2023 IEEE IAS Global Conference on Emerging Technologies (GlobConET)
Loughborough University, London, United Kingdom. May 19-21, 2023

2023 IEEE IAS Global Conference on Emerging Technologies (GlobConET) | 979-8-3503-3179-0/23/$31.00 ©2023 IEEE | DOI: 10.1109/GlobConET56651.2023.10150089

Decentralized Framework for Collection and Secure Storage of Google Street View Data: Case Study

Sanjaya Mallikarachchi, Bonnie Ho, Iyad Kanj
Jarvis College of Computing & Digital Media, School of Computing, DePaul University, Chicago, IL, USA.
{smallika, xhe23, ikanj}@depaul.edu

Oshani Seneviratne
Department of Computer Science, Rensselaer Polytechnic Institute, Troy, NY, USA.
senevo@rpi.edu

Isuru S. Godage
Department of Engineering Technology & Industrial Distribution, Texas A&M University, College Station, TX, USA.
igodage@tamu.edu

***Abstract*—Image data plays a pivotal role in the current data-driven era, particularly in applications such as computer vision, object recognition, and facial identification. Google Maps® stands out as a widely used platform that heavily relies on street view images. To fulfill the pressing need for an effective and distributed mechanism for image data collection, we present a framework that utilizes smart contract technology and open-source robots to gather street-view image sequences. The proposed framework also includes a protocol for maintaining these sequences using a private blockchain capable of retaining different versions of street views while ensuring the integrity of collected data. With this framework, Google Maps® data can be securely collected, stored, and published on a private blockchain. By conducting tests with actual robots, we demonstrate the feasibility of the framework and its capability to seamlessly upload privately maintained blockchain image sequences to Google Maps® using the Google Street View® Publish API.**

***Index Terms*—Cyber-Physical Systems, Swarm Robotics, Smart Contracts, Blockchain-based crowd-sourcing, blockchain databases**

I. Introduction

A. Motivation & Prior Work

Google Maps® has become a part of our daily routine of traveling. It tracks us on a map with the GPS data extracted from our mobile device in real time. Also, it comes with the well-known feature Google Street View® to enhance the navigation experience for the users. Google Street View® provides us a 360° first-person view in each GPS coordinate we step on. These Street View images are typically uploaded by the Google Street View® car or backpack. However, this feature is not available for all the GPS coordinates, especially places the Google Street View® car cannot access, such as narrow roads and parks. Using the backpack to cover those regions is also an inefficient approach. Anguelov et al. [1] show practical difficulties and the challenges that face Google collecting 360° images with the existing methods and equipment. The current centralized method is inefficient, and it is unable to keep the Google Street Views® up-to-date. Therefore we propose a distributed and decentralized mechanism (see Fig. 1) to collect Google Street Views® with the help of robotics, smart contracts, and blockchain technologies. In our framework, we propose an incentivized business model that can participate in Google Street View® collection using open-source robots. These incentive mechanisms have been proposed [2]–[4] alongside the crowd-sourcing models [5]. Aditya et al. [6] shows how robotics and blockchain can be integrated together in various applications. Hunhevicz et al. [4] propose a blockchain-based data collection model, where contributors can earn incentives by sharing the data peer to peer. Grey et al. [7] suggested using "Swarm Contracts" as a means of managing multi-agent robots by providing incentives to agents participating in specific tasks. They further extended the concept to manage warehouse management in [8].

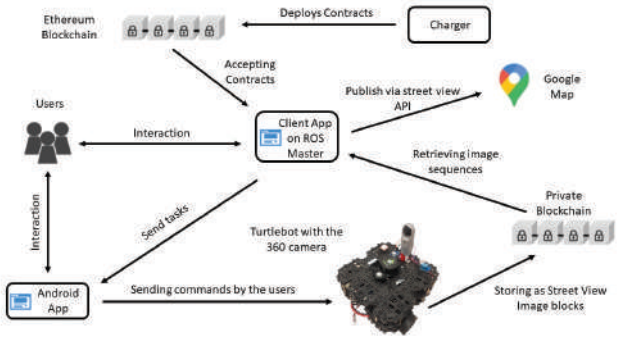

Fig. 1. The proposed distributed framework for collecting and storing Google Street View image data on a private blockchain.

