Decentralized Mobile Robot Collaboration in Multi-Vendor Contexts: Service Robot Crowdsourcing via Distributed Ledger Technology

Yoshito Watanabe[1 2], Ryohei Banno[2], and Takashi Miki[1]

[1]*Solid Surface Inc., Tokyo, Japan*
{yoshito.watanabe, takashi.miki}@solidsurface.co.jp
[2]*Hitotsubashi University, Tokyo, Japan*
banno@computer.org

***Abstract*—The collaboration among service robots operated by multiple vendors is in demand, but it is difficult to achieve due to issues such as the lack of interoperability among vendor-specific operating systems. To tackle this problem, we propose a crowdsourcing platform leveraging distributed ledger technology (DLT), where users can request services by flexibly combining multiple tasks assigned to robots. We also present a solution to avoid collisions of mobile robots by employing a universal space identification technology, called *Spatial-ID*, and defining a *Space of Attention (SoA)* as a location that requires attention for navigating robots. Our approach does not require the sharing of navigation maps among vendors unlike conventional methods. As a case study, we compared the transaction approval latency of the Ethereum blockchain and IOTA Tangle using their test networks through computer simulations. We then evaluated the risk of transaction conflicts, concluding that IOTA Tangle is a promising technology for implementing the proposed platform.**

***Index Terms*—Service robot, distributed ledger technology, crowdsourcing, Spatial-ID, Space of Attention**

I. Introduction

Mobile service robots are becoming more popular, and their demand has increased even further with the COVID-19 pandemic worldwide [1]. The Digital Agency in Japan has recently published a report on the cooperative operation of multiple mobilities [2], which asserts that mobilities owned by multiple vendors should be operated collaboratively, not only within a building but also across various fields. This requires a framework for communication among vendors and information sharing on field navigation rules, as well as enabling users to request services using multi-vendor robots.

Robots are conventionally controlled by siloed operation systems specific to each vendor; they typically adopt vendor-specific application interfaces, communication protocols, data formats, and maps for navigating robots. Therefore, direct vendor-to-vendor robot coordination would require interface adapters or data conversion for each individual vendor. Besides, such straightforward inter-vendor collaboration would necessitate interpersonal negotiation among vendors. Consequently, this approach is time-consuming and labor-intensive, resulting in a lack of scalability in multi-vendor collaboration. Although the open robotics middleware framework (Open-RMF) [3] exists and could help improve interoperability and interconnectivity, it does not fundamentally solve the scalability problem.

One might consider a solution where a third-party organization intervenes between vendor operation systems as a broker in a centralized manner. However, such a solution could create a *single point of failure* in collaborating robots. Moreover, such an organization could change rules and policies without the consent of the robot vendors and might also censor data from robot vendors.

To address these issues, this paper proposes a framework for the coordination of service robots operated by multiple vendors using distributed ledger technology (DLT). Specifically, we present a crowdsourcing platform for service robots where users can request robots to perform their desired tasks. Furthermore, we propose a method to avoid congestion and collisions during multiple robot operations. We utilize a universal space identification and attribute-storing technology called *Spatial-ID* [4], and define a *Space of Attention (SoA)* as a location that requires attention for navigating robots, such as narrow corridors and corners. Our approach does not require the sharing of navigation maps among vendors, thus allowing for high scalability in operational areas and adaptability to dynamic environmental changes.

As a preliminary investigation, we discuss the advantages and disadvantages of both public and private distributed ledger technologies (DLTs) from business and technical perspectives. We also evaluate and compare the transaction processing speeds of distinct DLTs using their test networks, providing suggestions for selecting the most suitable technology to implement the proposed platform.

II. Related Work

The topic of mobile robot collaboration has been widely studied in the literature. Zhao, et al. [5] propose a surplus topology generation algorithm (STGA) to construct global topological maps, effectively addressing path conflicts and blockages in multi-robot systems through enriched topological

relationships and efficient coordination strategies. Due to its advantages in terms of decentralization and security, blockchain and DLTs are often integrated for robot collaboration. Mokhtar et al. [6] introduce a multi-robot path planning method based on a permissioned blockchain, where robots share a common workspace specification and static obstacles. Salimi et al. [7] introduce a secure framework for heterogeneous multi-robot collaboration and docking integrating blockchain identities and smart contracts for efficient coordination between ground and aerial robots in industrial applications. Li et al. [8] propose a blockchain-based collaborative edge knowledge inference (BCEI) framework for multi-robot systems, where robots collect and share knowledge through edge nodes using a permissioned blockchain. Alsamhi and Lee [9] propose a blockchain-empowered multi-robot collaboration framework to combat COVID-19, utilizing a blockchain network to enhance robot coordination for tasks such as disinfection, monitoring, and delivery. Ferrer et al. [10] present a blockchain-based Byzantine-resilient framework for multi-robot systems, utilizing a reliable blockchain to securely broadcast leader signals to followers.

