Multi-robot architecture based on hybridized blockchain model

Rahul Harish Kumar, Gopalakrishnan Muthu Subramanian, Sahana Bailuguttu
Department of Electronics and Communication Engineering, Rashtreeya Vidyalaya College of Engineering, Bengaluru, India

Article Info

Article history:

Received Aug 3, 2023
Revised Oct 8, 2024
Accepted Oct 23, 2024

Keywords:

Hybridized blockchain model
Multi-robot path planning
Reciprocal velocity obstacle
Robot operating system
Secure communication
Semi-decentralization

ABSTRACT

Multi-robot systems (MRS) are groups of robots that coordinate to complete a given task. In communication-based systems, the integrity of the information shared between robots becomes highly important as any security threat due to a malicious node in the system can cause a chain reaction to compromise the entire system. This paper proposes a hybridized blockchain model-based architecture (HBMA) built on robot operating system (ROS) which offers a semi-decentralized environment into which any communication-based algorithm can be plugged in. A security monitoring system is also provided with the architecture that identifies and shuts down malicious robots while also sending out alerts about the threat. This architecture is used to create secured, coupled approaches to localization of multi-robots and multi-robot path planning. This approach is validated on both physical robots and simulations run on ROS.

Corresponding Author:

Rahul Harish Kumar
Department of Electronics and Communication Engineering, Rashtreeya Vidyalaya College of Engineering
Mysore Rd, RV Vidyaniketan, Post, Bengaluru, Karnataka 560059, India
Email: rahulhkumar9@gmail.com

1. INTRODUCTION

Multi-robot systems (MRS) are a rapidly growing field of robotics where a group of collaborative robots are used for a wide range of applications. These systems can find use in various tasks [1] including logistics, assembly, biomimetic research, search and rescue, where tasks are performed more efficiently and effectively than individual systems. This is due to both task division and reduced complexity of each individual robot. However, such systems come with their own challenges. Task allocation among different agents in the autonomous system, collision free path planning [2] for autonomous navigation and avoidance of malicious information in case of communication based systems are some of the primary issues in designing MRS. Security threats in communication based systems must be handled with utmost importance as any malicious information that travels through the system can compromise other aspects of the multi-robot environment leading to incorrect path planning and task allocation. These threats can be either endogenous (in form of malfunctioning transmitter or receiver in a robot), or exogenous (in form of injection attack or spoofing). This paper is an extension of the multi-robot security system based on robot operating system (ROS) and hybridized blockchain model (HBM) [3] that addresses similar issues. This paper however proposes an entire multi-robot architecture with multi-agent path planning [4] and localization integrated into the security system.

Multi-robot security systems become important in communication based path planning [5] as robots need to communicate their present coordinates to other robots in the system to make informed local trajectory corrections on their planned global paths from start to goal point. There are various approaches [5]–[7] to achieve collision free multi-agent path planning that are being tried and tested. Popular centralized approaches include safe interval path planning (SIPP) [8] and conflict based search (CBS) [9] methods. SIPP constructs a conflict graph based on robot trajectories and uses interval-based techniques to resolve conflicts.

CBS formulates the path planning problem as a constraint satisfaction problem and searches for collision free paths through a conflict resolution process. Other studies focus on priority based algorithms that use common single robot path planning algorithms like D* [10] as the backbone for their central node to plan individual paths for all robots. While both these methods do not rely on constant communication and are proven to work in dynamic environments, they suffer from common disadvantages of centralized approaches such as scalability, computational complexity and communication overhead. Additionally, in case a malicious agent attacks the centralized system, the entire system is compromised instantly. On the other hand, decentralized approaches like reciprocal velocity obstacle (RVO) [11] and nonlinear model predictive control (NMPC) [12] algorithms are scalable, adaptable and most importantly have enhanced tolerance to malicious information as each robot plans its own path independently. This paper proposes a semi-decentralized approach to integrate these path planning algorithms into the hybridized blockchain model-based architecture (HBMA). It also modifies the algorithms by creating an optimal communication method among robots to reduce the communication overhead while the robot paths are coupled, dynamic and secure by the blockchain model.

2. BACKGROUND

2.1. Semi-decentralized architecture using ROS

ROS is an open-source middleware that provides a collection of libraries and tools for building and managing robotic systems. One of the primary advantages of using ROS is the communication pipeline that it provides. It enables communication between different components of a robot system, such as sensors, actuators, and higher-level control algorithms. ROS supports a distributed architecture, allowing for modular development. The ecosystem enables users to record and visualize data using packages like rosbag and ROS visualization (rviz). A centralized ROS server connects multiple nodes which can communicate with each other using ROS messages on the publisher-subscriber model.

This feature of ROS is used to create a semi-decentralized architecture. Each robot has state information about itself through proprioceptive or exteroceptive sensors. Information regarding other robots is communicated through the subscribed ROS topics. As a result of this happening on all robots in the multi-robot system in parallel fashion, all robots have information about all other robot states directly or indirectly. This gives each agent the capability to make decisions for itself. The multi-robot system can be used to perform various tasks [13] such as mapping, path planning and task coordination, while each robot is executing independently taking decisions by itself and simultaneously communicating and cooperating with all other agents in the system. This gives the architecture a decentralized characteristic.

2.2. Hybridized blockchain model

Blockchains are used as distributed ledger systems to encrypt and secure data. Traditionally, they consist of participating nodes which keep track of transactions and are grouped as blocks which are linked horizontally by a chain. Taking inspiration from the same model for securing MRS, this paper came up with a HBMA which consists of parallel blocks belonging to individual robots that are linked both vertically and horizontally as shown in Figure 1.