Apart from the distributed and decentralized data collection mechanism, we propose a secure image storing mechanism similar to [9]. In our method, we store the image data directly on a block. A blockchain-based image storage system can provide a highly secure and transparent way to store and manage image data. The advantages of using a blockchain-based image storage system include security, decentralization,

This work is supported in part by the National Science Foundation (NSF) Grants IIS-2008797, CMMI-2048142, CMMI-2133019, and CMMI-2132994.

transparency, privacy, and data integrity. These systems can be particularly useful in applications where image data security and integrity are critical, such as healthcare, finance, and government. The primary advantage of a blockchain-based image storage system is that it provides a high level of security [9]–[13]. The distributed nature of the blockchain ensures that stored images are resistant to tampering or modification. A blockchain-based image storage system can provide a high level of data privacy [14]–[16]. The use of cryptographic techniques ensures that only authorized parties can access the data, and the use of smart contracts can enforce access controls and other privacy policies. The work reported in [17] proposes a privacy-preserving system for multi-source image retrieval in edge computing to address the privacy risks of centralized image-storing schemes. The use of a blockchain provides an immutable audit trail of all transactions, making it easy to trace the history of each image and ensure its authenticity. This can be particularly important in applications where compliance or regulatory requirements must be met. The work reported in [18] proposes a method for using blockchain technology to enhance the forensic analysis of digital images. The method is based on the concept of "permissible transformations", which are a set of image modifications that do not significantly alter the visual content of the image but can be used to enhance its forensic properties. A blockchain-based solution to improve drug traceability in the healthcare supply chain is reported in [19]. The authors argue that the current paper-based system is inefficient and prone to errors and propose a system based on blockchain technology to store information about drug transactions, including the manufacturer, distributor, and retailer.

B. Contributions

The proposed framework shown in Fig. 1 addresses the need for a secure and decentralized way to collect street view data for Google Maps. It utilizes open-source robots and a protocol for securely storing the resulting image sequences on a private blockchain. The work builds upon prior research on smart contract-based agent collaboration frameworks via blockchain technology [8], [20], [21]. The private blockchain ensures that the street-view image sequences remain tamper-proof and immutable, as we utilize the concept of forking to maintain independent image sequences on the blockchain. This ensures the security and integrity of the image data without the risk of alteration or deletion without network consensus. To validate the framework, we conduct experiments using actual robots to capture 360-degree images of street views. We verify the image blockchain to ensure the integrity of the data, and then conduct batch upload tests to Google Maps to ensure that the street view image sequences can be securely and efficiently uploaded to the platform.

II. System framework

A. Robot, Hardware and Software specifications

We use Turtlebot3 Waffle Pi model [22] as the robotic hardware (see Fig. 2) to navigate through the paths and capture the 360° images. We attached a 360°camera LG-G5 Friends 360°camera to capture the 360° street views (see Fig. 2). Further, we attached a GPS module (HiLetgo GY-NEO6MV2 NEO-6M) to the robot to record the GPS coordinates. This robot comes with a Single Board Computer (SBC) Raspberry Pi 3B+ and consists of dual motors handled by an open-source hardware controller (OpenCR). Open-source hardware can be easily integrated via General Purpose Input and Output (GPIO) on Raspberry Pi 3B+. The OpenCR controller has an accelerometer and a 3-axis gyroscope to provide the robot's orientation. SBC uses a Linux-based operating system and Linux-compatible driver to handle all the hardware. The original robot comes with an 8MP camera. We use it as optional visual feedback in navigation. We utilize the LiDAR sensor for Simultaneous Localization and Mapping (SLAM), which we later discuss under the navigation Sec. II-B.