Almost all of these studies consider robots as individual units and ensure decentralization through DLTs. In reality, however, robots are often centrally managed by a particular vendor. In the near future, it is anticipated that multiple vendors will operate robots in facilities and urban areas. Therefore, it is crucial to consider the context of a multi-vendor environment. To this end, we propose a crowdsourcing platform designed to allow vendor-specific proprietary systems to be connected to a network of DLT.

It is important to note that the idea of implementing crowdsourcing with DLT is not new, and its advantages, such as the ability to build an infrastructure that does not rely on centralized authority, have been well publicized in [11]–[13] on human crowdsourcing. A recent study in [14], which does not employ DLT though, proposes a task assignment method using a game theoretic approach for robotic crowdsourcing, but it still fails to consider scenarios where the robots are operated by multiple vendors. The main contribution of this paper is the design of a crowdsourcing platform where service robots are employed as workers, taking into account the context in which robots are managed by different vendors.

III. System Overview

This section provides an overview of the proposed system.

A. An Architecture of the Overall System

Figure 1 shows the overall architecture of the proposed system. We consider a business model where multiple robotic vendors participate in operating robots within a large commercial facility, although the model can also be applied to operations in wider-range public spaces. We define three layers in the overall architecture: the service configuration (SC), task coordination (TC), and task execution (TE) layers.

In the TE layer, there is a system to monitor and control the robots owned by each vendor. This vendor operation system is typically implemented in the cloud, but in some cases, a robot itself can perform the same function without using a dedicated system. In either case, there is no significant difference in the subsequent discussion.

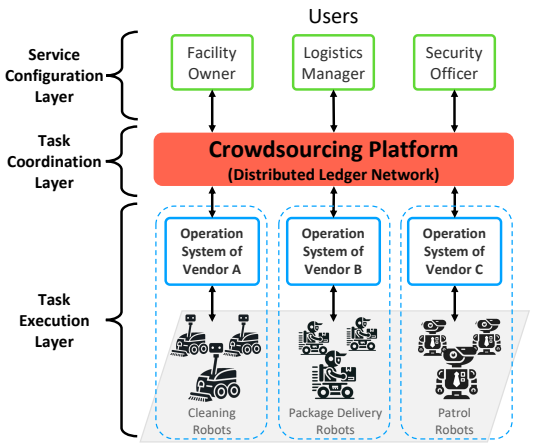

Fig. 1. An overall architecture of the proposed system.

In a conventional setup, the vendor operation systems are siloed, proprietary, and therefore incompatible with each other. This incompatibility makes it difficult for robots operated by different vendors to collaborate straightforwardly. The proposed system introduces a crowdsourcing platform for service robots to enable collaboration among different vendors in the TC layer. This layer serves as a hub for common interfaces and data formats for collaboration among multiple vendors and as a bridge between users and service robots.

The main users of the platform could be facility owners, logistics managers, and security officers in facilities, but the system should also be open to general consumers, such as tourists and visitors. The SC layer should provide the users with interfaces and applications that allow them to access the crowdsourcing platform so that they can request their demanded jobs from robots. The crowdsourcing platform is responsible for matching the supply and demand between robots and users. DLT is used to implement the proposed platform to operate it without any centralized authority.

The proposed architecture enables vendors to make full advantage of their already-implemented operation systems; thus, it is both natural and appropriate for the robots to connect to the distributed ledger through these systems. This approach allows the proposed system to provide interoperability among vendor operation systems.We utilize smart contracts to automate the process of robot collaboration, and all financial transactions in the system can be handled using cryptocurrency. An additional, yet significant, advantage of employing DLT is that since the data is recorded in a tamper-proof form, it can serve as credible evidence when tracking responsibility in the event of operational incidents.