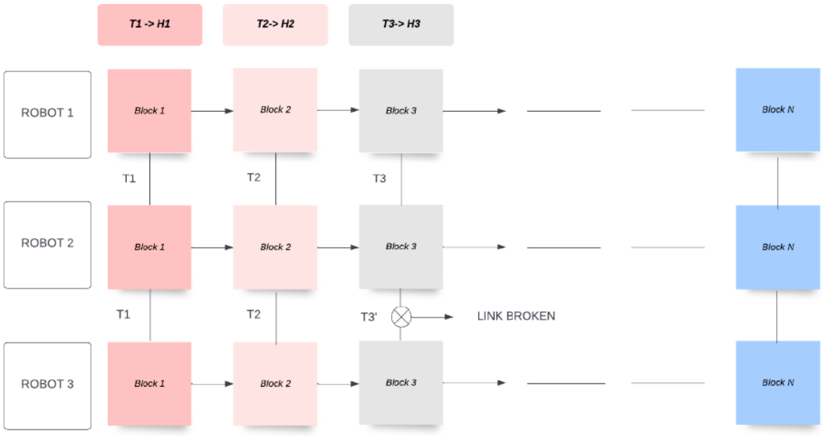


Figure 1. Same hash is generated at every instant when the system is secure. When a malicious robot publishes wrong information as represented by T3’, the hash generated by it will be different resulting in breaking of the vertical chain

The HBM is first made up of horizontally linked blocks corresponding to a single robot in the multi-robot system. The hash generated by each block depends on the received transaction at that particular instant. These transactions are made up of the sensor reading of the robot and messages received from other robots in the system. The messages from other robots are shared via subscribed ROS topics. Each robot publishes the state information obtained from the sensor readings to all the other robots in the system. The hashes generated by each robot are expected to be the same at every instant when all robots are working as properly. However, when a robot becomes malicious, it generates a hash by combining its original information with legitimate transactions received from other clients. The malicious robot shares incorrect transactions with the rest of the system due to endogenous or exogenous security threats, causing all other robots to generate a different type of hash. Consequently, the malicious robot ends up with a hash that differs from the rest of the robots in the system. Similarly, when there are multiple malicious robots present in the system, they all produce different and unique hashes. The robots that are non-malicious, all receive similar transactions from the semi-decentralized architecture discussed in the previous section, and all generate the same hashes. Hence, it is possible to monitor the entire multi-robot system using a centralized security monitoring node in the ROS security package.

2.3. Multi-robot systems

MRS is a rapidly evolving field at the intersection of robotics, artificial intelligence, and distributed systems. MRS involve the coordination between autonomous robots to achieve either a common goal or perform complex tasks. In a multi-robot system, the robots are equipped with sensing, computation, and actuation facilities to co-operate with other robots to exchange information and solve problems by collaboration. Multi-agent path planning is used in discretized environments, which are grids and planar graphs. The robots are assumed to be point robots, which do not have motion constraints. They are also called holonomic robots. The problem in a multi-agent path planning system [14] is described when the number of agents are at start locations with predefined goal locations, in a known environment. The task is to find collision free paths for the agent from the start location to the goal location, simultaneously optimizing the objective of the system.

3. LOCALIZATION USING HYBRIDIZED BLOCKCHAIN MODEL

One of the applications for the proposed multi-robot system is localization and thereby mapping of the environment by multiple robots [15]. Each robot in the multi-robot system has access to its own coordinate information. This can be from sensors like global positioning system (GPS), odometry sensor, lidar, and fiducial marker detection. All the robots publish their recorded coordinates in form of messages on ROS topic. Simultaneously, all robots subscribe to the respective ROS topics holding coordinate information about the publishing robots. Hence the robot's own localization along with the information received from each of the other robots in the system make up the transaction list for the HBMA. This is depicted in Figure 2.

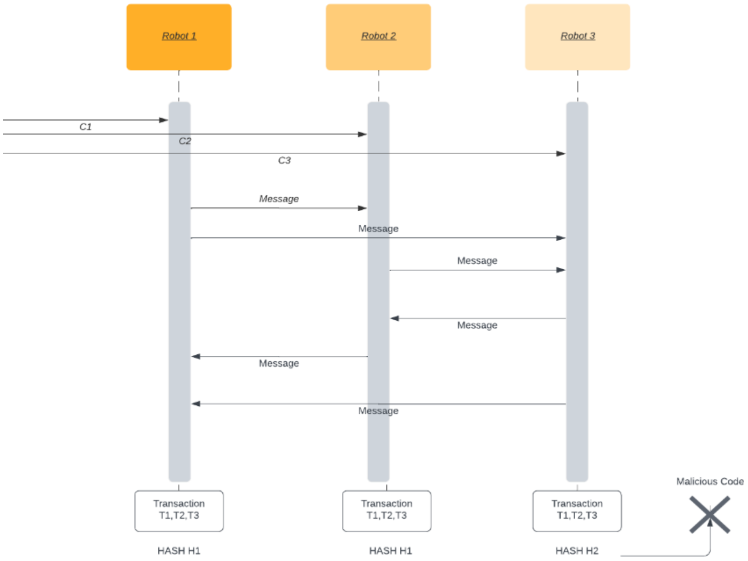
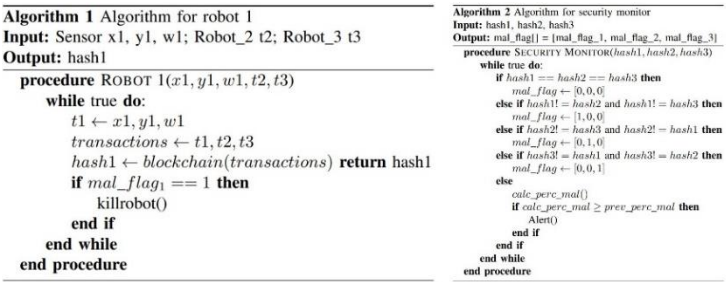


