces challenges with misreflected points that are parallel to the LiDAR's base. Additionally, based on our experience, the norm computation is not robust in cases of LiDAR sparsity, which is common in robots using LiDARs with fewer channels (e.g., 16 or 32). This limitation makes the *normal solution* less effective in such situations, a common scenario in robotic applications.

The *naive solution* has a mathematical complexity of $\mathcal{O}(n)$, as it processes each point only once. In contrast, the *normal solution* has a complexity of $\mathcal{O}(n^2)$, since it first computes the norm for each point (which takes $\mathcal{O}(n)$ time), and then checks both the intensity and the cosine of the angle between the LiDAR's z-axis vector and the point's norm (which adds another $\mathcal{O}(n)$ complexity).

In summary, the *naive solution* is faster, works well when the point cloud is sparse, and is easier to implement. On the other hand, the *normal solution* is computationally more expensive due to the norm computation. While it offers the advantage of maintaining xy-plane-parallel surfaces that are above the ground, this feature is less common in typical scenarios due to the generally low height of robots. However, such situations can still occur, making the *normal solution* beneficial in those specific cases.

We stated that the *naive solution* is generally preferred because delays are harmful in LiDAR odometry and SLAM systems. This choice becomes even more beneficial in outdoor environments, where the ground may not be parallel to the LiDAR, creating difficulties for the normal-based solution. Additionally, in scenarios where the robot is in a pit, the surrounding ground may be higher than the robot's height, which compromises the effectiveness of both solutions. However, there are two important considerations to note. First, when the ground is not parallel to the robot's base, the incidence angle of the LiDAR should be large enough to ensure the points fall outside the intensity threshold ψ^*. Secondly, in outdoor environments, the ground material is often opaque, which helps mitigate issues with incorrect reflections, especially when the incidence angle allows proper point detection.

To address the limitations of our solutions, a multi-sensor setup, such as cameras and LiDAR, could be employed. Alternatively, intensity and semantic information could be used to more accurately segment true ground points. However, in robots with constrained resources, using these methods would lead to higher power and computational demands, which could degrade the performance of odometry and SLAM. Therefore, simpler yet effective solutions, like the one presented in this paper, remain the preferred choice.

IV. Conclusion and Future Works

We proposed two different implementations, the naive and the normal, of a ground-aware intensity filtering method designed to remove points caused by erroneous reflections from semi-transparent or translucent surfaces, commonly found in indoor environments (e.g., windows). Each implementation offers distinct advantages and drawbacks depending on the environment. However, the simpler solution is preferred due to its faster performance and satisfactory results. Our evaluation of state-of-the-art LiDAR odometry systems confirmed that LiDAR-based systems significantly benefit from our simple yet effective ground-aware intensity filtering in environments with frequent erroneous reflections, enhancing both precision and robustness.

In future work, we aim to integrate this filter into semantic mapping frameworks, such as those discussed in [18], to further improve mapping performance and its subsequent applications [19], making them safer and more robust.

References

[1] A. G. Kashani, M. J. Olsen, C. E. Parrish, and N. Wilson, "A review of lidar radiometric processing: From ad hoc intensity correction to rigorous radiometric calibration", *Sensors*, vol. 15, no. 11, pp. 28 099–28 128, 2015.

[2] H. Wang, C. Wang, and L. Xie, "Intensity scan context: Coding intensity and geometry relations for loop closure detection", in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, 2020, pp. 2095–2101. DOI: 10.1109/ICRA40945.2020.9196764.

[3] G. Kim and A. Kim, "Scan context: Egocentric spatial descriptor for place recognition within 3d point cloud map", in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2018, pp. 4802–4809.

[4] L. He, W. Li, Y. Guan, and H. Zhang, "Igicp: Intensity and geometry enhanced lidar odometry", *IEEE Transactions on Intelligent Vehicles*, vol. 9, no. 1, pp. 541–554, 2023.

[5] P. J. Besl and N. D. McKay, "Method for registration of 3-d shapes", in *Sensor fusion IV: control paradigms and data structures*, Spie, vol. 1611, 1992, pp. 586–606.

[6] T. Guadagnino et al., "Fast sparse lidar odometry using self-supervised feature selection on intensity images", *IEEE Robotics and Automation Letters*, vol. 7, no. 3, pp. 7597–7604, 2022.

[7] H. Wang, C. Wang, and L. Xie, "Intensity-slam: Intensity assisted localization and mapping for large scale environment", *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 1715–1721, 2021.

[8] L. Hui, L. Di, H. Xianfeng, and L. Deren, "Laser intensity used in classification of lidar point cloud data", in *IGARSS 2008-2008 IEEE International Geoscience and Remote Sensing Symposium*, IEEE, vol. 2, 2008, pp. II–1140.

[9] J.-I. Park, J. Park, and K.-S. Kim, "Fast and accurate desnowing algorithm for lidar point clouds", *IEEE Access*, vol. 8, pp. 160 202–160 212, 2020.

[10] W. Wang et al., "A scalable and accurate de-snowing algorithm for lidar point clouds in winter", *Remote Sensing*, vol. 14, no. 6, p. 1468, 2022.

[11] H. Huang, X. Yan, J. Yang, Y. Cao, and X. Zhang, "Lidsor: A filter for removing rain and snow noise points from lidar point clouds in rainy and snowy weather", *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. 48, pp. 733–740, 2023.

[12] F. Rollo, V. Pericu, M. Roveri, A. Ajoudani, and N. Kashiri, "Leo-slam: A multi-level scan matching approach with submap-based loop closure detection", in *2025 European Conference on Mobile Robots (ECMR)*, IEEE, 2025, pp. 1–7.

[13] R. B. Rusu, "Semantic 3d object maps for everyday manipulation in human living environments", *KI-Künstliche Intelligenz*, vol. 24, pp. 345–348, 2010.

[14] "Point cloud library (pcl)". [Online]. Available: https://pointclouds.org/documentation/tutorials/normal_estimation.html.

[15] I. Vizzo et al., "KISS-ICP: In Defense of Point-to-Point ICP – Simple, Accurate, and Robust Registration If Done the Right Way", *IEEE Robotics and Automation Letters (RA-L)*, vol. 8, no. 2, pp. 1029–1036, 2023. DOI: 10.1109/LRA.2023.3236571.

[16] K. Chen, B. T. Lopez, A.-a. Agha-mohammadi, and A. Mehta, "Direct lidar odometry: Fast localization with dense point clouds", *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 2000–2007, 2022.

[17] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The kitti dataset", *The International Journal of Robotics Research*, vol. 32, no. 11, pp. 1231–1237, 2013.

[18] F. Rollo, G. Raiola, A. Zunino, N. Tsagarakis, and A. Ajoudani, "Artifacts mapping: Multi-modal semantic mapping for object detection and 3d localization", in *2023 European Conference on Mobile Robots (ECMR)*, IEEE, 2023, pp. 1–8.

[19] F. Rollo et al., "Semantic-based loco-manipulation for human-robot collaboration in industrial environments", in *European Robotics Forum*, Springer, 2024, pp. 55–59.

VaMAI-Validator: Agent Validation Platform for Autonomous Maritime Navigation

Christoph Baier, Joel Aschmann, Fulya Horozal, Marcel Dechert, Sebastian Scholze
Institute for Applied Systems Technology Bremen (ATB)
Wiener Straße 1
28359 Bremen, Germany
email: {baier, aschmann, horozal, dechert, scholze}@atb-bremen.de

***Abstract*—The development and validation of autonomous maritime navigation agents remain challenging, in part due to the lack of standardized, automated testing environments comparable to those in conventional software engineering. This paper introduces a novel modular validation platform, called the Validation of Maritime AI (VaMAI) Validator, that enables the simulation-based, automated validation of autonomous maritime navigation agents. The Validator provides a general-purpose, unified environment for evaluating agent performance, safety, and regulatory compliance across standardized scenarios, rules, and metrics. Through a network interface and client libraries, the Validator supports the integration of multiple agents, thereby enabling cross-project comparability and benchmarking. Plugins enable developers to define custom scenarios, rules, and metrics, and share them with others. A graphical user interface offers playback and detailed analysis of simulation results to facilitate debugging and performance assessment. The Validator can be deployed on-premises to ensure data privacy and operated on standard laptop hardware. Two agents and three proof-of-concept plugins were integrated, and validation of one agent using all plugins revealed behavioral flaws confirming the platform's diagnostic capability.**

Keywords-Automated validation; Simulation-based testing; Maritime autonomy; COLREGs compliance; Benchmarking.

I. Introduction

Productivity in software development is commonly enhanced by adhering to best practices such as test-driven development (TDD). Comprehensive test suites and automated validation pipelines have become de facto standards in modern software engineering. They enable developers to detect defects and ensure functional correctness within seconds after each code modification.

In contrast, achieving comparable testing rigor for autonomous maritime navigation agents remains challenging. Traditional unit tests provide limited value because agent performance depends on temporal behavior, environmental dynamics, and interactions with other vessels and obstacles. Verifying such systems requires evaluating decision-making over time in realistic maritime environments. Consequently, developers must rely on complex simulation setups to assess navigation scenarios. Establishing and maintaining such environments and validation procedures require substantial effort and are typically project-specific, which results in duplicated work and a lack of standardized benchmarks across projects. Furthermore, feature-rich debugging and visualization tools similar to Integrated Development Environments are often not available.