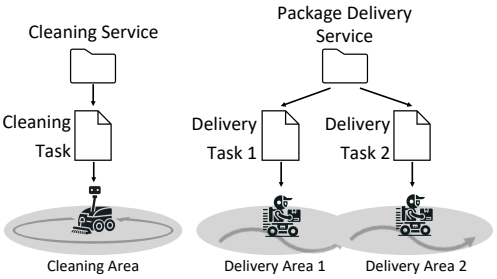

Fig. 2. The relationship between services and tasks.

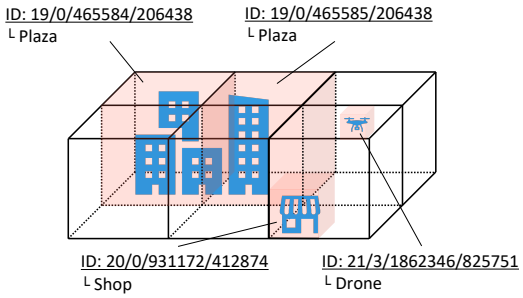

Fig. 3. An Example of Spatial-IDs.

B. Services and Tasks

Here we define the two terms, namely, *service* and *task*. A *service* is a single series of operations requested by a user and can consist of multiple *tasks*. A *task* is always assigned to a robot. Both services and tasks can have defined start and end times.

For example, a cleaning service for a specific area and hour can consist of a cleaning task within the specified area and time, which could be performed by a single robot. Similarly, a package delivery service across different areas can be decomposed into multiple delivery tasks for individual areas, each performed by a dedicated robot. The relationship between services and tasks is illustrated in Figure 2.

In the SC layer, users can request services by combining multiple tasks to be assigned to robots. For this purpose, robot vendors must register information about their robots with the crowdsourcing platform in advance, so that users can understand what types of tasks they can request. The minimum information to be registered includes vendor identification, which links the robot to a specific vendor operation system, the types of tasks the robot can perform (e.g., cleaning, delivery, security), the areas it can cover, and cost information (e.g., per unit of time or per unit of distance). Based on this information, the platform can provide users with details about the tasks and list of robots available in their desired areas and times. Consequently, users can flexibly build their own services based on the task information as well as the associated robot information.

C. Task Plans and Task Reports

Once a user registers a service, the platform assigns the tasks to the associated robots. The vendor operation system then prepares the robot to operate in the area and at the hour specified in the task information. At that time, the vendor operation system registers a *task plan*, which describes the time schedule and planned travel routes of the robot operation, with the crowdsourcing platform.

Task plans are used to manage the status of robots. Once a task plan is registered, the corresponding robot becomes occupied with the task, and thus no other tasks can be assigned during the designated hours.

When a robot completes a task, it registers a *task report* with the crowdsourcing platform. Based on this information, users can check the records of task execution and confirm the completion of the task. This information can also be used for purposes such as payments.

IV. Collision Avoidance by Spatial-ID and Space of Attention

One of the main challenges in coordinating robots is avoiding congestion and collisions. Several studies have addressed this topic, such as [6] and [5]. A study in [2] recently discusses a method of mediating navigation routes among vendors using common operation maps based on the assumption that maps for navigating robots are shared among vendors. However, different vendors use different maps in practice, and these maps or their creation processes may contain confidential technology and know-how of each company. Therefore, the assumption that maps can be shared among different vendors is not realistic.

In this section, we propose a method to avoid collisions without the assumption that maps can be shared among vendors. To this end, we introduce the concepts of Spatial-ID and SoA.

A. Spatial-ID

Several governmental organizations in Japan have jointly published a guideline [4]. This guideline provides a common framework and reference architecture for universally treating spatio-temporal information using a voxel-based data representation, which is called *Spatial-ID*. Spatial-ID is a mechanism that allows any space on earth to be designated by a voxel associated with a unique ID. It is regarded as an extension of slippy map tilenames [15] in the elevation direction.