Figure 2. Localization information is decentralized and all robots produce hashes independently using the HBM, securing the system from malicious information

To secure the localization explained above, each robot client in the security package follows the algorithm shown in algorithm in Figure 3. All the transactions recorded or listed by the respective robot are appended into a list of transactions. This list is converted to a string format and passed to a hashing function that forms the core of the HBMA and gives it the characteristic of security in the multi-robot system. The chosen hash function can be MD5, SHA-1, SHA-256, and CRC32. The common property of all these hash functions is that even a very minute change in the transaction string will cause a noticeable change in the output hash. This ensures that the system is secured in real time and any malicious threat that seeps into the system is noticeable and can be recorded instantly. Whenever any threat is recorded by the security system, a flag is activated corresponding to the malicious robot, which forces the robot to stop moving and stop publishing malicious information out to the multi-robot system, thereby securing all the other agents.



```
Algorithm 1 Algorithm for robot 1
Input: Sensor x1, y1, w1; Robot_2 t2; Robot_3 t3
Output: hash1
  procedure ROBOT 1($x1, y1, w1, t2, t3$)
    while true do:
      $t1 \leftarrow x1, y1, w1$
      $transactions \leftarrow t1, t2, t3$
      $hash1 \leftarrow blockchain(transactions)$ return hash1
      if $mal\_flag_1 == 1$ then
        killrobot()
      end if
    end while
  end procedure
```

```
Algorithm 2 Algorithm for security monitor
Input: hash1, hash2, hash3
Output: mal_flag[] = [mal_flag_1, mal_flag_2, mal_flag_3]
  procedure SECURITY MONITOR($hash1, hash2, hash3$)
    while true do:
      if $hash1 == hash2 == hash3$ then
        $mal\_flag \leftarrow [0, 0, 0]$
      else if $hash1! = hash2$ and $hash1! = hash3$ then
        $mal\_flag \leftarrow [1, 0, 0]$
      else if $hash2! - hash3$ and $hash2! - hash1$ then
        $mal\_flag \leftarrow [0, 1, 0]$
      else if $hash3! = hash1$ and $hash3! = hash2$ then
        $mal\_flag \leftarrow [0, 0, 1]$
      else
        $calc\_perc\_mal()$
        if $calc\_perc\_mal \geq prev\_perc\_mal$ then
          Alert()
        end if
      end if
    end while
  end procedure
```

Figure 3. HBM and security monitoring algorithm

4. PATH PLANNING USING HYBRIDIZED BLOCKCHAIN MODEL

Similar to localization on HBMA, communication-based path planning approaches require robot state information belonging to all the agents in the multi-robot system in order to plan their paths in a decentralized manner. However, unlike the localization application, the size of information shared for path planning could be larger as the robot position, angular state, constraints and kinematic information among others is required to make informed decisions about the path planning steps to be taken. Hence, the communication overhead while conducting path planning for communication-based methods might be high, especially for applications having higher dimensions. However, coupling decentralized algorithms with communication of state information increases the reliability of the path planning algorithm. Hence a modified semi-decentralized model for communication in case of path planning applications is proposed, as shown in Figure 4. In this paper, two common decentralized path planning algorithms, RVO and NMPC are modified and implemented on the HBMA architecture according to the block diagram shown in Figure 4.

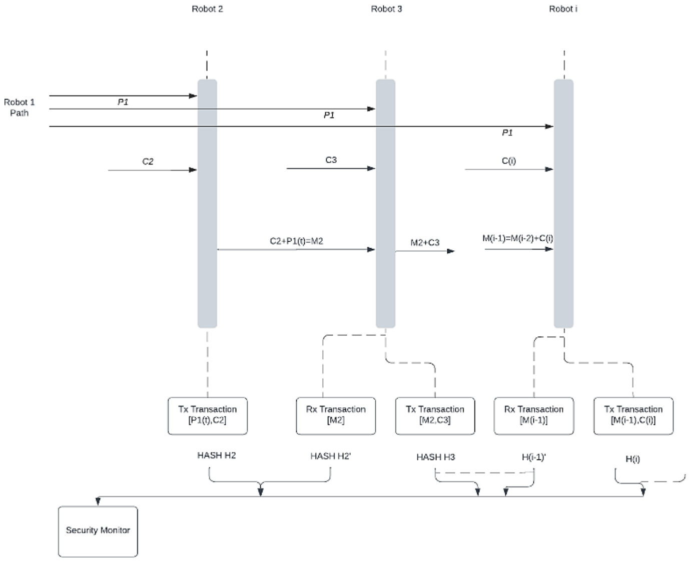


Figure 4. Modified decentralized, coupled path planning approach to reduce communication overhead

4.1. Reverse velocity obstacle

RVO is a decentralized, non-coupled approach to multi-robot path planning that results in collision free and oscillation free navigation [16]. Each robot implements path planning independently with no communication with other robots resulting in reduced communication overhead. However, there exists a possibility of collision in special cases such as kidnapped robots, where a single agent is moved to an arbitrary position away from the original trajectory due to some outside influence. This paper solves the above problem by modifying RVO [17] to a coupled, communication based method while securing it using the HBMA and ROS platform.