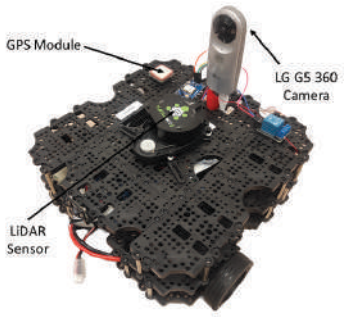

Fig. 2. Turtlebot with the LG- G5 360°camera, LiDAR sensor, and GPS module

Turtlebot is controlled by an operating system called Robotic Operating System (ROS). When configuring the SBC, the preferred version of ROS is installed. ROS provides the infrastructure to handle all the sensors and actuators of any robotic system. It primarily uses a Subscriber Publisher mechanism to communicate data within the nodes. ROS master is the main node that facilitates the services within ROS. ROS provides the ability to use community-supported algorithm packages to reuse without writing the code from scratch. Anyone can run these packages as a separate node and facilitate the robotic system by exchanging data. In our system, we run a separate client app to keep track of the deployed swarm contracts to Ethereum Blockchain and provide the task to the client Android app running on mobile, which we use for manual teleoperation.

B. Navigation Methods

We provide two navigation options for the robot to navigate. One option is manual teleoperation, where the user determines the path of the robot using the Android app. The Android app acts as a client node within the system. The Android app is utilized by integrating ROS Mobile [23]. In the second option (Semi-Automatic mode), the user is assisted with a benchmark search algorithm. In this mode, the user has to select the sub-goal point to reach the final destination. Each sub-goal point is

automatically reached by the search algorithm. We use SLAM to generate the real-time map for users to select the sub-goal points or to teleoperate. For this feature in the framework, we use the GMapping SLAM package to build a 2D map with LiDAR sensor data. As for the benchmark search algorithms, we use A-star, Greedy, and Dijkstra algorithms [23] to determine the path to each sub-goal point defined by the user. During the experiments, we evaluate the total search execution time and running times with the profits earned in each method.

C. Capturing, Storing and Publishing of 360° images

On Turtlebot, we installed The LG-G5 Friends 360° camera, as displayed in Fig. 2. This camera produced photospheres that were compatible with Google Street View®. Additionally, it used the Open Spherical Camera (OSC) API, which is a standard API found in most 360° cameras available in the market. We captured the images by sending HTTP requests through the OSC API. The 360° camera was connected to the internet via WiFi. To control the camera via OSC API and send HTTP requests, we implemented a separate capturing node. We also added a condition to capture an image for each 3-meter increment of the odometer reading of the robot, in compliance with Google Maps street view specifications

After capturing the images, we stored them in a private blockchain, as demonstrated in Fig. 4. To generate the hash per block, we used the actual image pixel data, timestamp, GPS coordinates, and previous hash. We concatenated these three elements into a single string, converting the pixel data into a string using Base64 string conversion. This conversion allowed us to convert the pixel data into a string and then back to the original image from Base64. Once we concatenated all of these strings, we used the SHA-256 hashing algorithm to convert the concatenated string into the hash and then added the block to the chain. This gave us a secure mechanism to store the captured 360° camera images while maintaining their integrity. Our protocol allowed for maintaining multiple image sequences per street, which was useful if someone preferred to use a different set of street views. This gave them the ability to switch between street view sequences. Maintaining street views can be confusing at junctions. To solve this issue, we used the concept of forking. We used the ending hash of one street view image chain to start new chains, creating multiple branches as illustrated in Fig. 6. New street-view image chains could be interchanged at these junctions, and we also could continue the chain without forking, as shown in Fig. 6. In this case, the two chains were independent and did not share any hash data.

The client app running on the ROS master PC extracted the images from the private blockchain proposed in the Fig. 4 and uploaded them batch-wise to Google Maps® using the Google Street View® Publish API to the corresponding GPS location. Google Street View® Publish API supported multiple languages such as Java, Python, and JavaScript. We used the Python API which allowed us to seamlessly integrate into the existing code base in Python. To use the API it was required to create a Google Cloud Platform® project and enable the

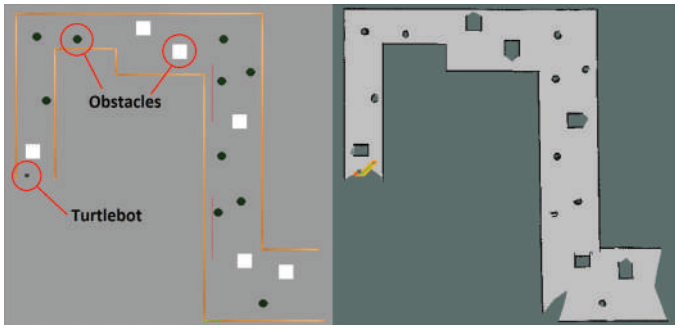

Fig. 3. Robot Navigation Path and Generated Map using SLAM during Semi Automatic-A* Navigation Method.

required APIs for the projects. Then with an existing Google account, we obtained a client ID and a client secret. It is important to highlight that the photosphere images should be compatible with the existing street views to upload the street view to Google Maps successfully.