This work introduces the *VaMAI Validator*, a platform designed to improve the productivity of autonomous maritime navigation agent development. It enables automated, simulation-based validation. The term *VaMAI* originates from the German Federal Ministry for Economic Affairs and Energy (BMWE) project *Validation of Maritime AI*, within which the Validator was developed. The Validator provides a unified platform for evaluating agents with respect to performance, safety, and regulatory compliance. Its modular architecture supports integration of agents and validation components. These enable standardized, reproducible, and extensible testing workflows. The main features of the Validator are summarized as follows.

- Multi-Agent Support: The Validator supports the validation of multiple agents within a single environment. This capability is currently demonstrated using two C++ agents: a path-planning agent for autonomous maritime vehicles and a basic waypoint-following navigation agent.
- Client Library: A standalone C++ client library is provided to enable agent integration with the Validator, requiring only minimal implementation effort.
- Standardized Validation Interfaces: The Validator offers three interfaces for extending the validation process with scenarios, rules, and metrics. These interfaces follow a standardized format and can be used with different agents.
  Scenarios define navigation tasks, such as safely reaching a target position within a specified time.
  Rules observe vessel actions and report noncompliant behavior, for example with respect to the 1972 Convention on the International Regulations for Preventing Collisions at Sea (COLREG).
  Metrics observe vessel behavior and provide quantitative feedback in the form of numerical values, such as a collision risk index (CRI).
- Centralized Validation: The Validator is implemented as a single dedicated server that evaluates scenario simulations for all integrated agents, eliminating the need for redundant, agent-specific test suites.
- Extensibility: Plugin-based interfaces enable custom scenarios, rules, and metrics while supporting reuse across projects.
- Automation: Validation is triggered by a single command, after which the entire process is automated, streamlining developer workflows.

- Monitoring and Graphical Analysis: Simulation logs and validation reports provide detailed insights into agent behavior and are accessible through the built-in graphical user interface (GUI).

In the following sections, we review related work in Section II and present the architecture of the validation process in Section III. Sections IV and V describe our proof-of-concept (PoC) plugin implementations and the Validator GUI, respectively. Section VI discusses the results of our approach and its current limitations. Section VII summarizes our results and outlines directions for future improvements to the Validator software.

II. Related Work

Maritime autonomous surface ships (MASS) have attracted increasing research attention in recent years, driven by growing interest in autonomy and automation within the maritime domain. Many studies focus on the development of agents that operate safely and efficiently while complying with the COLREGs [1].

A. Simulations

A common approach for evaluating such agents is the use of marine traffic simulation systems (MTSS) [1]. These simulations are comparable to test suites in software engineering, as they provide systematic feedback on agent behavior and performance.

Existing platforms, such as the Open Simulation Platform [2], provide high physical accuracy and support co-simulation of multiple independently modeled vehicles. However, achieving this level of realism typically requires a complex setup and substantial user expertise.

In contrast, this work targets a more constrained MTSS that emphasizes rapid feedback and ease of use. The proposed Validator is designed specifically for agent behavior validation rather than high-fidelity physical modeling.

Tools such as ShipNaviSim [3] and ASVTrafficSim [4] are conceptually closer to our approach. However, they primarily rely on recorded traffic data and do not provide dedicated mechanisms for integrating custom validation metrics and rules. While these tools currently offer more detailed environmental modeling, they lack explicit support for extensible behavior validation.

The Validator addresses this gap by enabling the reuse and sharing of scenarios, metrics, and rules across multiple agents with minimal setup effort.

B. Validation and Verification

Research on verification and validation of systems with AI components highlights challenges related to formalizing specifications, incomplete data, and subjective human feedback [5]. Validating agents for autonomous maritime navigation presents similar issues, including "unstructured environments, coarsely specified traffic rules, and largely varying vessel types" [6]. Furthermore, "there is currently no standard for simulating interactive maritime environments to rigorously benchmark autonomous vessel algorithms" [6], highlighting the lack of standardized MTSS-based benchmarking. Existing studies typically validate performance in scenarios or rules that are loosely based on COLREG or good seamanship principles. Often, this validation is performed inconsistently.

C. Scenarios

Common handcrafted validation sets include COLREG encounter typologies (head-on, crossing, overtaking), the Imazu problem set, and extended scenario collections [7][8]. Some works propose unified frameworks for scenario design [9], but there is no consensus on scenario parameterization, which makes cross-agent comparisons difficult [6]. In addition, many navigation algorithms are validated using only a few manually defined scenarios, which raises concerns about coverage and generality [1].

D. Rules

Rule compliance is a critical facet of maritime navigation. The natural approach is to integrate COLREG compliance into the agent, widely regarded as the canonical ruleset for safe maritime navigation. Many works attempt COLREG adherence in agent logic [10][11]. However, the COLREGs pose significant challenges, as they contain inherent ambiguities that allow divergent algorithmic interpretations [12]. In addition, many agent implementations support only a subset of COLREGs.

There is a wealth of knowledge from maritime navigation experts, often referred to as good seamanship, encompassing practices and decision-making principles developed through experience at sea. While recognized in the IMO COLREGs (Rule 2), a formal definition and quantification of good seamanship remain open challenges [13]. Consequently, these expert practices are not yet systematically adopted in autonomous navigation.

The Validator rule plugin interface provides a platform for experimentation where COLREG and good seamanship logic can be implemented once and used by multiple agents, which helps to reduce integration costs and to enable more extensive scientific exploration and unification.

E. Metrics

A common approach to quantitatively assess vessel behavior is through metrics, such as CRIs [14]–[16]. However, these are defined inconsistently across studies, and differ with respect to weighting schemes, ship-domain shapes, and threshold definitions [14].

Incompatibilities between an agent developer's MTSS implementation and a published metrics library may require metrics to be reimplemented, which is a complex and time-consuming process. Our Validator resolves this issue by enabling users to implement a metric plugin once and reuse it across all integrated agents.

F. Agents

A wide range of agent implementations exists, including classical path-planning algorithms [17], as well as neural network-based, reinforcement learning, and hybrid agents [7]. These agent implementations are fragmented and developed across diverse programming environments (e.g., C++, Python, MATLAB), embedding agent-specific logic,

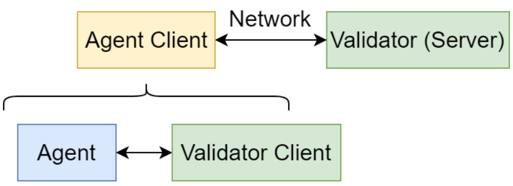


Figure 1. Components involved in validation.

APIs, and data pipelines. This heterogeneity complicates integration into a shared validation environment and hinders cross-system comparability.

Our Validator mitigates these issues by providing uniform interfaces and client libraries and simplifies the integration of different agents into a shared validation environment.

G. Summary

In summary, despite significant progress, to our knowledge, there is still no widely adopted, standardized, multi-agent-supporting validator that delivers automated, repeatable, and comparable behavioral validation results across different agents. The Validator addresses this gap in two ways. First, it provides a fully automated agent development platform that supports continuous integration workflows with simulations, detailed reports, and an integrated GUI for in-depth analysis. Second, it offers a modular architecture that counteracts widespread fragmentation by unifying scenarios, metrics, and rules within a single system that can be extended via plugins and integrate multiple agents.

III. System Architecture

This section outlines the overall architecture of the VaMAI Validator and describes its main components, their interactions, and the underlying setup.

A. Components

The Validator Server (referred to as *Validator* in this section) and the Agent Client are the two components that interact in the validation setup (see Figure 1). The Validator is provided as an executable together with a Validator Client library, which, combined with an agent implementation supplied by the agent developer, forms the Agent Client.

B. Workflow

Figure 2 illustrates the data workflow between the Agent Client and the Validator. The process begins when the agent developer wishes to test a new agent version. The agent developer uses the `runValidation` function of the Validator Client library, which starts the simulation and communication with the Validator. All subsequent steps are automated.

1) The Agent Client first requests scenarios from the Validator.
2) The Agent Client runs these scenarios in a time-stepped simulation in which the agent directs the vessel towards its target. Throughout the simulation, the Agent Client records the physical states of all vessels in logs and then sends them back to the Validator.
3) The Validator evaluates the logs and generates a report. Scenarios, logs, and reports are stored locally and made available to the agent developer for further analysis.

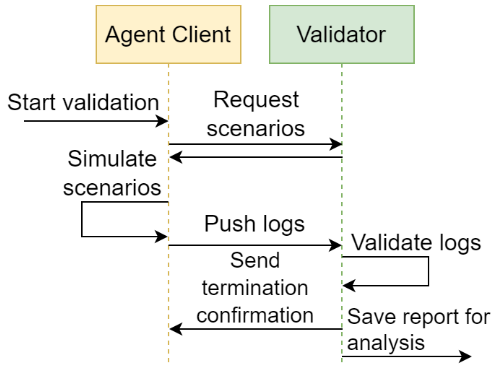


Figure 2. Validation Job Workflow. Displays the flow of communication between server and client during validation.

