Survey of Open-Source Robotics Frameworks

Dawson Burgess
Computer Science Department
University of Idaho, Moscow, ID
Moscow, ID, United States
burg1648@vandals.uidaho.edu

Abstract—This survey examines the current state and evolution of open-source robotics frameworks, focusing on four critical themes: complexity, standardization, security, and sustainability. By reviewing references from Robot Operating System to emerging tools and simulators, this paper reveals that complexity remains a core challenge with integration difficulties persisting despite incremental improvements. Attempts at standardization have made limited progress due to heterogeneous hardware platforms and varying development goals. Recent years have seen a sharp increase in attention to security concerns. These concerns are driven by the recognition of code vulnerabilities and the growing importance of threat mitigation strategies. In contrast to security, sustainability—the long-term health of these open ecosystems—remains under-explored. This lack of attention raises concerns about the maintenance of active communities, funding, and relevance. By identifying these trends and gaps, this survey aims to provide insights into where future research and collaborative efforts can advance the reliability, usability, and impact of open-source robotics frameworks.

***Keywords*—Open-Source Robotics, Robotics Frameworks, Complexity Management, Standardization, Security in Robotics, Sustainability, Model-Driven Engineering, Interoperability, Community Engagement, Autonomous Systems.**

I. Introduction

Robotics can be defined broadly as the design, construction, operation, and deployment of robots. Robotics is an interdisciplinary field involving advancements in mechanical engineering, electronics, computer science, and related disciplines [12]. Over the past two decades, the intersection of robotics with open-source software ecosystems has accelerated innovation and facilitated the emergence of globally shared platforms. These open-source robotics frameworks offer general-purpose solutions that abstract complex hardware interfaces, provide modular and reusable components, and encourage community-driven collaboration and improvement [1], [2], [7], [13], [14], [25], [26], [27], [28]. As robots find increasing application in domains such as industrial automation, healthcare, defense, and service industries, open-source frameworks have become a critical factor in shaping the rate and direction of innovation within the field.

The rise of open-source robotics frameworks can be traced back to early efforts such as the Robot Operating System (ROS) [1]. ROS established a standardized middleware layer enabling interoperability between sensors, actuators, and control algorithms. Further iterations of ROS came as a result of the increasing complexity and scalability needs of modern robotics applications [2]. Other frameworks such as Yet-Another-Robot-Platform (YARP) [13] and RoboComp [14] have focused on modularity, flexibility, and interoperability. Environments like PyRobot [25], [26] and the Carnegie Mellon Navigation (CARMEN) Toolkit [27] highlight the importance of use, reuse, and accessibility. Efforts like GenoM [28] and frameworks grounded in model-driven engineering principles [3] have emphasized systematic software design to improve maintenance and long-term sustainability.

The evolution of open-source robotics frameworks has encountered challenges. One widely acknowledged issue is the increasing complexity of software architectures as robots grow more sophisticated and are deployed in dynamic environments [1], [2], [3]. Researchers have responded by proposing domain-specific modeling languages, model-driven engineering, and component-based designs [3], [9]. These approaches often struggle to gain widespread adoption due to high initial learning curves and the heterogeneity of robotic platforms. Standardization is another challenge. Despite the existence of numerous frameworks—Player/Stage [21], URBI, OROCOS [17], and various simulation tools such as Gazebo [22], MORSE [18], Webots [20], and others [19], [23], [24]—there remains no universal consensus on communication protocols, component interfaces, or performance metrics. Such fragmentation can slow progress because developers spend valuable resources integrating disparate tools and adapting to incompatible design philosophies [3], [8], [9].

Security has also emerged as a critical point of concern in open-source robotics. Public code availability is important to transparency and collaboration, but it can inadvertently expose vulnerabilities that malicious actors may exploit [4], [11]. Some studies have shown that frameworks like ROS, as well as robotics systems more broadly, often lack robust security architectures [4], [15], [16]. Mobile robotic assistant systems raise further alarms due to flaws in their security mechanisms that can threaten not only data integrity and system reliability, but also user safety [5], [6]. Securing robotic software—from low-level communications to AI-driven decision-making—remains a top priority in domains where robots operate in close proximity to humans or handle sensitive tasks.

Beyond technical sophistication and security, the sustainability of open-source robotics frameworks hinges on active communities and long-term stewardship [7], [10]. These ecosystems usually rely on volunteer developers, academic researchers, and industry partners who contribute code, documentation, bug reports, and tutorials. Over time, issues like contributor burnout, shifting commercial interests, and insufficient funding threaten the stability and growth of these platforms. As new frameworks and simulators emerge (e.g., Orca [17], Deepbots [20], and reconfigurable robot platforms [24]), existing projects must find new ways to remain relevant. Aligning open-source software with open hardware movements, improving documentation practices, and implementing community governance structures have been proposed as methods to enhance both short-term engagement and long-term viability [7], [10].