$$\theta_i = \tan inverse\ \frac{rob_x - obj_x}{rob_y - obj_y} \tag{1}$$

$$\theta_i = \sin reverse\ \frac{C \times ROB\_RAD}{\sqrt{(rob_x - obj_x)^2 + (rob_y - obj_y)^2}} \tag{2}$$

$$\theta_{left} = \theta_i + \phi_i \tag{3}$$

$$\theta_{right} = \theta_i - \phi_i \tag{4}$$

4.2. Nonlinear model predictive algorithm

NMPC is a decentralized path planning method [18] that accounts for full vehicle dynamics without depending on pre-planned trajectories. It is a model-based controller where complete vehicle dynamics are taken into account to plan possible commands at every instant of the robot movement. A controller formulation takes the system vector representing the state of the robot and control input vector as the input. These parameters are passed through a cost function to attain the control velocity for collision avoidance [19]. While this model is scalable and suitable for a dynamic environment while being decentralized, it is sensitive to model inaccuracies and is computationally complex. This paper modifies the working of traditional NMPC [20]–[22] execution to reduce the communication overhead and the average obstacle matrix size to be considered by the system, thereby solving the problem of computational complexity. The security monitor that comes with the HBMA also helps decrease the model inaccuracies by detecting malicious clients and taking action to mitigate wrong information spread.

$$J(u) = c1(u) + c2(u) \tag{5}$$

$$c1(u) = ||x - x_{ref}|| \tag{6}$$

$$d = [rob_x - obj_{ix}, rob_y - obj_{iy}] \tag{7}$$

$$c2(u) = \sum \frac{Q_c}{1 + e(k \times (d - 2 \times ROB\_RAD))} \tag{8}$$

5. SECURITY MONITORING

A central security monitoring [23] node is run by the HBMA when robots are communicating and planning for themselves in a decentralized fashion. The node follows simple logic of hash comparison and flag setting to detect malicious robots in the system and take action to kill them. As shown in algorithm in Figure 3, the security monitor node subscribes to hashes from all robots in the security system simultaneously. In case of a single robot being malicious, a mismatch in hashes can detect the malicious robot easily. The hash from a robot that does not match with the rest of the hashes received is declared malicious and the flag corresponding to that robot is enabled. This flag will take action to kill the malicious agent, *i.e.* the malicious robot is made to halt and stop transmitting information to the rest of the system.

$$mal\_perc = 1 - (\frac{k}{n}) \tag{9}$$

$$\frac{k2 - k1}{n} = \begin{cases} -ve, \text{growth in malicious bot count} \\ 0, \text{stable error} \\ +ve, \text{decline in malicious bot count} \end{cases} \tag{10}$$

When there is a mismatch of multiple hashes one after the other, malicious robots are killed in sequential fashion. However, when multiple robots become malicious parallelly, the percentage of robots under attack needs to be monitored. The percentage of malicious robots in the security system is monitored using (9), where k represents the count of robots with the majority hash and n represents the total number of robots in the system. The variable *mal_perc* indicates the percentage of malicious robots present at any given time. Additionally, the growth of malicious robots is also monitored using (10). In this equation, $k2$ represents the current count of malicious clients, while $k1$ represents the count of malicious clients in the immediate previous instance. If new malicious clients are introduced into the system at a faster rate than the security monitor can eliminate them, an emergency alert is triggered to prompt manual intervention and prevent the system from being compromised.

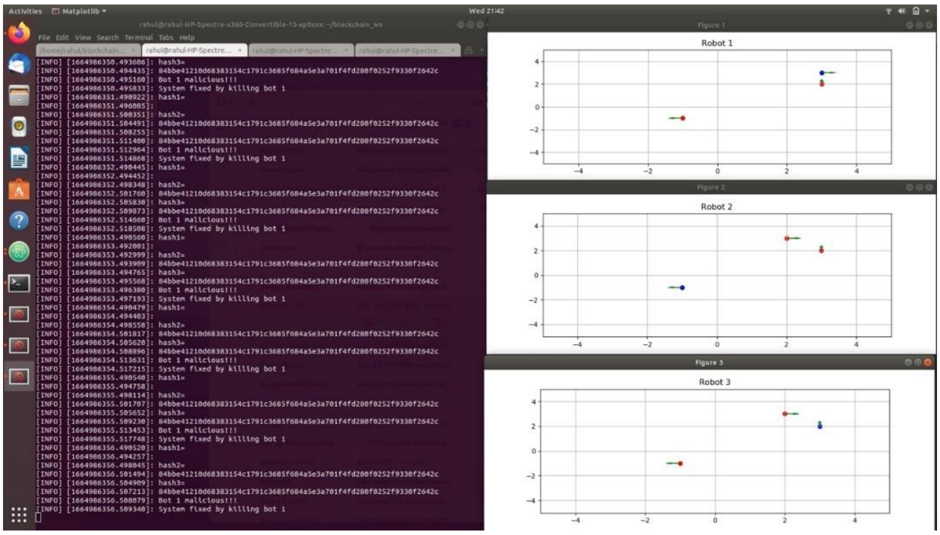
6. EXPERIMENTS AND SIMULATIONS

In order to demonstrate the HBMA concept proposed in this paper, both hardware and software setups were created. True to the minimalistic nature of swarm robots, small and lightweight robots were created. Each robot was of size 14×12×11 cm and weighed 400 g. Each robot had minimal processing power onboard. A NodeMCU board which runs on ESP8266 microcontroller was used to get instructions from a ROS master node via Wi-Fi. ROS nodes were initialized individually for each robot to make decisions in decentralized fashion. Raspberry Pi [24] can also be used for the same application and ROS nodes can run locally on the processor since ROS supports distributed architecture where different nodes can be connected through different machines. However, for this application, NodeMCUs were used and nodes were remotely run by sending commands to the robot through Wi-Fi. Each robot runs on 2 3.7 V Li-ion batteries, a pair of brushless outrunners (BO) motors and has a small servo for dropping objects. With no sensor on board, the robot relies on ArUco [25] markers stuck to the top plate and an overhead camera setup for its localization. These robots were used to demonstrate secured localization using HBMA.