The ID is represented by the format "$z/f/x/y$", where z is the zoom level, and f, x, y are indices in the elevation, longitude, and latitude directions, respectively. They are calculated using the following equations:

$$f = \left\lfloor \mathrm{alt} \cdot 2^{z-Z} \right\rfloor \tag{1}$$

$$x = \left\lfloor \frac{\mathrm{lon} + 180}{360} \cdot 2^{z} \right\rfloor \tag{2}$$

$$y = \left\lfloor \left(1 - \frac{\ln\left(\tan\left(\mathrm{lat} \cdot \frac{\pi}{180}\right) + \frac{1}{\cos\left(\mathrm{lat} \cdot \frac{\pi}{180}\right)}\right)}{\pi} \right) \cdot 2^{z-1} \right\rfloor, \tag{3}$$

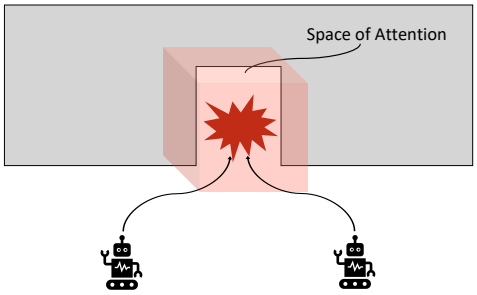

Fig. 4. An example of SoA.

where $Z = 25$, alt is the altitude in meters, and lon and lat are the longitude and latitude in degrees, respectively. The design ensures that the height of voxels is equal to 1 meter when $z = Z$.

Figure 3 illustrates an example of Spatial-IDs, where several voxels with different sizes, i.e., zoom levels, are depicted. One of the main advantages of using Spatial-ID is that the IDs can manage attribute information for arbitrary spaces. In other words, by using a uniquely calculated ID as a key, any information can be linked to arbitrary voxels[1]. In the figure, physical objects are retained as attribute information of voxels, but weather conditions and some tokens can also be managed. Additionally, it is possible to manage information that exists only for a specific time span by handling time information.

In the proposed system, the management of Spatial-IDs and their attribute information is implemented with smart contracts on top of the crowdsourcing platform. This system can be utilized not only for storing and referencing field traffic rules, such as right-of-way and no-stop zones, but also for managing Spaces of Attention (SoAs) as described below.

B. Space of Attention

Robots are usually equipped with built-in light detection and ranging (LiDAR) sensors and can autonomously avoid obstacles and walls while traveling. However, collisions and congestion are likely to occur when multiple robots traverse overlapping areas, especially narrow corridors and corners, simultaneously. Such a location that requires attention for navigating robots is referred to as an *SoA* in this paper. An example of an SoA is illustrated in Figure 4.

We employ Spatial-IDs to identify SoAs. These SoAs can be referenced and registered by robot vendors and the users of the crowdsourcing platform through smart contracts.

C. Collision Avoidance using SoAs

When a vendor operation system issues a task plan, it should also include information on the time, called *occupation hour*, that describes the planned time period for the robot to pass through SoAs. Subsequently, the crowdsourcing platform considers these SoAs to be in the *occupied state*. In other words, when a robot is scheduled to pass through SoAs, the vendor operation system applies for the occupation of the SoAs to the platform. No other robot can override the already-applied occupied state.

Other vendors can check whether the SoAs are in the occupied states. If they are, these vendors can recognize that robots owned by other vendors will pass through these SoAs during the designated occupation hour, and they can decide not to pass through the occupied SoAs. This process allows exclusive control over the passage of SoAs, thereby helping to avoid collisions. The occupied state and its occupation hour are managed as the attribute information of SoAs using Spatial-ID technology.

Note that some malicious or faulty vendor operation systems may attempt to cause conflicts in robot operation in SoAs even though these SoAs are in the occupied state. Nevertheless, such behaviors can be recorded in DLTs semi-permanently, and thus incentives should work to encourage cooperation.

V. Preliminary Investigation for Selecting a DLT

In this section, we discuss how we select a suitable DLT to implement the proposed crowdsourcing platform for service robots, referring to some simulation results from typical DLT test networks (testnets) as a case study. Note that since the proposed platform relies on smart contracts, our discussion will focus on DLTs that support smart contracts.

A. Public vs. Private

A public DLT functions like a decentralized database managed by multiple participants without any centralized authority. This type of DLT is open and accessible to anyone, allowing participants to interact with the ledger anonymously or pseudonymously.

Conversely, a private DLT, or a permissioned DLT, operates under the control of a selected group of participants or a single organization. Unlike public ledgers, access to a private ledger is restricted and requires authorization, which permits greater control over the transactions and interactions within the ledger. This type of ledger is especially useful for business or institutional applications.

While private DLTs offer controlled access and high efficiency in transaction processing, they limit the true potential of DLT by restricting participation and visibility. The study in [16] notes that existing private ledgers for commercial use, namely consortium ledgers, are state machine replication (SMR) systems and do not fulfill the requirements for a blockchain.