C. Architecture: Agent Client

Figure 3 shows the architecture of the Agent Client. All the agent developer needs to do is to implement the agent interface from the Validator Client library, which defines the `getNavigationRecommendation` function signature. This method receives the current simulation state, including the physical state data of vessels and obstacles (e.g., position, heading, velocity, and target position), and returns a navigation recommendation (i.e., target heading and target speed).

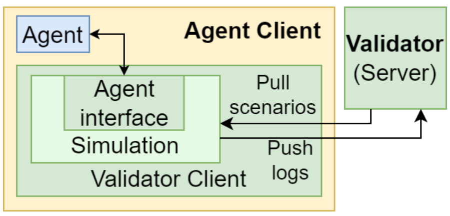


Figure 3. Agent Client architecture. Components of the client that runs the simulation.

During simulation, the process advances in time steps. At each step, the simulation calls `getNavigationRecommendation` with the updated scenario state and applies the returned navigation recommendation for each agent vessel. The simulation continues until either the target position is reached or the simulation times out.

D. Architecture: Validator

Figure 4 illustrates the internal architecture of the Validator. The Validator is written in the programming language Go and designed as a generic platform that can be extended by adding plugins. Plugins are executables that communicate with the Validator via Go's `net/rpc` implementation.

Scenarios requested by the Agent Client originate from Scenario Plugins, not from the Validator itself. This allows agent developers to create custom scenarios or reuse scenarios provided by others by simply placing the corresponding plugins in the Validator's plugin folder. The Validator automatically detects and loads all available plugins at startup. The simulation module executes scenario simulations using agent-controlled vessels (ACVs) and generates corresponding logs. When the Agent Client submits these logs, the evaluator module executes the rule and metric plugins on the logs and records the results in the report.

Rule plugins define the logical conditions that determine if an ACV has violated a rule. If so, the scenario is flagged as a failure. The default rule plugins verify whether the target position was reached and whether collisions occurred. Metric plugins provide quantitative safety indicators.

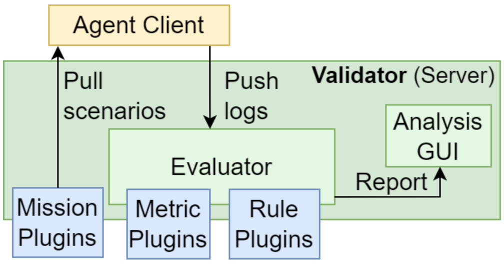


Figure 4. Components of the Validator that process simulation logs, check them against rules and metrics, and display the results in the GUI.

For example, they can offer a collision risk index that identifies potential hazards when vessels approach each other too closely or are on converging courses. When a metric reaches a critical value, the scenario is also flagged as a failure.

IV. PoC Plugin Implementations

For the proof of concept (PoC), we developed concrete implementations for each plugin interface, referred to as *PoC plugins*. These use simple illustrative methods rather than state-of-the-art approaches, as the goal is to demonstrate the plugin system. The focus lies on the Validator, which is designed to allow future integration of advanced plugins informed by current research.

The simulation model comprises an ACV, waypoint vessels following predefined routes without collision avoidance, and static obstacles such as circular or polygonal shapes.

Fifty scenarios were selected, ranging from simple navigation tasks to classic COLREG situations (head-on, crossing, overtaking) and more complex multi-vessel cases involving coastlines and obstacles.

A default rule plugin is included, which is applied universally to all scenarios. During simulation, it checks for collisions and verifies that each ACV reaches its target position within the scenario time limit. Any detected violation is recorded in the validation report.

Additionally, a PoC rule plugin implements a basic COLREG compliance checker, applicable to two-vessel scenarios without obstacles. It follows a two-step process: first identifying the COLREG situation based on relative headings and positions, then applying geometric compliance checks for the expected maneuvers. Noncompliance is flagged as a violation.

The metric plugin employs a simple formula based on the distance and time to closest point of approach (DCPA and TCPA). Exceeding defined thresholds, for example when ACVs approach too closely or are on a collision course, indicates a high risk of collision and is flagged as a violation.

V. Usage of the GUI

The Validator provides a GUI that allows agent developers to examine the results of validation runs on the analysis page (Figure 5). The analysis page functions like a video player and offers two perspectives. The map view (left) provides spatial and temporal visualization of vessel behavior. The data view (right) presents structured scenario metadata and plugin results, with violations marked in red.

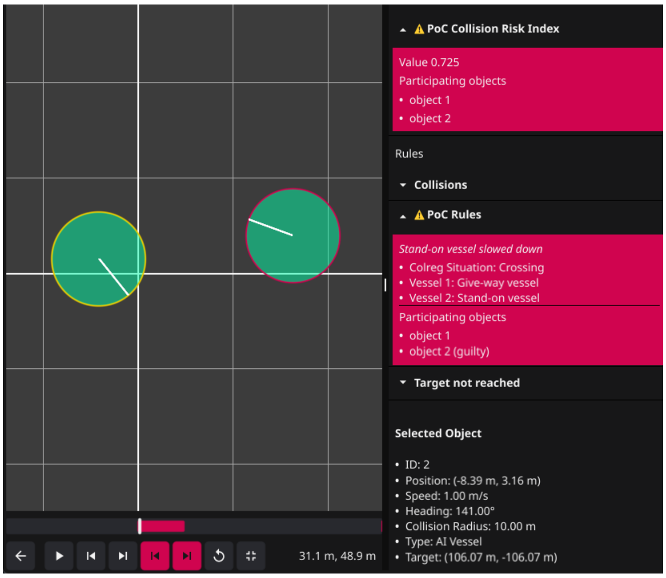


Figure 5. Screenshot of the Validator GUI showing two vessels with violations of a collision risk index metric and custom crossing-situation validation rules.

The map view includes playback controls that let users explore the scenario interactively by playing, pausing, or restarting via the control buttons, or by jumping to specific timestamps using the progress slider. Time steps with rule or metric violations are highlighted in red on the slider. Users can zoom into the map, inspect plugin outputs, and click on objects to display their physical parameters, such as position and speed.

Any plugin integrated with the Validator is automatically integrated into the GUI. As shown in Figure 5, the PoC rules plugin detected a crossing situation, flagged the incorrect behavior, and assigned responsibility to the vessel guilty of this violation. The PoC metrics plugin indicates excessive collision risk via the CRI in red.

VI. Discussion

In this section, we discuss the results of our approach and the Validator software.

A. Validation Approach

To evaluate the significance of our findings, we adopted two complementary methodologies: (i) virtual validation within the Agent Client simulation framework using a path-planning agent for autonomous maritime vehicles (Pathplanner), and (ii) live trials in which the same agent operated a unmanned underwater vehicle (UUV). Both the simulation environment and the UUV were supplied by TKMS ATLAS ELEKTRONIK GmbH.

The live trials did not include the Validator but were essential for verifying how closely the Pathplanner's behavior in the Agent Client virtual simulation corresponds to its performance in a physical setting.

To test the ability to integrate multiple agents, we implemented a dummy algorithm that navigates directly to the target position. This algorithm used the Validator client library and was run against the Validator. However, we have yet to test the system with fundamentally different agent implementations, such as neural network–based agents.

B. Pathplanner Algorithm

The Pathplanner uses an optimization-based obstacle-avoidance controller to identify situations governed by the COLREGs, as well as encounters with static obstacles. It continuously adjusts the UUV's heading and speed to maintain compliance with navigational rules while minimizing energy expenditure, incorporating detour penalties.

C. Live Simulation and Virtual Simulation

For the live simulation, the Pathplanner was installed on the UUV and connected to its onboard control system via the internal network. The UUV had a length of approximately 3 m. The live simulation was conducted in a lake in Bremen, Germany. The Pathplanner was responsible for navigating the UUV between georeferenced waypoints by issuing commands for target speed and heading. The UUV continuously transmitted state data, such as position, speed, and heading, back to the Pathplanner, enabling closed-loop control and real-time trajectory adjustment.

To simulate obstacles and other traffic, a pontoon operated according to scenario scripts and transmitted its position to the Pathplanner via Wi-Fi. Five scenarios were conducted, including avoiding a static obstacle, and basic COLREG situations such as crossing, and overtaking encounters. In all trials, the Pathplanner showed the same qualitative behavior as in our virtual simulation. This supports its use for behavior-level validation.

Quantitative accuracy is limited by the use of simplifying assumptions of the virtual simulation (see Section VI-F for details). Virtual and live runs lasted 80–219 s with travel distances of 75–214 m at a target velocity of 1.0 m/s. The average positional deviation between virtual and live runs was computed as the mean Euclidean distance per timestep. It was 1.5 m on average (minimum 0.6 m) under calm conditions, but increased to up to 14.6 m in the presence of wind and water currents.

D. Validator and Pathplanner Agent

The internals of the Pathplanner are treated as a black box, and the Validator only evaluates observable behavior from the logs. In the previous section, we established that the simulation is behaviorally faithful. Therefore, we used the Pathplanner as a case study in the simulation to examine how the Validator reveals agent deficiencies.