III. The Swarm Contract-based Business Model

Blockchain and smart contract technology provide the functionality of securely storing data and automating business logic transactions more effectively than any other conventional technologies. Blockchains securely store data and maintain the sequence of data with the immutable feature. The applications that require distributed processing and decentralization can adopt these technologies to implement more secure and transparent systems. According to present statistics, the Ethereum blockchain [24] is the most widely used smart contract platform being used to execute smart contracts. It acts like a distributed computer network that provides the infrastructure to execute secure code capable of handling complex business transactions automatically.

A. Swarm Contract

We define a swarm contract which use to manage transactions between parties in our business case. The term swarm is used since it is the contract that manages robots as swarms. In the experiments, we use only one robot but the system is capable of simultaneously managing multiple robots as discussed in [21]. The swarm contract utilized in this framework is a smart contract written in Solidity programming language [25]. Typically, a smart contract consists of a series of if-then conditions. When the conditions are met the defined action is executed. This automates the business logic Sec. III-B within the economy. In this research, we use the local testing environment Ganache [26], for Ethereum smart contract deployments and execution. In a deployed contract the task is specified in terms of longitudes and latitudes (start and end point).

B. The Business Model

The aforementioned swarm contract automates the business logic by performing transactions between parties in each scenario listed (see Table I). In this simple business logic, there are two main parties. The first party, who is paying for

TABLE I
BUSINESS SCENARIOS AND THEIR HANDLING BY SWARM CONTRACTS.

Scenario	Description	Swarm contract condition
1	The user accepts the contract and completes the task. (ideal scenario)	Release the full collateral amount and the incentive to the user.
2	The user accepts the contract and cancels before deadline without completing.	Penalizing the user for cancellation by deducting a percentage from the collateral.
3	The user accepts the contract, and do not copmlete before the deadline.	Penalizing the user for incompletion by deducting a percentage from the collateral.

the job mentioned in the swarm contract collects street views. This role we defined as *charger*. In our case study, Google Inc. can act as the *charger*. The other party is the users who accept the contract and get paid upon completion of the tasks. When a user is accepting a contract a percentage task value is charged as a *collateral*. l. It prevents from fraudulent users accepting the contracts since they lose the collected *collateral* amount if they are unable to complete it. Once the task is completed the user will receive the specified number of incentives on the contract as well as the collateral borrowed by the contract upon acceptance. These payment functions can only be invoked by the person who has deployed the contract to the blockchain in our case the *charger*.

More parties can be introduced to the business model depending on the nature of the business case. A possible party would be *adjudicators* who evaluate the completion of the contracted task as described in [7]. The payment function invoking privileges can be modified by giving the invoking right to both *adjudicators* and *charger*.

IV. EXPERIMENTS

The effectiveness of the framework is evaluated using simulations and real-world experiments with the actual robots. Simulation provides the ability to test the navigation experiments multiple times without overusing the actual hardware, while the actual tests give the ability to identify real-world specific problems. In the tests, the robot has to navigate through the given path 3 and capture the 360° views. Actual 360° views are only captured during the tests and recorded into the private blockchain. The performance of the navigation method is evaluated with the profits earned in each method.

A. Simulation Tests

The objective of performing simulation tests is to make sure the robotics technologies that are used within the framework give an approximation of what to expect in real-world experiments. We used the Gazebo physics simulation client to test the robot under physics conditions using the Open Dynamics Engine (ODE). During the real-world tests, the 360° images are captured. While capturing the images are stored in the blockchain and retrieved and published as described in II-C. Following are the four navigation methods that are tested in simulation and real-world tests.