7. RESULTS

7.1. Secured localization

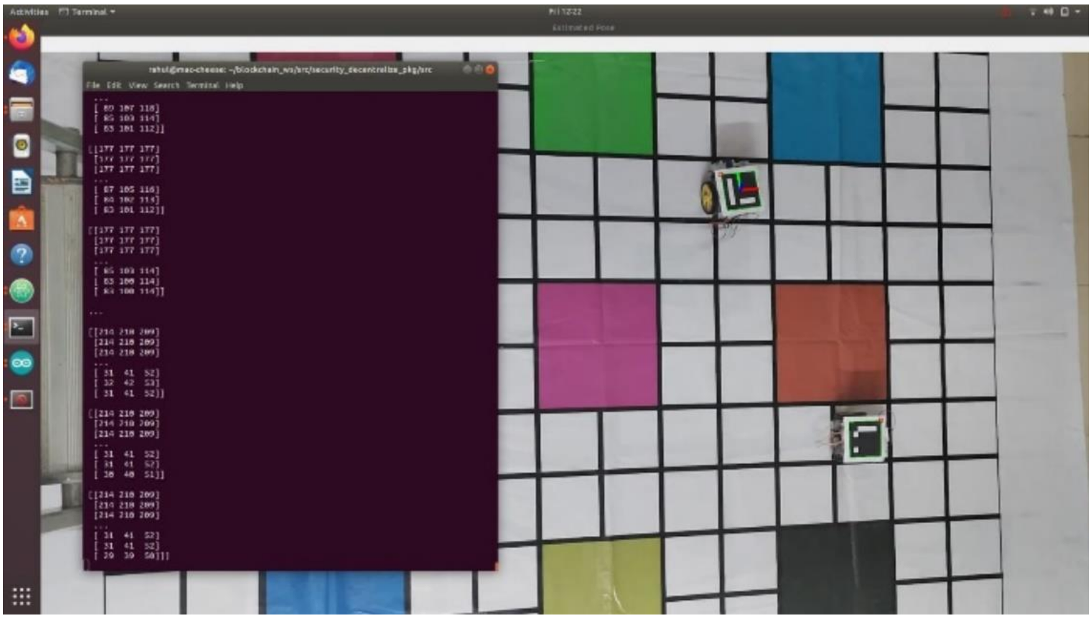
The ROS package was created according to the HBMA concept discussed in the previous sections. Different nodes were initialized for individual robots and the security monitor node was run parallel to the launching of all robot nodes. In the first stage, animation on matplotlib was used to visualize the secured localization as shown in Figure 5. Robot clients were publishing to other agents in the system and subscribing from other robots simultaneously using the publisher-subscriber model used to transmit ROS messages, and no delay or error in information exchange was observed. The security monitoring node continuously recorded hash comparisons and killed any malicious robots as soon as threat was detected. One such example where one of the robots in the 3 agent multi-robot system went malicious and the security monitor took action to kill the robot while sending out alerts can be seen in the Figure 5.


Activities
Matplotlib
Wed 21:42
rahul@rahul-HP-Spectre-x360-Convertible-13-ap0xxx: ~/blockchain_ws
File Edit View Search Terminal Tabs Help
/home/rahul/blockchain...
rahul@rahul-HP-Spectre...
rahul@rahul-HP-Spectre...
rahul@rahul-HP-Spectre...
Figure 1
Robot 1
Figure 2
Robot 2
Figure 3
Robot 3
4
2
0
-2
-4
-4
-2
0
2
4

```
[INFO] [1664986350.493686]: hash3=
[INFO] [1664986350.494435]: 84bbe41210d68383154c1791c3685f684a5e3a701f4fd280f0252f9330f2642c
[INFO] [1664986350.495160]: Bot 1 malicious!!!
[INFO] [1664986350.495833]: System fixed by killing bot 1
[INFO] [1664986351.490922]: hash1=
[INFO] [1664986351.496085]:
[INFO] [1664986351.500351]: hash2=
[INFO] [1664986351.504491]: 84bbe41210d68383154c1791c3685f684a5e3a701f4fd280f0252f9330f2642c
[INFO] [1664986351.508255]: hash3=
[INFO] [1664986351.511400]: 84bbe41210d68383154c1791c3685f684a5e3a701f4fd280f0252f9330f2642c
[INFO] [1664986351.512964]: Bot 1 malicious!!!
[INFO] [1664986351.514868]: System fixed by killing bot 1
[INFO] [1664986352.490445]: hash1=
[INFO] [1664986352.494452]:
[INFO] [1664986352.498348]: hash2=
[INFO] [1664986352.501760]: 84bbe41210d68383154c1791c3685f684a5e3a701f4fd280f0252f9330f2642c
[INFO] [1664986352.505830]: hash3=
[INFO] [1664986352.509873]: 84bbe41210d68383154c1791c3685f684a5e3a701f4fd280f0252f9330f2642c
[INFO] [1664986352.514660]: Bot 1 malicious!!!
[INFO] [1664986352.518508]: System fixed by killing bot 1
[INFO] [1664986353.490560]: hash1=
[INFO] [1664986353.492001]:
[INFO] [1664986353.492999]: hash2=
[INFO] [1664986353.493909]: 84bbe41210d68383154c1791c3685f684a5e3a701f4fd280f0252f9330f2642c
[INFO] [1664986353.494765]: hash3=
[INFO] [1664986353.495568]: 84bbe41210d68383154c1791c3685f684a5e3a701f4fd280f0252f9330f2642c
[INFO] [1664986353.496380]: Bot 1 malicious!!!
[INFO] [1664986353.497193]: System fixed by killing bot 1
[INFO] [1664986354.490479]: hash1=
[INFO] [1664986354.494403]:
[INFO] [1664986354.498558]: hash2=
[INFO] [1664986354.501817]: 84bbe41210d68383154c1791c3685f684a5e3a701f4fd280f0252f9330f2642c
[INFO] [1664986354.505620]: hash3=
[INFO] [1664986354.508896]: 84bbe41210d68383154c1791c3685f684a5e3a701f4fd280f0252f9330f2642c
[INFO] [1664986354.513631]: Bot 1 malicious!!!
[INFO] [1664986354.517215]: System fixed by killing bot 1
[INFO] [1664986355.490540]: hash1=
[INFO] [1664986355.494758]:
[INFO] [1664986355.498114]: hash2=
[INFO] [1664986355.501707]: 84bbe41210d68383154c1791c3685f684a5e3a701f4fd280f0252f9330f2642c
[INFO] [1664986355.505652]: hash3=
[INFO] [1664986355.509230]: 84bbe41210d68383154c1791c3685f684a5e3a701f4fd280f0252f9330f2642c
[INFO] [1664986355.513453]: Bot 1 malicious!!!
[INFO] [1664986355.517748]: System fixed by killing bot 1
[INFO] [1664986356.490520]: hash1=
[INFO] [1664986356.494257]:
[INFO] [1664986356.498045]: hash2=
[INFO] [1664986356.501494]: 84bbe41210d68383154c1791c3685f684a5e3a701f4fd280f0252f9330f2642c
[INFO] [1664986356.504909]: hash3=
[INFO] [1664986356.507213]: 84bbe41210d68383154c1791c3685f684a5e3a701f4fd280f0252f9330f2642c
[INFO] [1664986356.508079]: Bot 1 malicious!!!
[INFO] [1664986356.509340]: System fixed by killing bot 1
```