In addition to these established challenges, open-source robotics frameworks are increasingly integrated with advanced simulation and benchmarking tools to streamline development and testing. Simulators like CoppeliaSim (compared in [19]) and Gazebo [22] combined with testing frameworks [23] offer developers the ability to prototype and validate robot behaviors without risking expensive hardware or human safety. This trend complements the open-source ethos, allowing researchers and developers to share experimental results, replicate experiments, and refine robotic control strategies [22], [19], [20]. As research and development in autonomous navigation, manipulation, and multi-robot coordination intensify, standard simulation environments act as an impartial ground for benchmarking performance and comparing algorithms across different platforms [21], [24], [25].

The community-driven nature of open-source robotics has also catalyzed interdisciplinary collaboration. Initiatives bridging mechanical design, control theory, artificial intelligence, and human-robot interaction have emerged from open-source frameworks that simplify the integration of diverse components [1], [2], [13], [14]. As service robots, collaborative robots (cobots), and swarm robotics systems rise in use, open-source platforms play an increasingly key role in supporting complex behaviors and advanced autonomy features [2], [9], [27]. This cross-pollination is enabled by openly available tools and documentation. It encourages the formation of best practices and shared standards even as the field struggles with fragmentation and uneven adoption of proposed guidelines.

In light of these factors—growing complexity, lack of comprehensive standardization, persistent security risks, and concerns about sustainability—this paper aims to explore the current state of open-source robotics frameworks and identify emerging trends that may shape their future. Specifically, the following research questions will be explored:

1) Complexity and Usability: How do open-source robotics frameworks address the escalating complexity of robotic systems? What tools, methodologies, or design patterns are emerging to ensure ease of usability?

2) Standardization and Interoperability: To what extent have standardized models, languages, and protocols [3], [9] influenced the landscape of open-source robotics frameworks? What barriers remain to widespread consensus and practical adoption?

3) Security and Reliability: In what ways are open-source robotics frameworks incorporating security mechanisms to protect against vulnerabilities and attacks [4], [5], [6], [11], [15], [16]? How can these efforts be strengthened to ensure reliable and safe robotic operations?

4) Sustainability and Community Health: How are open-source robotics communities managing contributor engagement, funding, and resource allocation [7], [10]? What best practices are emerging to encourage long-term development, maintain relevance, and foster resilient ecosystems?

By systematically examining these questions, this survey seeks to provide an understanding of the current landscape of open-source robotics frameworks. The following sections present a literature review, discuss methodologies used in analyzing selected frameworks, and synthesize findings from various sources. This work aims to motivate further improvements, encourage collaboration, and guide both newcomers and established practitioners in navigating the complexities of the open-source robotics domain.

II. Literature Review

A. *Foundations of Open-Source Robotics Frameworks*

The literature on open-source robotics frameworks traces its foundations to efforts such as the ROS. Quigley et al. [1] introduced ROS as a middleware solution enabling developers to decouple hardware from software and to reuse modules across a range of robotic platforms. The architecture's publish-subscribe communication model simplified integration of diverse sensors, actuators, and control strategies, effectively lowering entry barriers for newcomers and facilitating large-scale code sharing. Macenski et al. [2] extended these principles with ROS 2, highlighting improved real-time capabilities, enhanced security, and support for distributed multi-robot systems. ROS and its successor accelerated the standardization of communication interfaces and device abstractions. Both frameworks inherited complexity and steep learning curves. Although widely adopted in research and industry, ROS-based systems often demand significant expertise.

Subsequent works by Brugali [3] and Nordmann et al. [9] introduced model-driven engineering and domain-specific modeling languages to mitigate complexity and improve maintainability. These approaches formalize software architectures and behaviors, reducing ad hoc development and enhancing reproducibility. Real-world uptake has remained modest due to concerns about initial overhead, tooling complexity, and the diverse hardware profiles in robotics that resist one-size-fits-all modeling solutions. Frameworks like YARP [13] and RoboComp [14] have emerged to promote modularity. YARP enables flexible component interconnection and RoboComp emphasizes tools-based approaches. These systems contribute to a richer ecosystem of reusable code but they also highlight how differences in design philosophies can hinder cross-framework compatibility, reinforcing the fragmentation that pervades the field.

B. *Complexity and Usability Considerations*

A key challenge repeatedly identified in the literature is how to manage complexity as robotic applications become more elaborate. Iñigo-Blasco et al. [7] explored how robotics software frameworks can cater to multi-agent systems, and argued for reusable building blocks that simplify adaptation to new tasks. PyRobot [25], [26] provides accessible Python-based interfaces and pre-packaged functionalities. This pre-packaging helps developers prototype and test robotic behaviors quickly. The Carnegie Mellon Navigation (CARMEN) Toolkit [27] also offers a unified approach to address navigation-specific challenges. Despite these attempts, user feedback often points to difficulties in mastering these systems. This difficulty stems from the multitude of available libraries, inconsistent documentation quality, and the lack of unified best practices.

Efforts to systematically evaluate and reduce complexity are hindered by a lack of established benchmarks. While model-driven approaches promise standardized architectures [3], [9], many frameworks still rely on community-driven conventions that vary widely. The result is an environment where some frameworks excel in certain niches yet fail to serve a broad audience without steep learning curves like ROS in manipulation and navigation tasks. Consequently, developers may repeatedly re-implement similar functionalities, thus reducing the overall efficiency that open-source ecosystems are meant to promote.