Overall, the Validator with the PoC plugins exposed flaws in the Pathplanner. The PoC scenario plugin provides instances of challenging geometry, such as narrow passages, where the planner would stall or orbit indefinitely. The Metrics plugin flagged overly aggressive or unsafe maneuvers, such as trajectories that were too close to other objects or that put the vessel on a collision course. The Rules plugin revealed situations in which the Pathplanner failed to reach its target position in time, collided with other objects, or violated the PoC rules in COLREG scenarios. Together, these plugins provide agent developers with quantitative and visual feedback, and offer clear diagnostic insight into where and when failures occurred.

We expect plugins developed by maritime navigation experts to improve detection accuracy and reveal additional shortcomings of validated agents.

E. Performance

TDD is a widely adopted practice in modern software engineering. Developers iteratively extend the agent's code while continuously validating it against a suite of predefined scenarios. After each modification, the scenarios are re-simulated. The simulation results are then validated to see if the changes improved behavior or introduced errors. This process is highly productive but depends on rapid feedback, ideally within seconds, to enable early detection and correction of issues.

To verify this performance requirement, measurements for the simulation and the Validator were conducted on a laptop with an Intel Core i5-1335U processor. A time resolution of one second was used, meaning that the simulation advances in one-second increments. The simulation achieves approximately 45 minutes of simulated scenario time for every real second. Each full validation run with our current setup takes under ten seconds to complete, which provided sufficient performance for efficient TDD cycles on mid-range hardware. We currently rely on the YAML encoder from the Go standard library. However, it is comparatively slow for our use case, which indicates significant potential for performance optimization.

F. Limitations

Our validation process consists of a separate client and server component, as well as plugins. This allows replacing plugins and agents but is more complex compared to a single monolithic component binding all test and production logic together. To achieve the best possible performance, the simulation engine and the agent communicate in-memory in the monolithic Agent Client. This makes the simulation engine dependent on the implementation language. Currently, only a Validator Client library for C++ is supported. This approach requires writing adapters around the simulation engine to work with agents implemented in other languages.

The virtual simulation currently uses a simplified environment model that does not account for wind, hydrodynamic drag, limited sight, drift, or communication between vessels.

This limits the variety of scenarios that can be created for it. Vessels are modeled as point-masses with circular collision shape that can be insufficient for more complex rules and metrics. This can also lead to problems in scenarios with near-miss situations which might be incorrectly detected as collisions.

VII. Conclusion and Future Work

This paper addresses fragmentation and lack of unified validation tools in the development of autonomous maritime navigation agents. We introduced the VaMAI Validator as a unified, agent-agnostic solution. Its architecture is designed for multi-agent integration. It provides standardized interfaces for scenarios, rules, and metrics, as well as GUI-based result analysis, automation, and plugin extensibility. The Validator reduces duplicated effort caused by agent-specific implementations by offering a platform with easy installation, streamlined automation, and uniform plugin interfaces, thereby enabling cross-agent benchmarking. A demonstration using a path-planning agent showed that the Validator can identify behavioral flaws in a virtual simulation environment. The architecture requires greater initial implementation effort and currently relies on relatively simple models in exchange for standardization. Nevertheless, it establishes a foundation for reproducible and extensible validation workflows in maritime agent development. Moreover, the architecture may yield similar benefits in other agent-validation domains beyond maritime navigation, potentially establishing a broader VaMAI methodology.

Several areas of future work could further enhance the Validator. Integrating additional maritime navigation agents would strengthen evidence of generality, with neural-network-based agents being of particular interest. In addition, the plugin ecosystem should be expanded to incorporate state-of-the-art models and recent scientific insights, increasing accuracy and strengthening the validity of agent performance evaluations. Finally, the system should be refined through deployment in real-world agent development workflows, and the physical models should be expanded to support different vessel types and the simulation of currents, wind, limited visibility, and inter-vessel communication.

Acknowledgment

The research project VaMAI leading to these results received funding from the German Federal Ministry for Economic Affairs and Energy (BMWE) with grant number 03SX587D.

References

[1] I. Porres et al., “On the Verification and Validation of AI Navigation Algorithms,” 2021, Accessed: Jan. 28, 2026. [Online]. Available: https://arxiv.org/abs/2101.06091

[2] “Open simulation platform,” Accessed: Jan. 28, 2026. [Online]. Available: https://opensimulationplatform.com/

[3] Q. A. Pham, C. B. Janaka, and K. Akshat, “ShipNaviSim: Data-driven simulation for real-world maritime navigation,” presented at the 24th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2025), May 19, 2025. Accessed: Jan. 28, 2026. [Online]. Available: https://ifaamas.csc.liv.ac.uk/Proceedings/aamas2025/pdfs/p1641.pdf

[4] C. Sauzé, “ASVTrafficSim: A simulator for autonomous surface vehicle and manned vessel collisions,” presented at the International Robotic Sailing Conference 2018, Aug. 31, 2018. Accessed: Jan. 28, 2026. [Online]. Available: https://www.researchgate.net/publication/331895545_ASVTrafficSim_A_simulator_for_Autonomous_Surface_Vehicle_and_Manned_Vessel_Collisions

[5] S. Mahmud, S. Saisubramanian, and S. Zilberstein, “Verification and Validation of AI Systems Using Explanations,” *Proceedings of the AAAI Symposium Series*, vol. 4, no. 1, pp. 76–80, Nov. 2024, ISSN: 2994-4317. DOI: 10.1609/aaaiss.v4i1.31774 Accessed: Jan. 28, 2026. [Online]. Available: https://ojs.aaai.org/index.php/AAAI-SS/article/view/31774

[6] H. Krasowski, S. Schärdinger, M. Arcak, and M. Althoff, *Intelligent Sailing Model for Open Sea Navigation*, 2025. DOI: 10.48550/ARXIV.2501.04988 Accessed: Jan. 28, 2026. [Online]. Available: https://arxiv.org/abs/2501.04988

[7] W. Xie, L. Gang, M. Zhang, T. Liu, and Z. Lan, “Optimizing Multi-Vessel Collision Avoidance Decision Making for Autonomous Surface Vessels: A COLREGs-Compliant Deep Reinforcement Learning Approach,” *Journal of Marine Science and Engineering*, vol. 12, no. 3, p. 372, Feb. 2024, ISSN: 2077-1312. DOI: 10.3390/jmse12030372 Accessed: Jan. 28, 2026. [Online]. Available: https://www.mdpi.com/2077-1312/12/3/372

[8] W. Wang, L. Huang, K. Liu, X. Wu, and J. Wang, “A COLREGs-Compliant Collision Avoidance Decision Approach Based on Deep Reinforcement Learning,” *Journal of Marine Science and Engineering*, vol. 10, no. 7, p. 944, Jul. 2022, ISSN: 2077-1312. DOI: 10.3390/jmse10070944 Accessed: Jan. 28, 2026. [Online]. Available: https://www.mdpi.com/2077-1312/10/7/944

[9] R. Sawada, K. Sato, and M. Minami, “Framework of safety evaluation and scenarios for automatic collision avoidance algorithm,” *Ocean Engineering*, vol. 300, p. 117 506, May 2024, ISSN: 00298018. DOI: 10.1016/j.oceaneng.2024.117506 Accessed: Jan. 28, 2026. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0029801824008436

[10] J. Gleeson, M. Dunbabin, and J. J. Ford, “COLREG Scenario classification and Compliance Evaluation with temporal and multi-vessel awareness for collision avoidance systems,” *Ocean Engineering*, vol. 313, p. 119 552, Dec. 2024, ISSN: 00298018. DOI: 10.1016/j.oceaneng.2024.119552 Accessed: Jan. 28, 2026. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0029801824028907

[11] P. Potočnik, “Model Predictive Control for Autonomous Ship Navigation with COLREG Compliance and Chart-Based Path Planning,” *Journal of Marine Science and Engineering*, vol. 13, no. 7, p. 1246, Jun. 2025, ISSN: 2077-1312. DOI: 10.3390/jmse13071246 Accessed: Jan. 28, 2026. [Online]. Available: https://www.mdpi.com/2077-1312/13/7/1246

[12] K. Wróbel, M. Gil, Y. Huang, and R. Wawruch, “The Vagueness of COLREG versus Collision Avoidance Tech-

niques—A Discussion on the Current State and Future Challenges Concerning the Operation of Autonomous Ships," *Sustainability*, vol. 14, no. 24, p. 16 516, Dec. 2022, ISSN: 2071-1050. DOI: 10.3390/su142416516 Accessed: Jan. 28, 2026. [Online]. Available: https://www.mdpi.com/2071-1050/14/24/16516

[13] Y. A. Prabowo, P. N. Hansen, D. Papageorgiou, and R. Galeazzi, "Good Seamanship Score Quantification in Complex and Congested Waterways," *IFAC-PapersOnLine*, vol. 58, no. 20, pp. 341–346, 2024, ISSN: 24058963. DOI: 10.1016/j.ifacol.2024.10.077 Accessed: Jan. 28, 2026. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S2405896324018329

[14] W. Li, L. Zhong, Y. Xu, and G. Shi, "Collision Risk Index Calculation Based on an Improved Ship Domain Model," *Journal of Marine Science and Engineering*, vol. 10, no. 12, p. 2016, Dec. 2022, ISSN: 2077-1312. DOI: 10.3390/jmse10122016 Accessed: Jan. 28, 2026. [Online]. Available: https://www.mdpi.com/2077-1312/10/12/2016