Figure 5. Matplot simulation of secured localization

Similarly in the next stage, the same secured localization concept was implemented practically. The custom robots described in the previous section were used for this purpose. ArUco markers on top of each robot were used to record localization information. An overhead camera streaming to each individual robot node was used to attain state information for the particular ID belonging to the robot node as shown in the Figure 6. The multi-robot system was able to localize all robots in a secured fashion. When error was induced manually to one of the robots, the system again detected the threat and killed the malicious client, *i.e.* stopped the malicious robot's movement and stopped publishing messages from that agent. Hence, it was successfully verified that the localization of agents in the multi-robot system was implemented in a secured, semi-decentralized fashion as per the objective and the HBMA prevented the entire system from being compromised when any security was induced.


Activities
Terminal
Estimated Pose
rahul@mac-cheese: ~/blockchain_ws/src/security_decentralize_pkg/src
File Edit View Search Terminal Help

```
 ...
 [ 89 107 118]
 [ 85 103 114]
 [ 83 101 112]]

[[177 177 177]
 [177 177 177]
 [177 177 177]
 ...
 [ 87 105 116]
 [ 84 102 113]
 [ 83 101 112]]

[[177 177 177]
 [177 177 177]
 [177 177 177]
 ...
 [ 85 103 114]
 [ 83 100 114]
 [ 83 100 114]]

...

[[214 218 209]
 [214 218 209]
 [214 218 209]
 ...
 [ 31  41  52]
 [ 32  42  53]
 [ 31  41  52]]

[[214 218 209]
 [214 218 209]
 [214 218 209]
 ...
 [ 31  41  52]
 [ 31  41  52]
 [ 30  40  51]]

[[214 218 209]
 [214 218 209]
 [214 218 209]
 ...
 [ 31  41  52]
 [ 31  41  52]
 [ 29  39  50]]]
```

Figure 6. Secured localization implemented on physical robots

7.2. Optimized multi-robot path planning

A separate ROS package was developed to test the architecture's compatibility with multi-robot path planning algorithms. The results were visualized and analysis was performed using a matplotlib animation. As explained in the previous sections, the RVO method and the NMPC method were both modified to be communication based, coupled and decentralized algorithms with reduced communication overhead. Both the algorithms were successfully tested for collision free autonomous navigation both in presence of static and dynamic obstacles. The robots were reaching their goals successfully while communicating and coordinating with each other through the publisher-subscriber model of ROS.

The average of computation time to calculate control velocity at each instant of navigation from start to goal of one particular robot was calculated and recorded as the number of obstacles increased. It was observed that RVO was a faster algorithm compared to NMPC, taking less time to compute the control velocities in presence of obstacles as seen in Table 1. However, it can be seen that NMPC is less sensitive to the increase in number of obstacles as compared to RVO. When the number of obstacles is increased from 1 to 4, the speed of the algorithm decreases 2.56 times for RVO and 1.17 times for NMPC approximately.

Table 1. Measurement of average computation time for different multi-robot path planning algorithms on HBMA in milli seconds

Number of obstacles	RVO	NMPC
1	0.63598	14.58845
2	1.00062	19.60314
3	1.33380	23.24947
4	1.63043	25.11543

Additionally, it can be observed from the Figure 7 that the path length taken by RVO is shorter than that of NMPC making it more optimal for navigation. The same conclusion can be made by observing Table 2. We can also see that NMPC improves much faster with decrease in number of obstacles, but RVO more or less takes the same path length with any number of obstacles. In a dynamic environment when the number of obstacles keep increasing, NMPC takes safer collision free paths compared to RVO, which has a tendency to reach very close to the obstacle.