C. *Security and Reliability*

The literature also emphasizes growing security concerns in open-source robotics. Yaacoub et al. [4] identified vulnerabilities in ROS that can expose robots to denial-of-service (DoS) attacks and unauthorized control. DeMarinis et al. [11] went further by demonstrating the ease with which internet-wide scans can locate and access unprotected ROS systems. This vulnerability affirms the urgent need for built-in authentication and encryption protocols. Dinh et al. [5] and Neupane et al. [6] emphasized that these issues are not isolated to stationary platforms. Mobile robotic assistant systems, which are increasingly integrated into healthcare and public environments, face elevated risk as security breaches may compromise not only data integrity but also human safety.

While some literature points toward emerging solutions these efforts remain patchy. Macenski et al. [2] highlight incremental security enhancements in ROS 2, and Dieber et al. [16] propose strategies for securing robotics software stacks. DiLuoffo et al. [15] argues for an approach that involves threat modeling, secured communications, and continuous code auditing. The literature makes it clear that a dedicated, field-wide push towards cybersecurity best practices is still necessary. Without more comprehensive guidelines and widely adopted standards, the open-source community risks building critical infrastructure on unstable security foundations.

D. *Standardization Efforts and Interoperability*

The fragmentation of open-source robotics frameworks is a recurring theme in the subject literature. Multiple studies highlight how the absence of universal standards leads to needless duplication of effort. While ROS ushered in a de facto standard for middleware, no single solution dominates the entire development stack—from low-level drivers to high-level reasoning. Player/Stage [21], OROCOS [17], URBI, and GenoM [28] each adopt different approaches to modularity, task description, and component integration. Simulation environments like Gazebo [22], MORSE [18], Webots [20], and others [19], [23], [24] further complicate matters by adopting unique interfaces and capabilities, which limits straightforward interoperability.

Nordmann et al. [9] reviewed domain-specific modeling and languages as a potential path toward greater uniformity, while Castro et al. [8] focused on performance metrics in human-robot collaboration, highlighting the need for standardized evaluation criteria. Despite these contributions, no consensus on standardization has emerged. The diversity of application domains, hardware platforms, and research objectives means that what works as a standard in one corner of the community may be considered overly restrictive or irrelevant in another. As a result, attempts at standardization tend to meet adoption hurdles, leaving the field in a state of "cooperative fragmentation," where frameworks evolve in parallel rather than converging on a single universal standard.

E. *Sustainability and Long-Term Viability*

The open-source model thrives on sustained community involvement, yet the literature is replete with warnings about its cessation. Iñigo-Blasco et al. [7] have noted that without continuous maintenance, software frameworks quickly become obsolete and lose compatibility with evolving operating systems, libraries, and hardware drivers. Patel et al. [10] connected sustainability to the broader open hardware movement, arguing that successful long-term viability necessitates an ecosystem where both hardware and software iterate together.

This challenge is compounded as contributors come and go and leave gaps in documentation, testing, and security patching. The literature suggests potential remedies: better funding models, active community governance, incentives for maintaining legacy code, and investments in developer onboarding. Practical examples of these remedies are scarce, and limited guidance exists for communities seeking to improve their sustainability practices. The emergence of new platforms like Orca [17], Deepbots [20], and reconfigurable robot platforms [24] underscores the dynamic nature of the field but also magnifies the need for a stable, long-lived foundation to ensure that new initiatives can build on, rather than reinvent, existing technology.

F. *Identified Gaps and Future Directions*

Reviewing these contributions and critiques reveals a series of persisting gaps in the literature. While many works highlight the benefits of modularity, few provide rigorous, comparable benchmarks to objectively measure the complexity reductions that frameworks claim. The literature describes security enhancements, but no universal blueprint for secure robotics software has materialized. Domain-specific languages [3], [9] and standardized evaluation metrics [8] remain as niche interests rather than broadly implemented practices. The majority of solutions focus on isolated issues like interoperability or security rather than the holistic integration needed for a mature, reliable ecosystem of open-source robotics frameworks.

These gaps present opportunities for future research. As robotics becomes more integral to our way of life, the pressure to formalize best practices, adopt rigorous security frameworks, and ensure long-term sustainability will only intensify. The community's greatest challenge—and perhaps its greatest opportunity—lies in overcoming cultural, technical, and institutional barriers to systematic development. By addressing these challenges collectively, future work can guide the maturation of open-source robotics frameworks into robust, secure, user-friendly, and sustainable platforms that advance the field as a whole.

Existing literature details a vast array of open-source robotics frameworks, each offering valuable lessons in modularity, accessibility, security, and standardization. The scattered and incremental nature of these efforts underscores the need for comprehensive frameworks that integrate these dimensions into coherent, future-proof ecosystems.