[15] Z. Liu, B. Zhang, M. Zhang, H. Wang, and X. Fu, "A quantitative method for the analysis of ship collision risk using AIS data," *Ocean Engineering*, vol. 272, p. 113 906, Mar. 2023, ISSN: 00298018. DOI: 10.1016/j.oceaneng.2023.113906 Accessed: Jan. 28, 2026. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0029801823002901

[16] X. Xin et al., "Multi-scale collision risk estimation for maritime traffic in complex port waters," *Reliability Engineering & System Safety*, vol. 240, p. 109 554, Dec. 2023, ISSN: 09518320. DOI: 10.1016/j.ress.2023.109554 Accessed: Jan. 28, 2026. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0951832023004684

[17] Y. Song and X. Cao, "Review of Intelligent Ship Path Planning Algorithms," *Frontiers in Management Science*, vol. 3, no. 1, pp. 90–101, Feb. 2024, ISSN: 27888592. DOI: 10.56397/FMS.2024.02.10 Accessed: Jan. 28, 2026. [Online]. Available: https://www.paradigmpress.org/fms/article/view/1023

The Energy and Autonomy Deficit: Barriers to Fielding Large Logistics UAS

Keirin Joyce
School of Engineering and Technology
University of New South Wales
Canberra, Australia
Email: k.joyce@unsw.edu.au

Keith Joiner
Capability Systems Centre
University of New South Wales
Canberra, Australia
Email: k.joiner@unsw.edu.au

Mark Hargreaves
School of Engineering and Technology
University of New South Wales
Canberra, Australia
Email: Mark.hargreaves@defence.gov.au

John Young
School of Engineering and Technology
University of New South Wales
Canberra, Australia
Email: John.young@unsw.edu.au

Vincent Daria
Air and Space Power Centre
Royal Australian Air Force
Canberra, Australia
Email: Vincent.daria@defence.gov.au

***Abstract*—Uncrewed Aerial Systems (UAS) have long been explored for logistics. While a number of systems offering small-payloads in commercial drone delivery and in military contexts have seen operational use, large-scale logistics UAS have yet to be adopted. A range of setbacks causes this, including technological and operational challenges that hinder the adoption of large logistics UAS. Here, we evaluate these challenges from a conceptual modelling perspective and forecast their applicability once these barriers are overcome. The study utilises technology trend modelling and activity mapping methodologies to predict the applicability of specific technologies that are currently identified to be operational challenges. Specifically, we model trends in technological improvements of the battery technology and aircraft control, and project its focus on landing zone autonomy and powertrain. The prediction illustration will focus on the current state of hybrid power and higher levels of automation as required for landing zone operations, and their development towards full autonomy. These models are validated through case studies of small commercial delivery drones and then applied to assess the feasibility and constraints of larger logistics UAS. Our analysis reveals that while small logistics UAS have been successfully integrated into operations, key technologies required for large-scale logistics UAS have yet to build-up a critical mass of research activity, particularly on landing zone autonomy and powertrain. Moreover, additional constraints beyond technological and operational challenges could include limitations in autonomy, certification hurdles, regulatory complexity, and the need for social and customer trust and acceptance.**

Keywords - Logistics UAS: autonomous drones: airspace regulation: drone certification.

I. INTRODUCTION

Logistics uncrewed aerial systems (UAS) have been in development for over six decades, at small and large scale [1]-[5]. Their business case is sound: reduced human intervention, timely delivery, efficiency, improved safety, automating supply chain, reducing cost, etc. And their impact will be profound once the technology scales.

At the small-scale end of uncrewed logistics aircraft, there are several historical examples of commercial success. In 2013, the multinational logistics company, DHL, delivered medicine using their Parcelcopter [6]. In short succession, a range of other examples followed, such as Google Wing, Amazon Prime Air and Zipline [7], with all three commencing initial operations in the period 2014 to 2016.

The success of fleets of these small parcel delivery was predicated on the maturing of several technology areas, including the development of more powerful and lighter lithium-ion batteries and solving the battery charging optimisation problem in 2018 [8][9]. Moreover, in 2017, small parcel delivery adopted automated flight controls, including improved avionics, fly-by-wire systems and optimised vehicle routing [6][10]-[13]. On mechanics, Distributed Electric Propulsion (DEP) made use of multiple electric motors and rotors, allowing for more stable and efficient vertical flight [14]-[18]. These drones were also made of lightweight materials such as carbon-fibre and other composites.

To be applicable for large logistics UAS, however, these drones need to be scaled up in size and capability and this is dependent on the maturity of key technologies [19]. Evolution needs to occur across five technology sub-sets: (1) autonomous flight operations integrated with traditional air traffic (including detect and avoid (DAA) and sense and avoid (SAA) technologies); (2) autonomous landing zone operations (including sensing technologies for that role); (3) powertrain developments commensurate to scaling to large logistics roles (including hybrid-electric powerplants); and (4) regulation and certification; and (5) and social license and acceptance. These last two include the regulation of future autonomous weapons systems [20][21][22], their ethical considerations [23]-[27] and social acceptance by law-abiding countries. Social acceptance in this regard has a distinct dual-use flavour, as large logistics UAS are expected to be used for commercial purposes, such as automated flying taxis and air ambulances, both of which require the public to be happy to be carried in them.

To forecast the applicability of large logistics UAS, it is important to review and evaluate key challenges. Here, we assess technological improvements of the battery technology and aircraft control, and project its focus on landing zone autonomy and powertrain. Specifically, the prediction illustration will focus on the current state of hybrid power and higher levels of automation as required for landing zone operations, and their development towards full autonomy. The revolution of small parcel delivery drones remains instructive. Identifying these technologies and forecasting their future applicability requires understanding innovation trajectories over time. We examine drone technologies through literature review and literature mapping, industry engagement, and activity analysis. Thence, we discuss these key technologies and analyse how each technology has evolved, ultimately aiming to qualitatively discuss and better forecast the applicability of large logistics drones.

II. METHODOLOGY

The technologies that led to the emergence of successful business models for small parcel delivery UAS are products of academic and community bodies of knowledge developed over several decades. These systems are based on small UAS used for photography in commercial roles and reconnaissance in military roles scaled in the early 2000's, that could carry a small amount of additional payload. Role scaling led to the mainstream adoption of multi-rotor UAS in the photography and real estate industries and to broad adoption in military operations and could be used to transport other equipment like envelopes, thumb drives, life vests, signal flares, etc. These aircraft were enabled by the maturity of miniaturised flight controllers, the reliability of batteries, and appropriate bandwidth data links.

To bring small parcel delivery drones to maturity, the primary problems that needed to be solved were the vehicle routing problem [6][10]-[13] and Battery-Charger Problem [8][9]. The commercial adoption of parcel delivery drones occurred once these problems were solved. Since then, the rates of industrial development and evolution in parcel delivery drones have significantly increased and outpaced academic technical publications. To forecast the technology's readiness, it is essential to track recent and contemporary developments in the industry. In Australia, where drone technology developments and innovation are mature, industry engagement is a valid method for exploring, scrutinising, and validating industry claims and publications. Notably, one of the most advanced small parcel delivery services, Google's Wing, was pioneered in Australia; another delivery service, Swoop Aero, was also based in Australia; and the most utilised flight controller for experimental UAS, ArduPilot, is also Australian. Hence, we've engaged with domestic industry to validate technological readiness of parcel delivery drones.

In general, we employed literature reviews, facilitated by bibliometric tool *Litmaps*, industry engagement, and activity analysis to examine logistics drone technologies and inform the assessment of future technology developments. Contemporary bibliometric analysis enabled by tools such as *Litmaps* is less structured than traditional approaches. The LitMaps tool proposes boolean queries/search strings and executes them for the researcher in the background. It also automatically builds links to identify key seed papers. We have chosen to display the *Litmaps* maps presented chronologically along the x-axis and citation count along the y-axis. To illustrate the relevance of citation count, the circles also increase in size in proportion to their citation counts. The plots are not linear: *Litmaps* uses a logarithmic scale that optimises the format for reader presentation. For this research activity, the data extraction occurred over March to November 2025 and the principle data source for citation count was *Google Scholar*. Modelling the operational reality is achieved when academic/industry breakthroughs correlate with commercial adoption. We can define the Time to Operational Reality (TOR) as:

$$\mathbf{TOR = T(breakthrough) + Lag}$$

Where T(breakthrough) is the year a technology subset reaches a critical mass of high-impact citations and Lag is a heuristic of historical 'breakthrough-to-use' observed in analogues. The methodology incorporates scoring rules and thresholds that are binary (pass/fail) based on a validation threshold (a technology is 'ready' when it shows a breakthrough pattern and commercial activity indicates scaling) and an 'invisible' penalty where, if regulatory maturity is low, a penalty factor is applied that can extend the forecast by decades.

Using predominantly small parcel delivery drone technologies we first confirm the methodology approach, which will then be expanded to include key areas of large-scale logistics UAS. For this paper, the model assumes that the successful trajectory of small supply drones (sub-25 kg) is a direct analogue for large-scale systems (payload of 500+ kg), that scarce academic research in high-stakes areas like landing autonomy is due to industrial secrecy rather than a total lack of technical progress and that primary uncertainty stems from strategic competition, where commercial or military competition can accelerates investment, shortening the Lag. Conversely, profound regulatory lag remains the largest source of unpredictability.