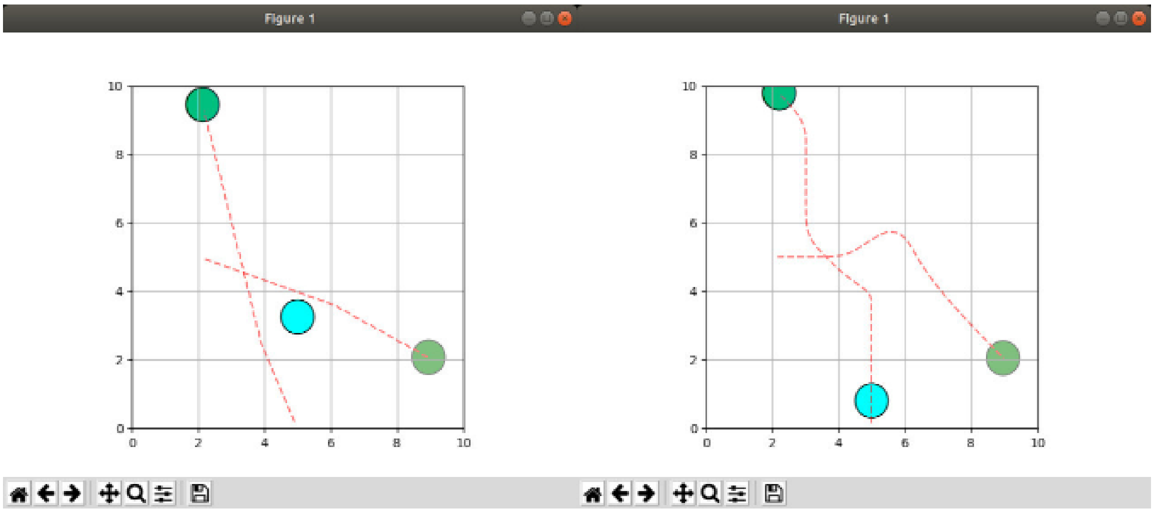


Figure 7. RVO and NMPC simulation on HBMA

Table 2. Measurement of path length for different multi-robot path planning algorithms on HBMA in milli seconds

Number of obstacles	RVO	NMPC
1	13.2828	7.5999
2	22.8879	7.5999
3	27.8010	7.5999
4	31.0863	7.7499

8. CONCLUSION

In this paper, a novel HBMA based architecture is proposed to run on a ROS package, into which algorithms for various multi-robot applications like localization, mapping and multi-agent autonomous navigation can be integrated. All decisions are implemented by individual robots running the algorithms in a decentralized fashion while a centralized security monitor is used to detect security threats and take action to mitigate the spread of malicious information. This proves to successfully safeguard the localization and path planning algorithms even when they are communication based and there is threat of malicious information being shared. The concept of fiducial marker detection for robot localization, RVO and NMPC methods for path planning are used for implementation of the discussed multi-robot applications. Additionally, modification is done to the two path planning algorithms to reduce the communication overhead and computational complexity of path decentralized path planning. The algorithms are tested on both Matplot Simulations and custom-made physical swarm robots, built using the ROS framework. Analyzing results from modified RVO and NMPC methods, various parameters such as speed of control velocity computation, sensitivity to number of obstacles and path length are used to compare the performance of the two algorithms. Future scope of the paper includes integrating more multi-robot applications such as trajectory planning and task allocation into the architecture while reducing the computational complexity of the algorithms running on each robot. Decentralizing the security monitor node along with the algorithms running on each robot will create a fully decentralized architecture that is safer from security threats and has reduced communication overhead. This paper has tested the localization and path planning algorithms for ground-based robots on both physical systems and on simulations. Future work can include aerial vehicles and manipulator systems to increase the dimensionality or degrees of freedom in the robots and conduct tests for the HBMA support on higher order systems. Mitigation of parallel security threats is another area of study future work can focus on.

ACKNOWLEDGMENT

We would like to thank the Department of Electronics and Communication Engineering, R.V. College of Engineering for their guidance in the formation of manuscript and assistance during the course of research. We would also like to thank Astra Robotics, R.V. College of Engineering for letting us use their workshop and providing necessary electronics for this project.

REFERENCES

[1] R. N. Darmanin and M. K. Bugeja, "A review on multi-robot systems categorised by application domain," in *2017 25th Mediterranean Conference on Control and Automation (MED)*, Jul. 2017, pp. 701–706, doi: 10.1109/MED.2017.7984200.

[2] Á. Madridano, A. Al-Kaff, D. Martín, and A. de la Escalera, "Trajectory planning for multi-robot systems: Methods and applications," *Expert Systems with Applications*, vol. 173, p. 114660, Jul. 2021, doi: 10.1016/j.eswa.2021.114660.

[3] R. H. Kumar and G. M. Subramanian, "Multi-robot security system based on robot operating system and hybridized blockchain model," in *2022 IEEE 3rd Global Conference for Advancement in Technology (GCAT)*, Oct. 2022, pp. 1–6, doi: 10.1109/GCAT55367.2022.9971918.

[4] J. Yu and S. M. LaValle, "Optimal multirobot path planning on graphs: complete algorithms and effective heuristics," *IEEE Transactions on Robotics*, vol. 32, no. 5, pp. 1163–1177, Oct. 2016, doi: 10.1109/TRO.2016.2593448.

[5] 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, doi: 10.1109/MCOM.001.2000419.

[6] Y. Zhang and Y. Meng, "A decentralized multi-robot system for intruder detection in security defense," in *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Oct. 2010, pp. 5563–5568, doi: 10.1109/IROS.2010.5652004.

[7] A. Bose, "Python implementation of a bunch of multi-robot path-planning algorithms," *GitHub*, 2022. https://github.com/atb033/multi_agent_path_planning (accessed Aug. 03, 2023).

[8] M. Phillips and M. Likhachev, "SIPP: Safe interval path planning for dynamic environments," in *2011 IEEE International Conference on Robotics and Automation*, May 2011, pp. 5628–5635, doi: 10.1109/ICRA.2011.5980306.