- **Nav. Method 1:** Semi-automatic with A-star algorithm
- **Nav. Method 2:** Semi-automatic with Greedy algorithm
- **Nav. Method 3:** Semi-automatic with Dijkstra algorithm
- **Nav. Method 4:** Manual Teleoperation

The path indicated in the (see Fig. 3) is used in the both simulation and the real-world tests. The entire path is 52 m long and search algorithms are used to find the path in between sub-goal points provided by the user in the semi-automatic mode. When providing the sub-goal points the points should be within the LiDAR sensor range, of 5 m and the map is generated using SLAM while the robot is reaching the sub-goal points (see Fig. 3). In manual teleoperation, mode user has to teleoperate to reach the end goal.

B. Real-World Tests

Real-world tests are performed using actual robotic hardware. By doing this it ensures how feasible it is to implement the proposed system framework in real-world conditions. Turtlebot covers the path using one of the navigation methods and captures 360° images and stores them in the private blockchain proposed. The odometer used in the robot has an error of 10%, which is negligible since the minimum travel distance to travel between one image to another is 3 m. Once the image sequence is captured, the ROS master node shown in Fig. 1 extracts the images from the blockchain and uploads them to Google Maps. The typical size of one image with 2K resolution ranges between $4 - 6\ MB$.

The cost and profit comparison is made for each navigation method discussed earlier. The Dijkstra-based navigation method showed the least efficiency compared to all other navigation methods, including manual teleoperation (see Fig. 5). The main reason is that it takes more time to generate a path, resulting in a higher idling time than other methods. So, it uses extra battery power compared to other methods. Even though the transactions use Ethereum (ETH), we do the costing in USD. To cover the path shown in Fig. 3, the total incentive is 10 USD.

Manual teleoperation was profitable in simulation and actual tests compared to search algorithm-supported navigation methods. An average of approximately 335 seconds is required to complete the prescribed path across all simulated and real-world attempts, with a deviation of 43.9 seconds. The storing process happens while the robot is navigating. Converting the captured image into Base64 string and storing the image on the proposed private blockchain takes around 100-200 ms per image. Except for the SA-Dijkstra method used in the navigation methods, all the other methods generated similar profit margins.

GMapping SLAM maps generated in real-world tests were noisy and inaccurately detected some objects, like the legs of the chairs near the walls of the tested path. These noisy maps made search algorithm-based navigation methods leading to generating complex paths in locally scanned regions where the sub-goals are defined. However, frequent restarts of the

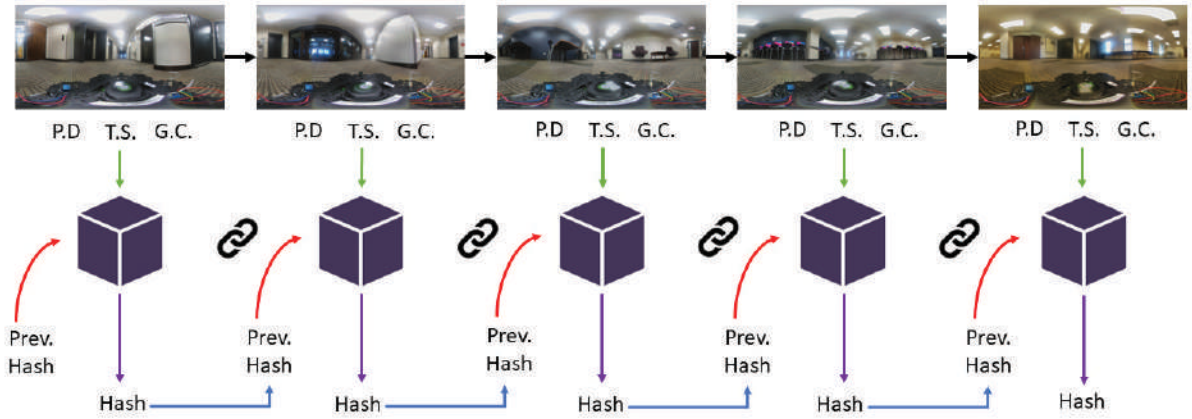

Fig. 4. Proposed blockchain-based secure storing mechanism. P.D. - Pixel Data. T.S. - Timestamp. G.C. - GPS Coordinate