III. Methodology

A. Methodologies and Approaches

To systematically evaluate the state of open-source robotics frameworks and address the research questions outlined in the introduction, this survey employed a multi-step methodology. This methodology encompassed data collection, selection criteria, comparative analyses, and quantitative visualization of thematic trends. The goal was to ensure rigor, reproducibility, and an objective lens through which to interpret the literature and identify key findings.

1) Data Collection: A comprehensive literature search targeted scholarly databases—IEEE Xplore, ACM Digital Library, and Google Scholar—focusing on works published between 2000 and 2024. Given the prominence of frameworks like ROS and ROS 2, initial keyword searches included: "open-source robotic frameworks," "Robot Operating System (ROS)," "robotic software standardization," "robotic simulation tools," "security in robotics," and "sustainability in open-source software." Reference lists of seminal works were also examined to identify additional influential texts [1], [2], [3]. A total of 28 references ([1]–[28]) were included, spanning key issues such as complexity management, standardization, security vulnerabilities, community-driven sustainability, and emerging simulation environments.

2) Framework and Theme Selection Criteria: The frameworks and studies analyzed were chosen based on three primary criteria:

a) Popularity and Usage: Frameworks widely adopted in both academic and industrial settings, e.g., ROS [1], ROS 2 [2].

b) Core Focus: Frameworks designed for modularity, reusability, and open-source collaboration—YARP [13], RoboComp [14], OROCOS [17], Player/Stage [21], URBI, GenoM [28].

c) Availability of Data: Frameworks discussed extensively in the literature, coupled with supporting documentation, user communities, and comparative analyses of tools such as Gazebo [22], MORSE [18], Webots [20], and others [19], [23], [24].

3) Content Analysis and Coding: Each of the 28 references was read and coded according to four thematic categories derived from the research questions: Complexity, Standardization, Security, and Sustainability. For example, sources discussing the integration difficulties and steep learning curves associated with frameworks were tagged as Complexity. Those emphasizing interoperability, domain-specific modeling, or the need for common benchmarks were tagged as Standardization. Papers highlighting vulnerabilities, threat models, or cybersecurity frameworks were coded as Security. Finally, studies exploring community health, funding models, and long-term viability were categorized under Sustainability.
If a reference addressed multiple themes, multiple tags were assigned. As proposed in foundational works [3], [7], [9], such tagging supports a structured assessment of the literature.

4) Quantitative Analysis, Visualization, and Interpretation of Graphs: After tagging, statistical analyses were performed to determine the relative frequency of each theme and to identify temporal and qualitative trends. Since the references did not provide explicit quantitative metrics for some dimensions (e.g., documentation quality), proxy values were inferred from the text. These approximations allow for meaningful comparisons and visual summaries, even if they do not represent absolute, objective measurements.

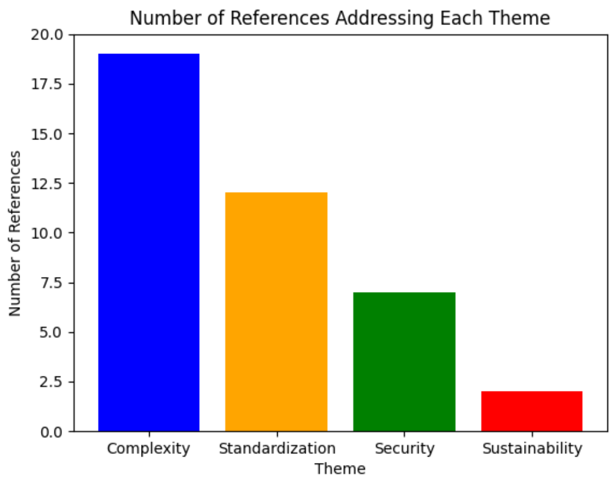

Figure 1: *References Addressing Each Theme*

Figure 1 displays the count of references addressing each theme (Complexity, Standardization, Security, and Sustainability). Bar charts are straightforward visual tools: each bar's height corresponds to how many references discuss that particular theme. For instance, Complexity had 19 references, indicating it is the most frequently discussed issue. Sustainability had 2 references, making it the least represented. The results of Figure 1 imply that while complexity remains a well-recognized topic, sustainability is comparatively underexplored, highlighting a potential gap in the literature.

Data for Figure 1:

- Complexity: 19 references
- Standardization: 12 references
- Security: 7 references
- Sustainability: 2 references

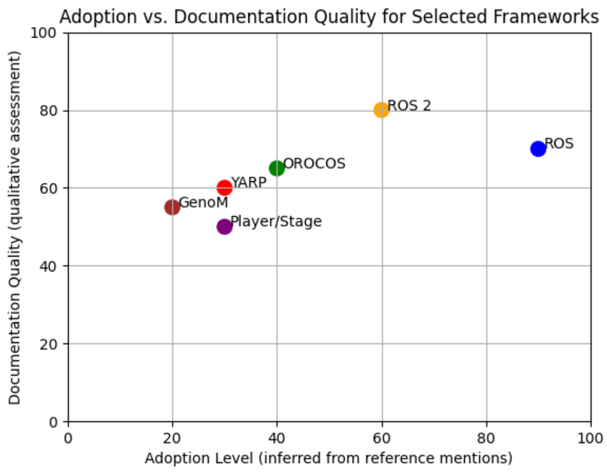

Figure 2: *Adoption vs. Documentation Quality for Selected*

Figure 2 is a scatter plot that compares selected frameworks—ROS, ROS 2, OROCOS, Player/Stage, YARP, and GenoM—based on inferred adoption levels and documentation quality. The x-axis represents adoption (approximated by how often frameworks are mentioned in the references), and the y-axis represents documentation quality (inferred from textual descriptions of community support, tutorials, and user-friendly features).