[9] S. Sharma and N. A. Menezes, "Multi robot path planning using priority based algorithm," in *2022 IEEE International Conference on Electronics, Computing and Communication Technologies (CONECCT)*, Jul. 2022, pp. 1–6, doi: 10.1109/CONECCT55679.2022.9865690.

[10] N. Malsa, V. Vyas, J. Gautam, R. N. Shaw, and A. Ghosh, "Framework and smart contract for blockchain enabled certificate verification system using robotics," in *Machine Learning for Robotics Applications. Studies in Computational Intelligence*, Singapore: Springer, 2021, pp. 125–138, doi: 10.1007/978-981-16-0598-7_10.

[11] Y. Zhang and Y. Meng, "Dynamic multi-robot task allocation for intruder detection," in *2009 International Conference on Information and Automation*, Jun. 2009, pp. 1081–1086, doi: 10.1109/ICINFA.2009.5205078.

[12] M. Kamel, J. Alonso-Mora, R. Siegwart, and J. Nieto, "Nonlinear model predictive control for multi-micro aerial vehicle robust collision avoidance," *arxiv.org/abs/1703.01164*, Mar. 2017.

[13] J. J. Roldán *et al.*, "Multi-robot systems, virtual reality and ROS: developing a new generation of operator interfaces," in *Robot Operating System (ROS). Studies in Computational Intelligence*, Cham: Springer, 2019, pp. 29–64, doi: 10.1007/978-3-319-91590-6_2.

[14] H. Tahir, M. N. Syed, and U. Baroudi, "Heuristic approach for real-time multi-agent trajectory planning under uncertainty," *IEEE Access*, vol. 8, pp. 3812–3826, 2020, doi: 10.1109/ACCESS.2019.2962785.

[15] A. Martinelli, F. Pont, and R. Siegwart, "Multi-robot localization using relative observations," in *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*, pp. 2797–2802, doi: 10.1109/ROBOT.2005.1570537.

[16] J. Snape, J. van den Berg, S. J. Guy, and D. Manocha, "The hybrid reciprocal velocity obstacle," *IEEE Transactions on Robotics*, vol. 27, no. 4, pp. 696–706, Aug. 2011, doi: 10.1109/TRO.2011.2120810.

[17] J. Snape, J. van den Berg, S. J. Guy, and D. Manocha, "Independent navigation of multiple mobile robots with hybrid reciprocal velocity obstacles," in *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Oct. 2009, pp. 5917–5922, doi: 10.1109/IROS.2009.5354821.

[18] H. Xiao and C. L. P. Chen, "Incremental updating multirobot formation using nonlinear model predictive control method with general projection neural network," *IEEE Transactions on Industrial Electronics*, vol. 66, no. 6, pp. 4502–4512, Jun. 2019, doi: 10.1109/TIE.2018.2864707.

[19] D. H. Shim, H. J. Kim, and S. Sastry, "Decentralized nonlinear model predictive control of multiple flying robots," in *42nd IEEE International Conference on Decision and Control (IEEE Cat. No.03CH37475)*, 2003, pp. 3621----3626 vol. 4, doi: 10.1109/CDC.2003.1271710.

[20] D. H. Shim, H. J. Kim, and S. Sastry, "Decentralized reflective model predictive control of multiple flying robots in dynamic environment," *Unpublised*, 2003, Accessed: Aug. 03, 2023. [Online]. Available: https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=e10c6e163885e012f302487c1e2b1beae99a80fb

[21] F. Künhe, J. Gomes, and W. Fetter, "Mobile robot trajectory tracking using model predictive control," in *II IEEE latin-american robotics ...*, 2005, pp. 1–7.

[22] J. Wurts, J. L. Stein, and T. Ersal, "Design for real-time nonlinear model predictive control with application to collision imminent steering," *IEEE Transactions on Control Systems Technology*, vol. 30, no. 6, pp. 2450–2465, Nov. 2022, doi: 10.1109/TCST.2022.3154370.

[23] L. E. Parker and B. A. Emmons, "Cooperative multi-robot observation of multiple moving targets," in *Proceedings of International Conference on Robotics and Automation*, vol. 3, pp. 2082–2089, doi: 10.1109/ROBOT.1997.619270.

[24] C. Williams and A. Schroeder, "Utilizing ROS 1 and the Turtlebot3 in a multi-robot system," *arxiv.org/abs/2011.10488*, Nov. 2020.

[25] L. Arcos, C. Calala, D. Maldonado, and P. J. Cruz, "ROS based experimental testbed for multi-robot formation control," in *2020 IEEE ANDESCON*, Oct. 2020, pp. 1–6, doi: 10.1109/ANDESCON50619.2020.9272073.

BIOGRAPHIES OF AUTHORS

Rahul Harish Kumar received his B.E. in electronics and communication engineering from RV College of Engineering in 2023. He is currently pursuing M.S. in robotics from University of Michigan, Ann Arbor. His interests are robotics, control systems, and perception. He can be contacted at email: rahulhkumar9@gmail.com.

Gopalakrishnan Muthu Subramanian received his B.E. in electronics and communication engineering from RV College of Engineering in 2023. His interests are embedded systems, computer science and machine learning. He can be contacted at email: muthusubramanian00@gmail.com.

Sahana Bailuguttu is an assistant professor in the Department of Electronics and Communication Engineering at RV College of Engineering, Bengaluru. She holds a Ph.D. in communication networks and a M.Tech. degree in computer network engineering from Dayananda Sagar College of Engineering. She has a teaching experience of 14 years and has taught the subjects basics of electronics engineering, analysis and design of digital circuits, computer communication networks, optical fiber communication, ARM processor to undergraduate and postgraduate students. Her current areas of interest are fault tolerance in networking and machine learning. She can be contacted at email: sahanab@rvce.edu.in.