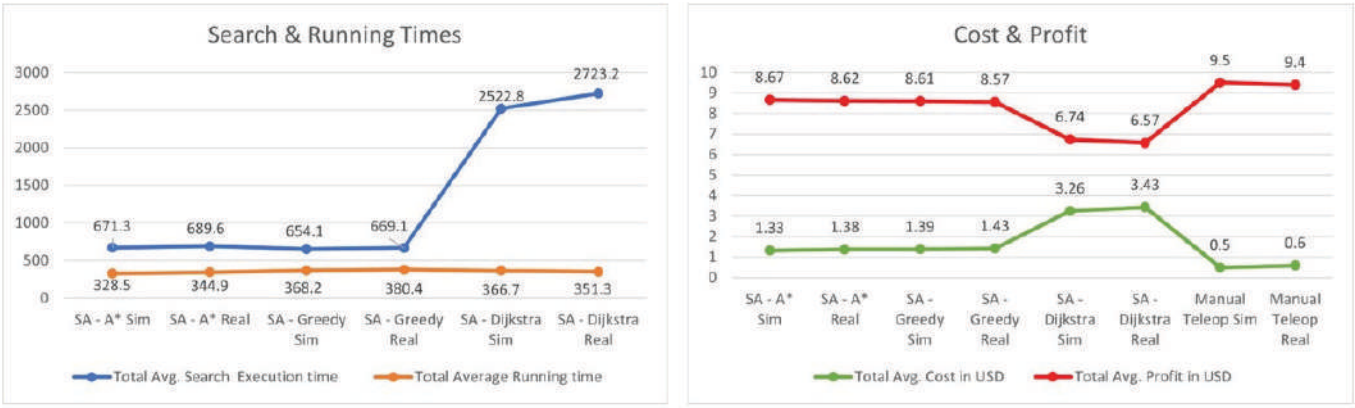

Fig. 5. Comparison of search and running times, as well as costs and profits, for each navigation method (Sim & Real) evaluated in the experiments.

GMapping SLAM algorithm helped us overcome the generated maps' noises. Once the images are stored in the proposed private blockchain, the client app running on ROS master retrieves the images from the Turtlebot by making an SSH connection. The entire part consists of 14 images. Retrieving the batch of 14 images (Base64 String to JPG) took around 1200 ms. It is negligible compared to the complete batch upload time, around 30 s. So the average time per cycle is around 366 s. The entire process algorithm has the complexity of $\mathcal{O}(n)$. However, this cycle time may vary depending on the length of the navigation path, the navigation method, and the size of the captured images.

V. Conclusions and Future Directions

In this paper, a distributed and decentralized approach to autonomously collecting Google Street View using open-source robots and a protocol for securely storing the resulting image sequences on a private blockchain was proposed. A private blockchain was utilized to maintain the integrity of the Google Street View image sequences and to handle any privacy issues stemming from storing image data publicly. The concept of forking was utilized to maintain independent Google Street View image sequences on the blockchain. This ensured that the image data remained secure and could not be altered or deleted without the consensus of the network. To test the efficacy of the framework, experiments were conducted using actual robots to capture 360-degree images of street views. The images on the blockchain were then verified to ensure the integrity of the data. The security of the storing can be further enhanced by signing the blocks by the users who collect them. Finally, batch upload tests were conducted to Google Maps to ensure that the Google Street View® image sequences could be securely and efficiently uploaded to the Google Maps®.

References

[1] D. Anguelov, C. Dulong, D. Filip, C. Frueh, S. Lafon, R. Lyon, A. Ogale, L. Vincent, and J. Weaver, "Google street view: Capturing the world at street level," *Computer*, vol. 43, no. 6, pp. 32–38, 2010.

[2] Y. He, H. Li, X. Cheng, Y. Liu, C. Yang, and L. Sun, "A blockchain based truthful incentive mechanism for distributed p2p applications," *IEEE access*, vol. 6, pp. 27 324–27 335, 2018.

[3] A. K. Shrestha, J. Vassileva, and R. Deters, "A blockchain platform for user data sharing ensuring user control and incentives," *Frontiers in Blockchain*, vol. 3, p. 497985, 2020.