Interpreting this graph involves noting where each framework lies in relation to others. For example, ROS sits in a high-adoption, moderate-documentation zone, signifying widespread usage but room for improvement in user guidance. ROS 2, with slightly lower adoption but better documentation, suggests a framework that is growing in popularity and has made strides in accessibility. This reveals opportunities to invest in better documentation for figure widely-used frameworks or increase awareness of well-documented but under-adopted systems.

Data for Figure 2 (Adoption, Documentation):

- ROS: (90, 70)
- ROS 2: (60, 80)
- OROCOS: (40, 65)
- Player/Stage: (30, 50)
- YARP: (30, 60)
- GenoM: (20, 55)

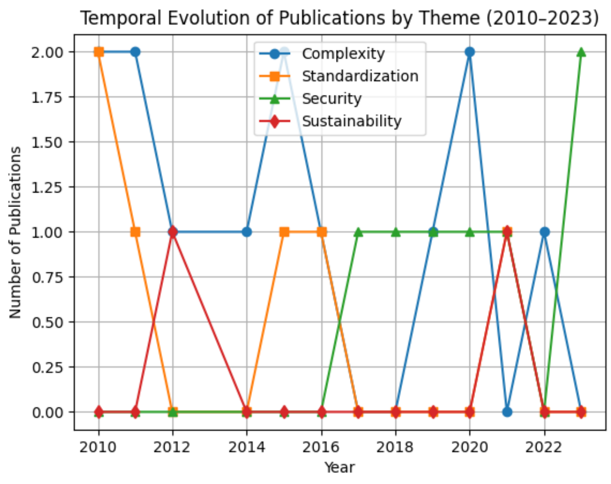

Figure 3: *Temporal Evolution of Publications by Theme*

Figure 3 shows how the focus on different themes has changed over time (2010–2023). Each line represents a theme, and each data point indicates how many references published in that year addressed that theme. The temporal evolution is critical for understanding trends: for example, security-related discussions were almost nonexistent before 2015 but saw a notable increase afterward. Complexity and standardization, by contrast, remained relatively steady concerns over the years, indicating ongoing, long-standing challenges. Sustainability appeared sporadically, pointing to either emerging interest or insufficient consistent focus over time.

By plotting these values annually, we observe shifts in priorities and areas of growing concern (like security) versus enduring issues (like complexity and standardization). This time-based perspective can guide future research by clarifying which topics are currently gaining momentum and which require renewed attention.

Data for Figure 3 (Number of Publications Per Year by Theme):

- Complexity: Present most years with counts ranging from 1–2 references.
- Standardization: Discussed intermittently (0–2 references/year).
- Security: Increases notably post-2015, reaching up to 2 references in 2023.
- Sustainability: Rarely addressed, with occasional mentions in a few years only.

5) Cross-Comparisons and Benchmarking: Patterns emerged while comparing frequency counts, tagging results, and temporal distributions. Complexity and standardization have been discussed extensively since early foundational works [1], [3], while security concerns rose significantly in recent years [2], [4], [15], [16]. Sustainability remains less frequently addressed, but a few references [7], [10] point to its growing importance.

These graphs work in concert to paint a multidimensional picture of the literature. Figure 1 clarifies how much attention each theme receives in aggregate. Figure 2 sheds light on how frameworks fare in terms of community adoption and helpfulness of documentation. Figure 3's temporal lens adds a historical perspective, showing that certain topics, like security, may surge in importance due to emerging threats or new industry standards.

B. *Findings and Trends*

The integrated coding and quantitative approaches revealed several key trends:

1) Complexity Dominance: Nearly two-thirds of the references addressed complexity, indicating that use, modulation, and management remain central challenges. Despite incremental improvements in user-friendly frameworks and tools, complexity persists as a core issue. Figure 1's complexity metric corroborates this point.

2) Partial But Persistent Standardization Efforts: About 12 references discussed standardization, suggesting ongoing but limited efforts. Attempts at domain-specific modeling [3], [9] have not yet resulted in widely adopted standards. Both Figures 1 and 3 confirm that while standardization is a topic of interest, it does not eclipse complexity or security in attention.

3) Rising Security Awareness: The temporal data from Figure 3 shows a marked increase in security-related discussions after 2015. Though fewer in number than complexity references, the growing body of work on vulnerabilities and mitigation strategies [4], [5], [6], [11], [15], [16] reflects an emerging priority for the community. Figure 1, though showing fewer total security references than complexity or standardization, must be interpreted in conjunction with Figure 3 to appreciate the recent surge in emphasis.