III. ANALYSING LOGISTICS DRONE TECHNOLOGY READINESS

It is worth noting that small logistics UAS gained their social license and acceptance, not due to a technology or commercial milestone, but due to the global pandemic of 2020, which forced much of the world into accepting the benefits of small parcel delivery [28]-[31]. Thus, although not without contention, small parcel delivery has become a ubiquitous element of modern economics. Additionally, lightweight structural material for aircraft was pioneered in the 1980s, firstly by the military, and then by commercial airline manufacturers [32]-[35]: it was already a very mature technology field by the time that small logistics drones needed composite airframes. This section explores key technological and operational readiness of large logistics UAS.

A. Landing zone autonomy

Small, commercial parcel delivery drones undertake landing operations with Global Positioning System (GPS) for location and simple and cheap altitude sensors, predominantly developed by the automotive industry. They can be simple because most operations are undertaken to and from well surveyed, urban areas, and the delivery landing zone is prescribed to be a cleared, flat area, usually a rooftop, driveway or clear backyard. However, robust landing conditions need to be considered for full applicability of parcel delivery drones and military logistics drones could provide some guidance. In highly contested areas, GPS cannot be assured, and the landing zones are not necessarily known in advance of a mission and cannot be assured to be clear from vegetation or other obstructions. Nor can perception systems from autonomous ground vehicles, which have benefited from development since the 1970's [36]-[40] and can be connected to the Internet, be transplanted onto drones as these aircraft are very weight sensitive and need sensors tuned to much longer ranges than those of cars. As such, military logistics drone landing zone operations need to be highly automated or semi-autonomous and augmented by multi-sensor terrain profiling. The field is not yet mature despite there being a range of academic sources over the past two decades [41]-[48], as seen in Figure 1.

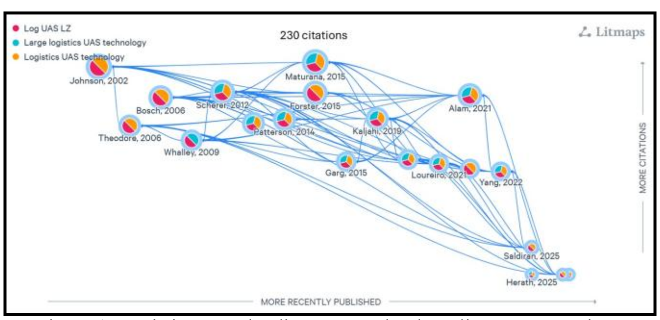


Figure 1. Logistics UAS landing zone technology literature mapping.

Our analysis reveals that publications are scarce and have a low impact, with the highest achieving only ~230 citations. None of the highest cited works show a citation pattern that indicates that they were 'breakthrough' research, as discussed by Faidi [49]. We surmise that there must be additional, unpublished research occurring within defence research institutions behind the curtains of security that cannot be accessed by open-source research, or as internal research and development within aerospace companies that is not being shared to retain ownership of the intellectual property of any breakthroughs. The motives for this are understood, but are noted as a likely significant cost factor compared to the more rapid development of this technology by a global research community.

Global, open-source research has been pointed out as a contributor to the successes of other autonomy technology developments over the past 20 years [16][50]-[53]. We should expect the development of landing zone autonomy for large logistics UAS to remain slow until the status quo changes. It is only with a significant, open, and prolonged investment in the development and integration of autonomy sensing systems that the challenges described above can be solved. Applying traditional, physical, and military acquisition approaches could mean it is not achieved until the 2030s at the earliest.

At a larger level, when considering whole of aircraft autonomy, it is noteworthy that the US Air Force, which is collaboratively prototyping with Joby Aviation a large, electric logistics and passenger aircraft, has not yet fully defined the Government Reference Architecture for Autonomy (A-GRA) [54]-[56] to which designers can design to. As of 2025, this is still in development. Thus, the journey towards fully autonomous large logistics aircraft still has a long way to go, as there remains a dearth of academic effort towards completing design reference architectures.

B. Powertrain

Batteries alone cannot power a large logistics drone across the distances that are needed to move large quantities of heavy parcels for commercial package delivery agencies, paying passengers for commercial taxi companies, patients for air ambulances, or combat supplies for the military. They just don't have the power density required [57]. For this reason a significant body of research has gone into hybrid-electric [58]-[60] and fuel cell [61]-[63] powerplants, with a considerable focus on hydrogen. Hydrogen provides distinct advantages of high power, long-range endurance, quieter operations and zero emissions, and enables the exploration of novel aircraft design concepts.

The focus on hydrogen is shared with and spun off from the automotive industry [64]-[66], and will need to consider the fundamental inputs to capability that will underpin that, such as generation, transportation, and storage infrastructure [67]. Field storage of hydrogen will be a particularly unique challenge for the military and may reduce the overall efficiency dividends presented by the military use of hydrogen as a fuel [68]-[72]. Our analysis shows that this is not a mature field, despite the range of academic sources over the past two decades, as seen in Figure 2.

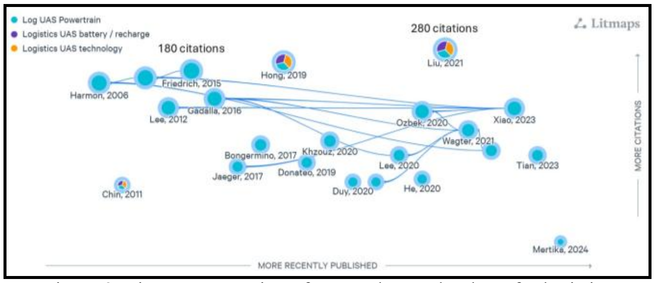


Figure 2. Literature mapping of powerplant technology for logistics uncrewed aerial systems.

Our analysis reveals that publications are scarce and have a low impact, with the highest achieving only ~280 citations on future battery chemistry to improve power density marginally (Liu, 2021), and the most impactful regarding hydrogen-powered aircraft, achieving only ~180 citations since 2015. None of the highest cited works show a citation pattern that indicates that they were ‘breakthrough’ research, as discussed by Faidi [49]. We surmise similar conclusions to our consideration of landing zone autonomy: that there must be additional, unpublished research within defence research institutions or internal research and development within aerospace companies that is not being shared to own the intellectual property of any breakthroughs. As such, we should expect hydrogen powerplant development for large logistics UAS to remain slow until the status quo changes. It is only with a significant, open and prolonged investment in the development and integration of these systems that the challenges described above can be solved. Applying traditional, physical and military acquisition approaches could see that not being achieved until the 2030s at the earliest, especially while significant technical challenges remain such as storage of liquid hydrogen, cryogenic management, lightening fuel cells, and thermal management of power electronics [73]-[77].

In the meantime, large logistics-like aircraft will exist, but they will be constrained to short distances on battery power only and will require the expense of a qualified pilot to be in the aircraft. If semi- or fully-autonomous large logistics drones are to succeed in a market breakthrough during a period of sustained strategic competition and conflict, significant profit could be generated, but may yet be years away, perhaps even decades. Poor awareness of technology maturity can lead to poor investment decisions and wasted money.

The case study of small parcel delivery drones in a commercial market, and on the recent application within the Ukrainian battlefield, illustrated the effectiveness of the methodology. The extension of that method to predict landing zone autonomy and powertrains capable of the ranges required for large logistics drones demonstrates the benefits.

IV. CONCLUSIONS

Our study has highlighted a distinct bifurcation in the developmental trajectory of logistics UAS. While small-scale logistics drones have achieved commercial viability and operational ubiquity, exemplified by the successes of Wing, Amazon Prime Air and Zipline, and their military counterparts, large-scale logistics UAS remain in a nascent, pre-commercial phase. By employing bibliometric analysis through Litmaps and validating trends against industry activities, this analysis package has demonstrated that the successful proliferation of small parcel delivery drones was predicated on the specific maturation of algorithmic solutions to the vehicle routing problem and the Battery-Charger Problem between 2014 and 2018. The correlation between high-impact academic literature and subsequent commercial adoption in the drone delivery sector serves as a validated heuristic for forecasting the readiness of larger systems.

Applying this methodology to large-scale logistic drones reveals a significant maturity gap. The analysis indicates that the critical technologies required to scale operations such as automated/autonomous landing zone sensing and high-endurance powertrains have not yet reached the ‘breakthrough’ levels of academic impact seen in earlier small-drone innovations. The scarcity of high-impact citations in these fields suggests that vital research is either stalling or, more likely, being sequestered within proprietary industrial silos or classified defence programs. The lack of open-source knowledge transfer acts as a brake on rapid innovation, preventing the wider industry from leveraging the collective problem-solving that propelled the small drone revolution. Consequently, the transition from retrofitted, expensive optionally piloted helicopters like the Unmanned K-MAX or U-Hawk to fully autonomous, purpose-built logistics platforms is unlikely to occur rapidly under the current development paradigm.