[4] J. J. Hunhevicz, T. Schraner, and D. M. Hall, "Incentivizing high-quality data sets in construction using blockchain: a feasibility study in the swiss industry," in *ISARC. Proceedings of the International Symposium on*

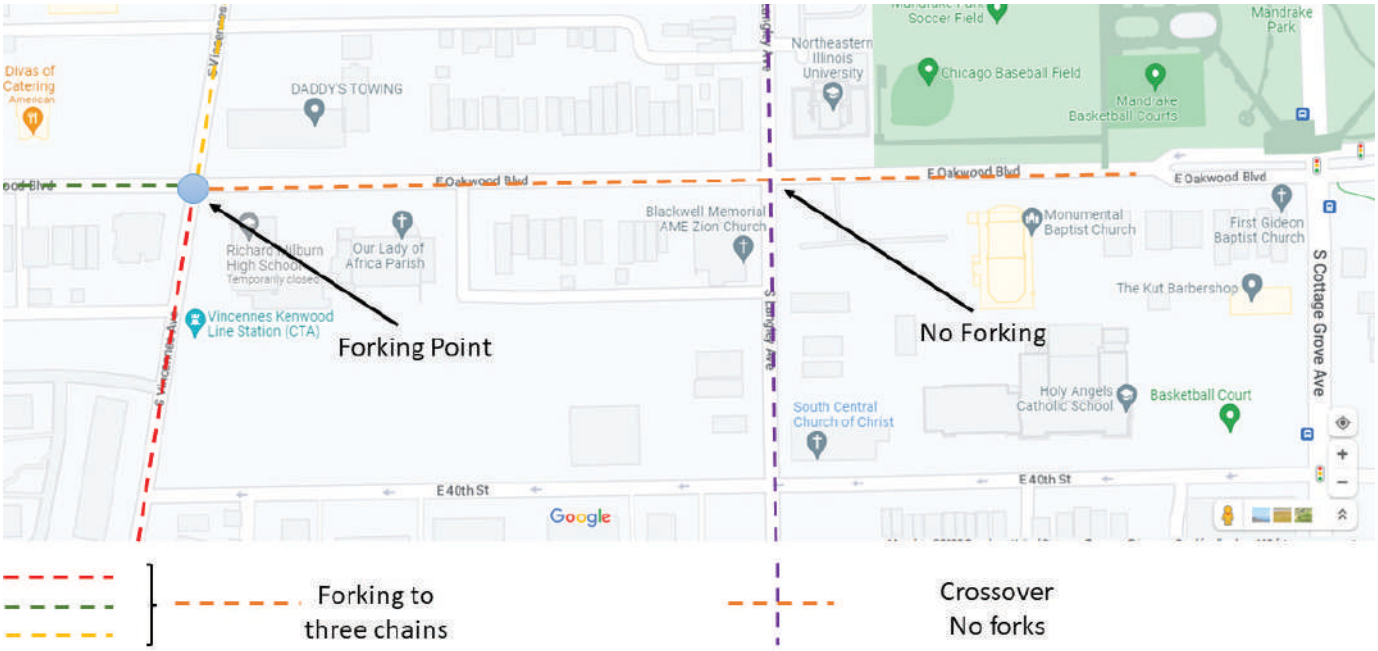

Fig. 6. How the storing protocol is employed at junctions. Two scenarios are possible: either the blockchain can be split to share the same hash (a fork) at a junction, or the junction can be ignored and the two streets can be maintained as independent chains.

Automation and Robotics in Construction, vol. 37. IAARC Publications, 2020, pp. 1291–1298.

[5] C. Li, X. Qu, and Y. Guo, "Tfcrowd: A blockchain-based crowdsourcing framework with enhanced trustworthiness and fairness," *EURASIP Journal on Wireless Communications and Networking*, vol. 2021, no. 1, pp. 1–20, 2021.

[6] U. S. Aditya, R. Singh, P. K. Singh, and A. Kalla, "A survey on blockchain in robotics: Issues, opportunities, challenges and future directions," *Journal of Network and Computer Applications*, vol. 196, p. 103245, 2021.

[7] J. Grey, I. Godage, and O. Seneviratne, "Swarm contracts: Smart contracts in robotic swarms with varying agent behavior," in *IEEE International Conference on Blockchain (Blockchain)*. IEEE, 2020, pp. 265–272.