4) Limited Focus on Sustainability: Sustainability remained the least frequently discussed theme, with only 2 references [7], [10]. While Figure 1 clearly shows low overall frequency for sustainability, Figure 3's temporal perspective might encourage future researchers to focus more on building long-term, stable ecosystems as the field matures.

C. *Integration of Analyses*

By correlating thematic frequencies, temporal trends, and inferred metrics of adoption and documentation quality, the methodology provides a nuanced understanding of the open-source robotics landscape. The results indicate that the field deals with complexity and integration issues, gradually recognizing security concerns as integral to robotics development, and sporadically addressing sustainability and standardization.

Taken together, the figures and the underlying data guide us to several key conclusions. High complexity and moderate documentation quality point to a need for simplified integration processes, better tutorials, and more accessible tools. The historical spike in security-related publications suggests that frameworks and their communities are responding to newly recognized vulnerabilities and will likely remain a priority as robots enter more sensitive domains. Finally, the scarcity of sustainability-focused literature signals that greater effort may be required to ensure long-term framework viability, community health, and adaptability.

In summary, the methodology—augmented with carefully chosen visualization techniques—enables a structured, data-informed assessment of open-source robotics frameworks. Rather than relying solely on qualitative descriptions, these figures illustrate the field's priorities, identify underserved research areas, and help frame opportunities for future inquiry and improvement.

IV. Conclusion

The survey presented in this paper examined a rich body of literature on open-source robotics frameworks, synthesizing insights related to complexity, standardization, security, and sustainability. By applying a structured methodology—collecting data from a wide set of references [1]–[28], coding them into thematic categories, and visualizing trends—this work illuminated both the progress and persistent challenges in the field.

A. *Summary of Key Findings:*

The analysis revealed that complexity remains a central and enduring issue. Many studies [1], [2], [3], [7], [9], [12], [13], [14], [18]–[28] discuss difficulties associated with rapidly evolving robotics architectures, steep learning curves, and the integration of heterogeneous systems. Although frameworks like ROS and ROS 2 have eased some barriers [1], [2], and model-driven approaches [3], [9] aim to formalize development, complexity persists as a core challenge.

Standardization emerged as an important but elusive goal. While early and foundational works [1], [3], [9], [12], [13], [14] and attempts at domain-specific modeling provided guiding principles, no single standard has unified the field. Fragmentation among frameworks like Player/Stage [21], OROCOS [17], YARP [13], and others has led to increased effort in integration rather than convergence.

Previously overlooked security concerns are now receiving considerable attention [2], [4], [5], [6], [11], [15], [16]. The community is recognizing that open-source code transparency, while beneficial for collaboration, also exposes vulnerabilities. Efforts to incorporate more robust security architectures into platforms like ROS 2 and related frameworks are ongoing, yet a widely adopted security blueprint remains unrealized.

Sustainability has been discussed the least, with only a few references [7], [10]. Despite growing communities and volunteer-driven support, long-term viability risks—such as maintaining active contributor bases, preventing software obsolescence, and securing steady funding—are insufficiently addressed. The relative paucity of literature on sustainability underscores the need for more systematic approaches to community health and long-term planning.

B. *Advancing the Research Topic:*

By identifying these patterns, this survey contributes a consolidated view of the state of open-source robotics frameworks. It helps clarify which issues have seen substantial progress (improved documentation and interfaces for certain frameworks), which challenges are gaining momentum (security), and which remain underexplored (sustainability). The statistical data, charts, and thematic coding provide an empirical basis for understanding the field's current trajectory and highlight the interconnected nature of these challenges: addressing complexity often intersects with standardization, that security solutions must integrate seamlessly with existing software stacks, and that sustainability efforts can influence all aspects of community-driven frameworks.

C. Challenges, Gaps, and Limitations:

This survey's integrative approach reveals certain gaps. Chief among them is the scarcity of quantitative benchmarks and standardized metrics. For instance, while complexity is widely acknowledged, there is no universal way to measure improvements in usability or reductions in integration time. Similarly, security discussions remain disjointed, with no common framework for quantifying vulnerability impact or assessing the effectiveness of countermeasures. Standardization continues to be a goal rather than a reality, hampered by the sheer diversity of use cases and hardware configurations. Finally, sustainability remains peripheral in most research endeavors, pointing to a fundamental oversight that could threaten long-term innovation if left unaddressed.

Another limitation is that this survey relied on publicly available sources and textual descriptions and sometimes required qualitative judgments. The absence of uniform reporting standards across studies made it challenging to form universally comparable metrics. Additionally, the selected time frame may have excluded older pioneering works or very recent developments not yet documented.

V. Future Research

The findings suggest several avenues for future improvement:

1) Establishing Quantitative Benchmarks: Future work could focus on developing standard metrics and evaluation protocols for complexity and usability. For instance, controlled user studies, task completion times, or consistent code complexity measures would help researchers objectively assess whether proposed solutions reduce complexity [3], [9].