Ultimately, the future of large-scale logistics UAS will depend on a shift in acquisition and development strategies. It requires moving beyond the procurement of hardware and towards the co-development of certifiable autonomy architectures and the active sponsorship of regulatory frameworks. Until the ‘invisible’ barriers of regulation and the ‘visible’ barriers of power and sensing technologies are resolved in tandem, large-scale logistics drones will remain a niche capability rather than the revolution in military sustainment they promise to be. The timeline for widespread adoption is likely to stretch into the 2030s or beyond, requiring strategic patience and targeted investment in fundamental research rather than immediate procurement.

ACKNOWLEDGMENT

We acknowledge the Royal Australian Air Force, the Spitfire Association and the Sir Richard Williams Foundation for their support in completing this research program.

REFERENCES

[1] L. R. Newcome, *Unmanned aviation: a brief history of unmanned aerial vehicles*. AIAA, 2004.
[2] J. E. Peters, S. Seong, A. Bower, H. Dogo, A. L. Martin, and C. G. Pernin, *Unmanned Aircraft systems for logistics applications*. Rand Corporation, 2011.
[3] K. Joyce, "Logistics support for unmanned systems," in *Operations research for unmanned systems*: John Wiley and Sons, 2016.

[4] J. Choi, "#LOGBOTs – Making Army logistics “hard to find, hard to hit and hard to kill”," ed. Grounded Curiosity, 2018.
[5] Y. Li, M. Liu, and D. Jiang, "Application of unmanned aerial vehicles in logistics: a literature review," *Sustainability,* vol. 14, no. 21, p. 14473, 2022, doi: https://doi.org/10.3390/su142114473.
[6] A. Solomasov, "Analysis of supply chain operational performances using vehicle routing with UAV delivery in city logistics," Master en sciences de gestion, Université de Liège, Liège, Belgique, 2019.
[7] A. Jazairy, E. Persson, M. Brho, R. von Haartman, and P. Hilletofth, "Drones in last-mile delivery: A systematic literature review from a logistics management perspective," *The International Journal of Logistics Management,* vol. 36, no. 7, pp. 1-62, 2025.
[8] S. Park, L. Zhang, and S. Chakraborty, "Battery assignment and scheduling for drone delivery businesses," in *2017 IEEE/ACM International Symposium on Low Power Electronics and Design (ISLPED)*, 2017: IEEE, pp. 1-6.
[9] I. Hong, M. Kuby, and A. Murray, "A deviation flow refueling location model for continuous space: A commercial drone delivery system for urban areas," in *Advances in Geocomputation: Geocomputation 2015--The 13th International Conference*, 2017: Springer, pp. 125-132.
[10] C. C. Murray and A. G. Chu, "The flying sidekick traveling salesman problem: Optimization of drone-assisted parcel delivery," *Transportation Research Part C: Emerging Technologies,* vol. 54, pp. 86-109, 2015, doi: https://doi.org/10.1016/j.trc.2015.03.005.
[11] K. Dorling, J. Heinrichs, G. G. Messier, and S. Magierowski, "Vehicle routing problems for drone delivery," *IEEE Transactions on Systems, Man, and Cybernetics: Systems,* vol. 47, no. 1, pp. 70-85, 2016, doi: https://doi.org/10.1109/TSMC.2016.2582745.
[12] N. Agatz, P. Bouman, and M. Schmidt, "Optimization approaches for the traveling salesman problem with drone," *Transportation Science,* vol. 52, no. 4, pp. 965-981, 2018, doi: https://doi.org/10.1287/trsc.2017.0791.
[13] A. Otto, N. Agatz, J. Campbell, B. Golden, and E. Pesch, "Optimization approaches for civil applications of unmanned aerial vehicles (UAVs) or aerial drones: A survey," *Networks,* vol. 72, no. 4, pp. 411-458, 2018, doi: https://doi.org/10.1002/net.21818.
[14] D. Mellinger and V. R. Kumar, "Minimum snap trajectory generation and control for quadrotors," *IEEE International Conference on Robotics and Automation,* 2011, doi: 10.1109/icra.2011.5980409.
[15] R. Mahony, V. Kumar, and P. Corke, "Multirotor aerial vehicles: Modeling, estimation, and control of quadrotor," *IEEE robotics & automation magazine,* vol. 19, no. 3, pp. 20-32, 2012, doi: https://doi.org/10.1109/MRA.2012.2206474.
[16] ArduPilot. "ArduPilot: Versatile, trusted, open." https://ardupilot.org/ (accessed 08 Jul 25.
[17] R. Pol and M. V. N. Aher, "Enhanced Drone Control With F4v3s Controller: A Technical Analysis," *European Chemical Bulletin,* vol. 12, no. Special Issue 4, 2023.
[18] Betaflight. "Betaflight: Pushing the limits of UAV performance." https://betaflight.com/ (accessed 05 Sep 25.
[19] K. Stepanenko, "The battlefield AI revolution is not here yet: The status of current Russian and Ukrainian AI drone efforts," ed: Institute for the Study of War, 2025, pp. https://understandingwar.org/backgrounder/battlefield-ai-revolution-not-here-yet-status-current-russian-and-ukrainian-ai-drone.
[20] J. Williams, "Locating LAWS: Lethal autonomous weapons, epistemic space, and “meaningful human” control," *Journal of Global Security Studies,* vol. 6, no. 4, p. ogab015, 2021, doi: https://doi.org/10.1093/jogss/ogab015.
[21] M. Homayounnejad, "Lethal autonomous weapon systems under the law of armed conflict," King's College London, 2019.
[22] D. Copeland, R. Liivoja, and L. Sanders, "The utility of weapons reviews in addressing concerns raised by autonomous weapon systems," *Journal of Conflict and Security Law,* vol. 28, no. 2, pp. 285-316, 2023, doi: https://doi.org/10.1093/jcsl/krac035.
[23] C. Hoyos, "Development of autonomous UAVs raises ethical questions," *The Financial Times,* pp. 3-3, 2013.
[24] S. K. Devitt and D. Copeland, "Australia's approach to AI governance in security and defence," in *The AI wave in defence innovation*: Routledge, 2023, pp. 217-250.
[25] E. H. Christie, A. Ertan, L. Adomaitis, and M. Klaus, "Regulating lethal autonomous weapon systems: exploring the challenges of explainability and traceability," *AI and Ethics,* vol. 4, no. 2, pp. 229-245, 2024.
[26] T. Hellström, "On the moral responsibility of military robots," *Ethics and information technology,* vol. 15, no. 2, pp. 99-107, 2013, doi: https://doi.org/10.1007/s10676-012-9301-2.
[27] C. Enemark, "A Code of Ethics for Drone Users. ," in *Air/Space Blog*, ed: ASPC, 2023.
[28] S. Melo, F. Silva, M. Abbasi, P. Ahani, and J. Macedo, "Public acceptance of the use of drones in city logistics: A citizen-centric perspective," *Sustainability,* vol. 15, no. 3, p. 2621, 2023, doi: https://doi.org/10.3390/su15032621.
[29] Z. Zhang, C.-Y. Xiao, and Z.-G. Zhang, "Analysis and empirical study of factors influencing urban residents’ acceptance of routine drone deliveries," *Sustainability,* vol. 15, no. 18, p. 13335, 2023, doi: https://doi.org/10.3390/su151813335.
[30] D. Zhang, P. P.-J. Yang, and J.-Y. Tsou, "Advancing Social Equity in Urban UAV Logistics: Insights from the Academic Literature and Social Media," *Drones,* vol. 8, no. 11, p. 688, 2024, doi: https://doi.org/10.3390/drones8110688.
[31] S. Schmidt and A. Saraceni, "Consumer acceptance of drone-based technology for last mile delivery," *Research in Transportation Economics,* vol. 103, p. 101404, 2024, doi: https://doi.org/10.1016/j.retrec.2023.101404.
[32] A. P. Mouritz, *Introduction to aerospace materials*. Elsevier, 2012.
[33] S. Pantelakis and K. Tserpes, *Revolutionizing aircraft materials and processes*. Springer, 2020.
[34] S. Kumar and N. P. Padture, "Materials in the aircraft industry," in *Metallurgical design and industry: Prehistory to the space age*: Springer, 2018, pp. 271-346.
[35] S. Anand and A. K. Mishra, "High-performance materials used for UAV manufacturing: Classified review," *International Journal of All Research Education and Scientific Methods,* vol. 10, no. 7, pp. 2811-2819, 2022.
[36] S. P, R. Gudla, V. Telidevulapalli, J. Kota, and G. Mandha, "Review on self-driving cars using neural network architectures," *World Journal of Advanced Research and Reviews,* vol. 16, pp. 736-746, 11/30 2022, doi: 10.30574/wjarr.2022.16.2.1240.
[37] F. Rosique, P. J. Navarro, C. Fernández, and A. Padilla, "A systematic review of perception system and simulators for autonomous vehicles research," *Sensors,* vol. 19, no. 3, p. 648, 2019.
[38] A. Townsend, "The 100-Year History of Self-Driving Cars: What the long history of the autonomous vehicle reveals about its fast-approaching future," ed. Medium, 2020.
[39] J. Lindsey, "Self-driving cars vs the world," *Popular Mechanics,* pp. 44-58, 2024.
[40] L. Jones, "Driverless when and cars: where?[automotive autonomous vehicles]," *Engineering & Technology,* vol. 12, no. 2, pp. 36-40, 2017.
[41] S. Scherer, L. Chamberlain, and S. Singh, "Autonomous landing at unprepared sites by a full-scale helicopter," *Robotics and Autonomous Systems,* vol. 60, no. 12, pp. 1545-1562, 2012, doi: https://doi.org/10.1016/j.robot.2012.09.004.
[42] F. Amzajerdian, D. Pierrottet, L. B. Petway, G. D. Hines, V. E. Roback, and R. A. Reisse, "Lidar sensors for autonomous landing and hazard avoidance," in *AIAA SPACE 2013 conference and exposition*, 2013, p. 5312.
[43] T. Patterson, S. McClean, P. Morrow, G. Parr, and C. Luo, "Timely autonomous identification of UAV safe landing zones," *Image and Vision Computing,* vol. 32, no. 9, pp. 568-578, 2014.
[44] D. Maturana and S. Scherer, "3d convolutional neural networks for landing zone detection from lidar," in *2015 IEEE international conference on robotics and automation (ICRA)*, 2015: IEEE, pp. 3471-3478, doi: https://doi.org/10.1109/ICRA.2015.7139679.
[45] M. A. Kaljahi *et al.*, "An automatic zone detection system for safe landing of UAVs," *Expert systems with applications,* vol. 122, pp. 319-333, 2019.
[46] V. Turan, E. Avşar, D. Asadi, and E. A. Aydın, "Image processing based autonomous landing zone detection for a multi-rotor drone in