[8] ——, "Blockchain-Based Mechanism for Robotic Cooperation Through Incentives: Prototype Application in Warehouse Automation," in *IEEE Blockchain Conference*. IEEE, 2021.

[9] K. Koptyra and M. R. Ogiela, "Imagechain—application of blockchain technology for images," *Sensors (Basel, Switzerland)*, vol. 21, no. 1, 2021.

[10] C. E. J. Singh and C. A. Sunitha, "Chaotic and paillier secure image data sharing based on blockchain and cloud security," *Expert Systems with Applications*, vol. 198, p. 116874, 2022.

[11] M. Shen, Y. Deng, L. Zhu, X. Du, and N. Guizani, "Privacy-preserving image retrieval for medical iot systems: A blockchain-based approach," *IEEE Network*, vol. 33, no. 5, pp. 27–33, 2019.

[12] J. Chi, Y. Li, J. Huang, J. Liu, Y. Jin, C. Chen, and T. Qiu, "A secure and efficient data sharing scheme based on blockchain in industrial internet of things," *Journal of Network and Computer Applications*, vol. 167, p. 102710, 2020.

[13] A. Ara, A. Sharma, and D. Yadav, "An efficient privacy-preserving user authentication scheme using image processing and blockchain technologies," *Journal of Discrete Mathematical Sciences and Cryptography*, vol. 25, no. 4, pp. 1137–1155, 2022.

[14] A. Malhi and S. Batra, "Privacy-preserving authentication framework using bloom filter for secure vehicular communications," *International Journal of Information Security*, vol. 15, pp. 433–453, 2016.

[15] S. Jiang, H. Wu, and L. Wang, "Patients-controlled secure and privacy-preserving ehrs sharing scheme based on consortium blockchain," in *2019 IEEE Global Communications Conference (GLOBECOM)*. IEEE, 2019, pp. 1–6.

[16] H. Jin, Y. Luo, P. Li, and J. Mathew, "A review of secure and privacy-preserving medical data sharing," *IEEE Access*, vol. 7, pp. 61 656–61 669, 2019.

[17] Y. Yan, Y. Xu, Z. Wang, X. Ouyang, B. Zhang, and Z. Rao, "Privacy-preserving multi-source image retrieval in edge computing," *IEEE Transactions on Services Computing*, pp. 1–14, 2022.

[18] R. Zou, X. Lv, and B. Wang, "Blockchain-based photo forensics with permissible transformations," *Computers & Security*, vol. 87, p. 101567, 2019.

[19] A. Musamih, K. Salah, R. Jayaraman, J. Arshad, M. Debe, Y. Al-Hammadi, and S. Ellahham, "A blockchain-based approach for drug traceability in healthcare supply chain," *IEEE Access*, vol. 9, pp. 9728–9743, 2021.

[20] S. Mallikarachchi, B. Ho, I. Kanj, O. Seneviratne, and I. Godage, "Decentralized data collection via swarm contracts: Study on crowd-sourced google maps," in *IEEE International Conference on Conference on Control, Automation and Robotics (ICCAR)*. IEEE, 2023, p. accepted.

[21] S. Mallikarachchi, C. Dai, O. Seneviratne, and I. Godage, "Managing collaborative tasks within heterogeneous robotic swarms using swarm contracts," in *IEEE International Conference on Decentralized Applications and Infrastructures (DAPPS)*. IEEE, 2022, pp. 48–55.

[22] (2022) Robotis e-manuel @ONLINE. [Online]. Available: https://emanual.robotis.com/docs/en/platform/turtlebot3/appendix_lds_02/

[23] X. Zhang, J. Lai, D. Xu, H. Li, and M. Fu, "2d lidar-based slam and path planning for indoor rescue using mobile robots," *Journal of Advanced Transportation*, vol. 2020, pp. 1–14, 2020.

[24] V. Buterin, "A next-generation smart contract and decentralized application platform," https://ethereum.org/en/whitepaper/, 2013.

[25] C. Dannen, *Introducing Ethereum and Solidity*. Springer, 2017, vol. 1.

[26] ConsenSys Software Inc. 2021. Ganache – One Click Blockchain. [Online]. Available: https://www.trufflesuite.com/ganache