2) Toward Practical Standardization: Although domain-specific modeling and formal methodologies exist [3], [9], their adoption lags. Future research might explore lightweight standards, interfaces, and APIs that balance flexibility and interoperability. Incentivizing communities to adopt common formats, or working with industry consortia to define core standards could help frameworks coalesce around shared practices.

3) Comprehensive Security Frameworks: With vulnerabilities becoming more evident [4], [5], [6], [11], [15], [16], the field requires integrated security models that address everything from low-level communication protocols to AI-driven decision-making algorithms. Future research could aim to produce universally applicable guidelines, tools for automated vulnerability scanning, and security certification processes that encourage framework developers to adopt secure-by-design principles.

4) Long-Term Sustainability Models: As robotics frameworks proliferate, sustaining them over time will be vital. Future studies might analyze successful open-source projects in other domains, investigating funding models, contributor incentivization mechanisms, training programs, and robust documentation pipelines [7], [10]. Identifying what conditions enable a healthy ecosystem would inform best practices to help robotics frameworks thrive.

5) Aligning Software and Hardware Innovation: With robotics tightly coupled to evolving sensors, actuators, and computing hardware, future research could integrate open hardware movements more closely with open-source software efforts [10]. Designing frameworks that adapt seamlessly to new hardware platforms could reduce fragmentation and future-proof existing codebases.

In summary, addressing complexity, standardization, security, and sustainability together would foster a more cohesive, robust ecosystem for open-source robotics. As researchers, developers, and end-users collaborate across these domains, future studies can chart a course toward frameworks that accelerate innovation and broaden the impact of robotics worldwide.

References

[1] Quigley, M., et al. (2009). "ROS: an open-source Robot Operating System." *IEEE International Conference on Robotics and Automation (ICRA)*. Url: *https://api.semanticscholar.org/CorpusID:6324125*

[2] Macenski, S., et al. (2020). "Robot Operating System 2: Design, architecture, and uses in the wild." *Science Robotics*, 5(47), eabb4702.

[3] Brugali, D. (2015). "Model-driven software engineering in robotics: Models are designed to use, reuse, and maintain code." *IEEE Robotics & Automation Magazine*, 22(2), 155-166. doi: https://doi.org/10.48550/arXiv.2211.07752

[4] Yaacoub, J.-P. A., Noura, H. N., Salman, O., & Chehab, A. (2021). "Robotics cyber security: vulnerabilities, attacks, countermeasures, and recommendations." *International Journal of Information Security*, 21, 115–158. https://doi.org/10.1007/s10207-021-00545-8

[5] Dinh, L. Q., Nguyen, D. T., Vu, T. C., Nguyen, T. V., & Nguyen, M. T. (2023). "Comprehensive Review of Security Problems in Mobile Robotic Assistant Systems: Issues, Solutions, and Challenges." *Journal of Communications and Technology Advances*. https://doi.org/10.62411/jcta.11408

[6] Neupane, S., Mitra, S., Fernandez, I. A., Saha, S., Mittal, S., Chen, J., Pillai, N., & Rahimi, S. (2023). "Security Considerations in AI-Robotics: A Survey of Current Methods, Challenges, and Opportunities." *Mississippi State University*, Department of Computer Science & Engineering. Doi:10.1109/ACCESS.2024.3363657

[7] Iñigo-Blasco, P., Diaz-del-Rio, F., Romero-Ternero, M. C., Cagigas-Muñiz, D., & Vicente-Diaz, S. (2012). "Robotics Software Frameworks for Multi-Agent Robotic Systems Development." *University of Seville, Escuela Técnica Superior de Ingeniería Informática*. doi:10.1016/j.robot.2012.02.004

[8] Castro, A., Silva, F., & Santos, V. (2021). "Trends of Human-Robot Collaboration in Industry Contexts: Handover, Learning, and Metrics." *Sensors*, 21(12), 4113. https://doi.org/10.3390/s21124113

[9] Nordmann, A., Hochgeschwender, N., Wigand, D., & Wrede, S. (2016). "A Survey on Domain-Specific Modeling and Languages in Robotics." *Research Institute for Cognition and Robotics (CoR-Lab), Bielefeld University, Bonn-Rhein-Sieg University. Journal of Robotics and Automation*, 1, 75-99. url: https://pub.uni-bielefeld.de/record/2788380

[10] Patel, V. V., Liarokapis, M. V., & Dollar, A. M. (2021). "Open Robot Hardware: Progress, Benefits, Challenges, and Best Practices." *IEEE Robotics and Automation Magazine*, Student and Senior Member, IEEE.vol. 30, no. 3, pp. 123-148, Sept. 2023, doi: 10.1109/MRA.2022.3225725.

[11] N. DeMarinis, S. Tellex, V. P. Kemerlis, G. Konidaris and R. Fonseca, "Scanning the Internet for ROS: A View of Security in Robotics Research," *2019 International Conference on Robotics and Automation (ICRA)*, Montreal, QC, Canada, 2019, pp. 8514-8521, doi: 10.1109/ICRA.2019.8794451.