Moreover, from the viewpoint of ledger management, the challenge of equalizing influence among participants with varying numbers of nodes presents a significant administrative and technical burden in a private DLT network. In contrast, public DLTs democratize participation, where the cost and effort of node maintenance are distributed among a much larger group of stakeholders.

As a consequence, it may be suitable for our platform to leverage a public DLT to achieve a truly decentralized, accessible, and equitable environment. Nevertheless, some DLTs, including

[1]Typically, a database can be used to store attribute information and link it to Spatial-IDs.

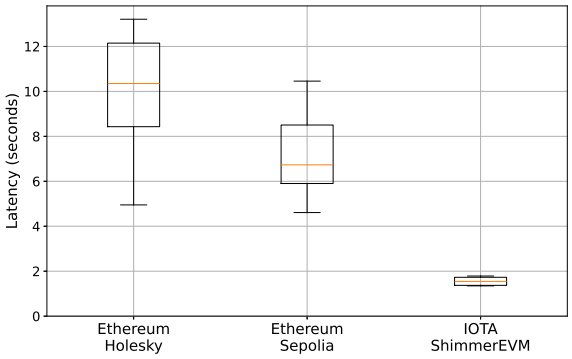

Fig. 5. Boxplots on the transaction approval latency for the testnets of different DLTs.

Ethereum [17], have the disadvantage of requiring transaction costs (gas fees) to execute smart contracts, the price of which can be high depending on external market factors. Therefore, DLTs that do not charge gas fees for executing contracts are also attractive. We discuss IOTA Tangle [18] as a representative example and compare it with the Ethereum blockchain in the following subsections.

B. Ethereum Blockchain vs. IOTA Tangle in Terms of Transaction Processing Speed

On our platform, it is anticipated that the registration of services and the applications for SoA occupation occur simultaneously, thus requiring a certain degree of real-time responsiveness to avoid transaction conflicts as much as possible. To this end, we compare the performance of transaction processing speeds between two typical DLTs, namely, Ethereum blockchain and IOTA Tangle, using their public testnets. Specifically, we evaluate the latency from the time a transaction is issued to the time it is approved and recorded in the ledger.

These two DLTs differ significantly in the structure of their ledgers. A blockchain is a hash chain of blocks containing multiple transactions in a way that does not allow for branching, i.e., forking. Tangle, on the other hand, is a partially-ordered directed acyclic graph (DAG) structure where the hash chain is composed of transaction units and there are multiple parent transactions. It should be noted that, since a blockchain is also a type of DAG strictly, we distinguish a blockchain as a *totally ordered set (toset)* and Tangle as a *partially ordered set (poset)* from the perspective of order theory. Although it is clear that the characteristics of Tangle suggest that its transaction processing speed is faster than that of a blockchain, to the best of our knowledge, no studies have quantitatively compared the latency across distinct DLTs and testnets, which is why we evaluate it in this paper.

The simulation was performed as follows: 1) implement a simple smart contract that can store arbitrary information; 2) deploy it on the testnets; 3) call the function to register the information to the contract; and 4) confirm that the transaction is recorded in the ledgers. The latency was obtained by calculating the elapsed time between processes 3 and 4. The computer used was a MacBook Air equipped with an Apple M2 chip and 16 GB RAM. The smart contract was implemented in Solidity, and the Python interface of the Web3 library [19] was used for accessing the deployed smart contract.

Figure 5 shows a boxplot illustrating the transaction approval latency over ten iterations for three different testnets: Sepolia and Holesky of Ethereum, and ShimmerEVM of IOTA. The results highlight the intrinsic differences in the design of DLTs as well as testnets.

Sepolia, designed primarily as a development environment for decentralized applications (DApps), leverages a permissioned Proof of Stake (PoS) consensus mechanism [20]. This network architecture supports a smaller validator set, which contributes to its rapid state synchronization. A median of the transaction latency recorded from the Sepolia testnet is 6.73 seconds, which can be attributed to its efficient handling of network states.

On the other hand, Holesky, which serves as a broad testing ground for validation and staking mechanisms, operates with an open validator set that includes over 1.5 million active validators [20]. Despite its capacity for a high volume of concurrent validations, Holesky exhibited a higher median transaction latency of 10.35 seconds. This increased latency likely stems from the complexities involved in coordinating a large number of validators.