emergency situations," *Turkish Journal of Engineering,* vol. 5, no. 4, pp. 193-200, 2021.

[47] M. S. Alam and J. Oluoch, "A survey of safe landing zone detection techniques for autonomous unmanned aerial vehicles (UAVs)," *Expert Systems with Applications,* vol. 179, p. 115091, 2021, doi: https://doi.org/10.1016/j.eswa.2021.115091.

[48] T.-T. Nguyen, V. Van Rijswijck, G. De Cubber, B. Janssens, and H. Bruyninckx, "State-of-the-art autonomous landing solutions for UAVs on moving platforms," in *Autonomous Systems for Security and Defence II*, 2025, vol. 13680: SPIE, pp. 129-168.

[49] S. Faidi, *Assessing Bibliometrics for the Automation of Technology Readiness Level Assessments*. University of Toronto (Canada), 2021.

[50] S. Baldi, D. Sun, X. Xia, G. Zhou, and D. Liu, "ArduPilot-based adaptive autopilot: Architecture and software-in-the-loop experiments," *IEEE Transactions on Aerospace and Electronic Systems,* vol. 58, no. 5, pp. 4473-4485, 2022.

[51] FPV University, "How and why Betaflight was made? | From MultiWii to Betaflight and INAV," ed. YouTube, 2022.

[52] P. Burdziakowski, N. Razmjooy, V. Estrela, and J. Hemanath, "Open-source software (OSS) and hardware (OSH) in UAVs," 2020.

[53] D. Tezza and M. Andujar, "First-person view drones and the FPV pilot user experience," in *International Conference on Human-Computer Interaction*, 2022: Springer, pp. 404-417.

[54] J. Kauffman, "Reliable Robotics, U.S. Air Force collaborate on autonomy architecture," ed. Defense and Munitions, 2025.

[55] D. Taylor, "Collaborative Combat Aircraft autonomy architecture, air vehicle being developed separately, official says," ed. Military Embedded Systems, 2025.

[56] G. C. Allen and I. Goldston, "The Department of Defense's Collaborative Combat Aircraft Program: Good News, Bad News, and Unanswered Questions," ed. Center for Strategic and International Studies (CSIS), 2025.

[57] A. Townsend, I. N. Jiya, C. Martinson, D. Bessarabov, and R. Gouws, "A comprehensive review of energy sources for unmanned aerial vehicles, their shortfalls and opportunities for improvements," *Heliyon,* vol. 6, no. 11, 2020.

[58] M. Soleymani, V. Mostafavi, M. Hebert, S. Kelouwani, and L. Boulon, "Hydrogen propulsion systems for aircraft, a review on recent advances and ongoing challenges," *International Journal of Hydrogen Energy,* vol. 91, pp. 137-171, 2024.

[59] M. Dudek, P. Tomczyk, P. Wygonik, M. Korkosz, P. Bogusz, and B. Lis, "Hybrid fuel cell–battery system as a main power unit for small unmanned aerial vehicles (UAV)," *International journal of electrochemical science,* vol. 8, no. 6, pp. 8442-8463, 2013.

[60] C. Friedrich and P. A. Robertson, "Hybrid-electric propulsion for aircraft," *Journal of Aircraft,* vol. 52, no. 1, pp. 176-189, 2015.

[61] Ó. González-Espasandín, T. J. Leo, and E. Navarro-Arévalo, "Fuel cells: A real option for unmanned aerial vehicles propulsion," *The Scientific World Journal,* vol. 2014, no. 1, p. 497642, 2014.

[62] Z. Pan, L. An, and C. Wen, "Recent advances in fuel cells based propulsion systems for unmanned aerial vehicles," *Applied Energy,* vol. 240, pp. 473-485, 2019.

[63] Y. Pan, Q. Chen, N. Zhang, Z. Li, T. Zhu, and Q. Han, "Extending delivery range and decelerating battery aging of logistics UAVs using public buses," *IEEE Transactions on Mobile Computing,* vol. 22, no. 9, pp. 5280-5295, 2022.

[64] A. Albatayneh, A. Juaidi, M. Jaradat, and F. Manzano-Agugliaro, "Future of electric and hydrogen cars and trucks: an overview," *Energies,* vol. 16, no. 7, p. 3230, 2023.

[65] R. Shinnar, "The hydrogen economy, fuel cells, and electric cars," *Technology in society,* vol. 25, no. 4, pp. 455-476, 2003.

[66] D. W. Keith and A. E. Farrell, "Rethinking hydrogen cars," vol. 301, ed: American Association for the Advancement of Science, 2003, pp. 315-316.

[67] D. Ross, "Hydrogen storage: the major technological barrier to the development of hydrogen fuel cell cars," *Vacuum,* vol. 80, no. 10, pp. 1084-1089, 2006, doi: https://doi.org/10.1016/j.vacuum.2006.03.030.

[68] S. M. Katalenich and M. Z. Jacobson, "Toward battery electric and hydrogen fuel cell military vehicles for land, air, and sea," *Energy,* vol. 254, p. 124355, 2022, doi: https://doi.org/10.1016/j.energy.2022.124355.

[69] W. T. Mikolowsky and L. W. Noggle, "The potential of liquid hydrogen as a military aircraft fuel," *International Journal of Hydrogen Energy,* vol. 3, no. 4, pp. 449-460, 1978, doi: https://doi.org/10.1016/0360-3199(78)90005-8.

[70] K. D. Pointon and B. Lakeman, "Prospects for hydrogen as a military fuel," in *Assessment of Hydrogen Energy for Sustainable Development*: Springer, 2007, pp. 97-106.

[71] N. Sifer and K. Gardner, "An analysis of hydrogen production from ammonia hydride hydrogen generators for use in military fuel cell environments," *Journal of power sources,* vol. 132, no. 1-2, pp. 135-138, 2004, doi: https://doi.org/10.1016/j.jpowsour.2003.09.076.

[72] A. Soboń, D. Słyś, M. Ruszel, and A. Wiącek, "Prospects for the use of hydrogen in the armed forces," *Energies,* vol. 14, no. 21, p. 7089, 2021, doi: https://doi.org/10.3390/en14217089.

[73] M. T. Ahad, M. M. H. Bhuiyan, A. N. Sakib, A. Becerril Corral, and Z. Siddique, "An overview of challenges for the future of hydrogen," *Materials,* vol. 16, no. 20, p. 6680, 2023, doi: https://doi.org/10.3390/ma16206680.

[74] S. E. Hosseini and B. Butler, "An overview of development and challenges in hydrogen powered vehicles," *International Journal of Green Energy,* vol. 17, no. 1, pp. 13-37, 2020, doi: https://doi.org/10.1080/15435075.2019.1685999.

[75] M. Prewitz, A. Bardenhagen, and R. Beck, "Hydrogen as the fuel of the future in aircrafts–Challenges and opportunities," *International Journal of Hydrogen Energy,* vol. 45, no. 46, pp. 25378-25385, 2020, doi: https://doi.org/10.1016/j.ijhydene.2020.06.238.

[76] B. Sarlioglu and C. T. Morris, "More electric aircraft: Review, challenges, and opportunities for commercial transport aircraft," *IEEE transactions on Transportation Electrification,* vol. 1, no. 1, pp. 54-64, 2015, doi: https://doi.org/10.1109/TTE.2015.2426499.

[77] Y. Gu, M. Wiedemann, T. Ryley, M. E. Johnson, and M. J. Evans, "Hydrogen-powered aircraft at airports: a review of the infrastructure requirements and planning challenges," *Sustainability,* vol. 15, no. 21, p. 15539, 2023, doi: https://doi.org/10.3390/su152115539.