[12] A. Hentout, A. Maoudj and B. Bouzouia, "A survey of development frameworks for robotics," *2016 8th International Conference on Modelling, Identification and Control (ICMIC)*, Algiers, Algeria, 2016, pp. 67-72, doi: 10.1109/ICMIC.2016.7804217.

[13] G. Metta, P. Fitzpatrick, and L. Natale, "YARP: Yet Another Robot Platform," presented at the 1st IEEE-RAS International Conference on Humanoid Robots (Humanoids), 2001. doi: 10.5772/5761.

[14] Manso, L., Bachiller, P., Bustos, P., Núñez, P., Cintas, R., Calderita, L. (2010). RoboComp: A Tool-Based Robotics Framework. In: Ando, N., Balakirsky, S., Hemker, T., Reggiani, M., von Stryk, O. (eds) Simulation, Modeling, and Programming for Autonomous Robots. SIMPAR 2010. Lecture Notes in Computer Science(), vol 6472. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-17319-6_25

[15] DiLuoffo V, Michalson WR, Sunar B. Robot Operating System 2: The need for a holistic security approach to robotic architectures. International Journal of Advanced Robotic Systems. 2018;15(3). doi:10.1177/1729881418770011

[16] B. Dieber, B. Breiling, S. Taurer, S. Kacianka, S. Rass, and P. Schartner, "Security for the Robot Operating System," Robotics and Autonomous Systems, vol. 98, pp. 192–203, 2017. doi: 10.1016/j.robot.2017.09.017.

[17] Brooks, Alex & Kaupp, Tobias & Makarenko, Alexei & Williams, Stefan & Orebäck, Anders. (2007). Orca: A Component Model and Repository. 10.1007/978-3-540-68951-5_13.

[18] G. Echeverria, N. Lassabe, A. Degroote and S. Lemaignan, "Modular open robots simulation engine: MORSE," 2011 IEEE International Conference on Robotics and Automation, Shanghai, China, 2011, pp. 46-51, doi: 10.1109/ICRA.2011.5980252.

[19] A. Farley, J. Wang, and J. A. Marshall, "How to pick a mobile robot simulator: A quantitative comparison of CoppeliaSim, Gazebo, MORSE and Webots with a focus on accuracy of motion," Simulation Modelling Practice and Theory, vol. 120, p. 102629, 2022. doi: 10.1016/j.simpat.2022.102629.

[20] Kirtas, M., Tsampazis, K., Passalis, N., & Tefas, A. (2020). Deepbots: A Webots-Based Deep Reinforcement Learning Framework for Robotics. Artificial Intelligence Applications and Innovations, 584, 64 – 75. url: https://api.semanticscholar.org/CorpusID:218980681

[21] Michal, D. S., & Etzkorn, L. (2011). A comparison of Player/Stage/Gazebo and Microsoft Robotics Developer Studio. Proceedings of the 49th Annual Southeast Regional Conference on - ACM-SE '11. doi:10.1145/2016039.2016062

[22] N. Koenig and A. Howard, "Design and use paradigms for Gazebo, an open-source multi-robot simulator," 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE Cat. No.04CH37566), Sendai, Japan, 2004, pp. 2149-2154 vol.3, doi: 10.1109/IROS.2004.1389727.

[23] W. Qian et al., "Manipulation task simulation using ROS and Gazebo," 2014 IEEE International Conference on Robotics and Biomimetics (ROBIO 2014), Bali, Indonesia, 2014, pp. 2594-2598, doi: 10.1109/ROBIO.2014.7090732.

[24] M. Fujita, H. Kitano, and K. Kageyama, "A reconfigurable robot platform," Robotics and Autonomous Systems, vol. 29, no. 2–3, pp. 119–132, 1999. doi: 10.1016/S0921-8890(99)00047-0.

[25] Murali, A., Chen, T., Alwala, K.V., Gandhi, D., Pinto, L., Gupta, S., & Gupta, A.K. (2019). PyRobot: An Open-source Robotics Framework for Research and Benchmarking. ArXiv, abs/1906.08236. Doi: 10.48550/arXiv.1906.08236

[26] S. Lemaignan, A. Hosseini and P. Dillenbourg, "PYROBOTS, a toolset for robot executive control," 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Hamburg, Germany, 2015, pp. 2848-2853, doi: 10.1109/IROS.2015.7353769.

[27] M. Montemerlo, N. Roy and S. Thrun, "Perspectives on standardization in mobile robot programming: the Carnegie Mellon Navigation (CARMEN) Toolkit," Proceedings 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003) (Cat. No.03CH37453), Las Vegas, NV, USA, 2003, pp. 2436-2441 vol.3, doi: 10.1109/IROS.2003.1249235.

[28] A. Mallet, C. Pasteur, M. Herrb, S. Lemaignan and F. Ingrand, "GenoM3: Building middleware-independent robotic components," 2010 IEEE International Conference on Robotics and Automation, Anchorage, AK, USA, 2010, pp. 4627-4632, doi: 10.1109/ROBOT.2010.5509539.