ShimmerEVM, operating under IOTA Tangle, i.e., a poset-based DLT, demonstrates significantly lower transaction delays compared to those of the Ethereum testnets, with a median of 1.55 seconds. This is attributed to Tangle's unique design, which allows transactions to be processed in parallel, inherently supporting faster transaction confirmations.

It seems that the greater the transaction approval latency, the higher the risk of transaction collisions. We will conduct a theoretical analysis of this risk based on the above results in the subsequent subsection.

C. A Risk Analysis of Transaction Conflicts

Based on the above results regarding transaction approval latency, we now analyze the risk of transaction conflicts. Let us assume, for simplicity, that the event of a transaction issue follows a Poisson process, and denote the number of event occurrences during the time t as X_t. The probability that the event occurs exactly k times during t can be expressed as $P(X_t = k) = \frac{(\lambda t)^k e^{-\lambda t}}{k!}$, where λ represents the average number of events per second.

Conditioned on the fact that one transaction has already been issued, the probability P_c that one or more other transactions are further issued to call the identical smart contract within t can be calculated by $P_c = P(X_t \geq 1) = 1 - P(X_t = 0) = 1 - e^{-\lambda t}$.

Figure 6 shows the theoretical event probabilities of transaction conflicts based on the medians of the testnets as t with respect to λ. It is evident that the probability increases as λ increases. For the Ethereum testnets, namely Holesky and Sepolia, there are steep increases in the event probabilities;

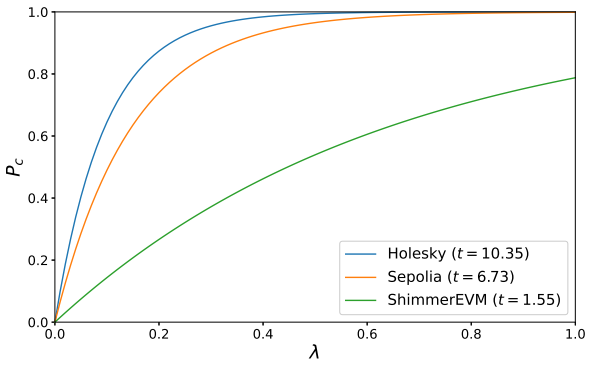

Fig. 6. Theoretical event probabilities of transaction conflicts based on the medians of the testnets as t.

$P_c = 0.5$, where half of the published transactions cause conflicts, occurs when $\lambda \simeq 0.067$ and 0.103, respectively. In our collision avoidance scenario, λ represents the average number of applications for the occupation of a specific Space of Attention (SoA) per second, and therefore depends on the frequency of service registration, the number of vendors participating, and the number of robots operated in the same area simultaneously. During the daytime, when robots are operated at a relatively high frequency, a λ value of about 0.1 is considered quite likely. In such situations, conflicts in half or even more of the published transactions could significantly degrade the efficiency of robot operation.

On the other hand, the characteristics of the curve for ShimmerEVM of IOTA are more gradual, and it is more than four times as robust as that of Sepolia to reach $P_c = 0.5$ in terms of λ. While the analysis presented focuses on the testnets, it should be noted that the distinct behaviors observed reflect the foundational designs of their respective main networks (mainnets). Although it cannot eliminate conflicts perfectly, IOTA Tangle seems to be the best option among the candidates of DLTs in this paper for the proposed platform.

VI. Conclusion

To enable multi-vendor robot collaboration, we proposed a crowdsourcing platform for mobile service robots based on DLT. The overall system architecture consists of three layers, and the proposed platform serves as a hub for siloed vendor-specific operating systems and as a bridge for users to request robots to perform tasks at the TC layer. We also proposed a collision avoidance mechanism for robots using Spatial-ID technology and the concept of SoA, which does not require the sharing of navigation maps among vendors. Based on the simulation results of transaction approval latency in the testnets, we discussed the appropriate DLTs to implement the proposed platform. We have concluded that IOTA Tangle is more promising than Ethereum for the implementation of the proposed platform. Our future work will involve practical analysis of the relationship between the spatial distribution of SoAs and the efficiency of robot operation.

References

[1] International Federation of Robotics, "World Robotics 2021 – Service Robots report released." [Online]. Available: https://ifr.org/ifr-press-releases/news/service-robots-hit-double-digit-growth-worldwide

[2] Digital Agency, Government of Japan, "Final Report of the Empirical Research on Cooperative Operation of Multiple Mobilities (Detailed Version)," May 2024, Japanese site. [Online]. Available: https://www.digital.go.jp/policies/mobility

[3] "Open-RMF." [Online]. Available: https://www.open-rmf.org

[4] Information-Technology Promotion Agency, "A Guideline for 4-Dimensional Spatio-Temporal Information Platform (ver. γ) | Digital Transformations in Industries and a Society," Feb. 2024, Japanese site. [Online]. Available: https://www.ipa.go.jp/digital/architecture/guidelines/4dspatio-temporal-guideline.html

[5] W. Zhao, R. Lin, S. Dong, and Y. Cheng, "A Study of the Global Topological Map Construction Algorithm Based on Grid Map Representation for Multirobot," *IEEE Transactions on Automation Science and Engineering*, pp. 1–14, 2022.

[6] A. Mokhtar, N. Murphy, and J. Bruton, "Blockchain-based Multi-Robot Path Planning," in *2019 IEEE 5th World Forum on Internet of Things (WF-IoT)*, Apr. 2019, pp. 584–589.

[7] S. Salimi, P. T. Morón, J. P. Queralta, and T. Westerlund, "Secure Heterogeneous Multi-Robot Collaboration and Docking with Hyperledger Fabric Blockchain," in *2022 IEEE 8th World Forum on Internet of Things (WF-IoT)*, Oct. 2022, pp. 1–7.

[8] J. Li, J. Wu, J. Li, A. K. Bashir, M. J. Piran, and A. Anjum, "Blockchain-Based Trust Edge Knowledge Inference of Multi-Robot Systems for Collaborative Tasks," *IEEE Communications Magazine*, vol. 59, no. 7, pp. 94–100, Jul. 2021.

[9] S. H. Alsamhi and B. Lee, "Blockchain-Empowered Multi-Robot Collaboration to Fight COVID-19 and Future Pandemics," *IEEE Access*, vol. 9, pp. 44 173–44 197, 2021.

[10] E. C. Ferrer, E. Jiménez, J. L. Lopez-Presa, and J. Martín-Rueda, "Following Leaders in Byzantine Multirobot Systems by Using Blockchain Technology," *IEEE Transactions on Robotics*, pp. 1–17, 2021.

[11] M. Li *et al.*, "CrowdBC: A Blockchain-Based Decentralized Framework for Crowdsourcing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 30, no. 6, pp. 1251–1266, Jun. 2019.

[12] G. Han, Z. Feng, Y. Xu, X. Xue, and S. Chen, "Building a Decentralized Crowdsourcing System with Blockchain as a Service," in *2023 IEEE International Conference on Web Services (ICWS)*, Jul. 2023, pp. 196–205.

[13] M. Wang, T. Zhu, X. Zuo, M. Yang, S. Yu, and W. Zhou, "Differentially Private Crowdsourcing With the Public and Private Blockchain," *IEEE Internet of Things Journal*, vol. 10, no. 10, pp. 8918–8930, May 2023.

[14] H. Xiao, Z. Huang, Z. Xu, S. Yang, W. Wang, L. Zhong, and C. Xu, "Task-Driven Cooperative Internet of the Robotic Things Crowdsourcing: From the Perspective of Hierarchical Game-Theoretic," *IEEE Internet of Things Journal*, pp. 1–1, 2024.

[15] "Slippy map tilenames - OpenStreetMap Wiki." [Online]. Available: https://wiki.openstreetmap.org/wiki/Slippy_map_tilenames

[16] K. Saito *et al.*, "Requirement Analyses and Evaluations of Blockchain Platforms per Possible Use Cases," *arXiv:2103.03209 [cs]*, Mar. 2021. [Online]. Available: http://arxiv.org/abs/2103.03209

[17] V. Buterin, "Ethereum: A Next-Generation Smart Contract and Decentralized Application Platform." 2014. [Online]. Available: https://ethereum.org/content/whitepaper/whitepaper-pdf/Ethereum_Whitepaper_-_Buterin_2014.pdf

[18] S. Popov, "The Tangle," pp. 1–28, 2018. [Online]. Available: https://www.iota.org/research/academic-papers

[19] "Ethereum/web3.py." [Online]. Available: https://github.com/ethereum/web3.py

[20] Chainlink, "Sepolia vs Holesky - Comparing Ethereum Testnets," Dec. 2023. [Online]. Available: https://blog.chain.link/sepolia-vs-holesky-comparison/