Controlling Robots using Image Analysis and a Consortium Blockchain

Vasco Lopes[1], Luís A. Alexandre[1] and Nuno Pereira[1]

Abstract— **Blockchain is a disruptive technology, normally used within financial applications, however, it can be very beneficial in certain robotic contexts, such as when an immutable register of events is required. Among the several properties of Blockchain that can be useful within robotic environments, we find not just immutability but also data decentralization, irreversibility, accessibility and non-repudiation. In this paper, we propose an architecture that uses blockchain as a ledger, and smart-contracts for robotic control by using oracles to process data. We show how to register events in a secure way, how it is possible to use smart-contracts to control robots and how to interface with external algorithms for image analysis. The proposed architecture is modular and can be used in multiple contexts such as in manufacturing, network control, robot control, and others, since it is easy to integrate, adapt, maintain and extend to new domains, only requiring new tailored smart-contracts.**

I. INTRODUCTION

Automation is virtually everywhere, allowing for mundane and repetitive tasks to be automated. Robots are getting more common, especially in industrial tasks, where they are used for transporting, weight lifting, and repetitive tasks [1]. Moreover, the use of robots outside factories is increasing exponentially, where they have been useful in a wide range of tasks, such as medical, therapy aid, surveillance, military use, drone delivery, among others. Even though robots are extremely useful in solving many tasks, in Human-Robot Interaction (HRI) environments, most robots require human control, usually by supervision or teleoperation [2].

Many ways to control robots have been proposed over the years, [3], [4], [5]. In this paper, we propose a novel way of controlling robots by leveraging the power of blockchain and smart-contracts, which has been a disruptive technology in many fields, such as financial applications, internet-of-things, networks, and has seen a spike interest in Robotics, due to its information sharing and security capabilities. Our approach uses blockchain as a decentralized ledger, that stores information about robots and coupled sensors, like cameras, and also allows for the use of smart-contracts to define the logic of the robots control. By having oracles processing the data and inserting analytics into the blockchain, smart-contracts can use that information to control a robot. We demonstrate this architecture by controlling a real UR3 arm in a scenario that replicates a factory pick-and-place task. As the proposed model is modular, it is easy to integrate new modules that perform other tasks and the method can be used with any robot.

This proposal is specifically designed for factory environments. Where usually, humans and robots work alongside, wherein security and control of the robot actions are essential to detect the responsible entities of any fault, and ensure correct use. Our blockchain addresses these problems by logging all the interactions, and by enabling multiple deployments of algorithms to different robots presented in the blockchain. The ease of deployment is mostly needed for scenarios where the robot needs to execute multiple tasks within different settings, or robots that require specific algorithms that require extra computation power. More, by using the proposed method, one could replicate the algorithm used by a robot to many, simply by using the smart-contract that defines the task.

With our architecture, it is possible to deploy algorithms to the robots and control every setting, as well log the environment where humans are working.

In short, the contributions of this work can be summarized as follows:

- We propose a general architecture that allows any type of algorithms, including Artificial Intelligence (AI) ones, to interact with the blockchain and the robots in a secure and permissioned way by using smart-contracts.
- We demonstrate that it is possible to control robots with smart-contracts, by proposing a general approach that can be easily used to control any robot.

II. RELATED WORK

Some of the challenges that robotics has to overcome include security among robot networks, coordination and control of those networks and assurance that the robots actuate within certain ethical and moral limits [6]. Blockchain can be the solution to these problems, if used properly. Blockchain is mainly used for cryptographic currency transactions, but it is a very powerful tool, allowing for data decentralization and homogeneous registries among all the peers, immutability, irreversibility, accessibility and non-repudiation. Since its first idealization, this technology served many purposes, starting as the base for Bitcoin, and more recently being used in applications across multiple domains, especially for having an immutable ledger and to share the information, securely, across peers.

The integration of AI with blockchain has been focused on the development of marketplaces and the use of the data contained in it to do predictive analysis and conduct sales [7].

This work was partially supported by the Tezos Fundation through a grant for project RobotChain and partially supported by project 026653 (POCI-01-0247-FEDER-026653) INDTECH 4.0 – New technologies for smart manufacturing, cofinanced by the Portugal 2020 Program (PT 2020), Compete 2020 Program and the European Union through the European Regional Development Fund (ERDF), and by NOVA LINCS (UIDB/04516/2020) with the financial support of FCT-Fundação para a Ciência e a Tecnologia, through national funds.

[1]NOVA LINCS, Universidade da Beira Interior, Covilhã, Portugal

However, preliminary work presents the possible benefits of combining these two technologies [8], [9], especially with swarm robotics and robotic hardware [10]. The predominant benefits shown are the possible use of global information within robotic swarms in a secure and validated way, and a faster way to change the behaviour of the network, which will ultimately lead to higher productivity and easier maintenance. The authors of [11] present a conceptualisation of a possible integration of robotics and blockchain, which consists of a method to share critical data among robots in a secure way. Blockchain and robotics have been combined to develop methods to improve robot capabilities by detecting anomalies in robots [12], and by monitoring robotic workspace to avoid unwanted interactions and collisions with humans [13]. Both proposals use blockchain as a ledger and smart-contracts' technologies. In [14], an architecture to allocate tasks within multi-agent systems using a private blockchain is proposed. This proposal uses blockchain as a ledger and has a platform built outside of it to allocate tasks to the robots. In [15], knowledge processors and information stored on the blockchain are used to create coalitions of robots. Security in swarm robots has also been studied and in [16], it is shown how it is possible to use a blockchain with a reputation system to achieve consensus in robotic networks and at the same time, be robust enough to deal with byzantine robots [17].

The integration of blockchain with robotics is a growing field and once standards are defined, industrial and research applications will grow exponentially. These approaches differ from the one proposed in this paper, as we propose a method that goes beyond using blockchain as a ledger, and use its inner components to autonomously and automatically control robots using smart-contracts, providing a modular architecture capable of being used in multiple problems.

III. Blockchain and Robotics

A. Blockchain

The core idea behind the blockchain is to have a model capable of reaching a consensus over a network of computers in which the computers may be unreliable. Blockchain is a digital ledger that is implemented in a decentralized way that allows it to store sequences of blocks. These blocks hold information, which is, in most cases, representations of transactions. This technology was popularized mainly by using it as basis for cryptocurrencies. However, it is being used in a panoply of applications thanks to the smart-contract technology implemented over it and because it allows to decentralize data in a secure way. The vast majority of blockchain implementations either use Proof-of-Work (PoW) or Proof-of-Stake (PoS) as a consensus algorithm. PoW requires miners to solve a cryptographic puzzle in order to "mine" a block. In PoS, miners are selected to validate a block by different algorithms that take into account the miners stake (number of tokens they hold). There have been many consensus algorithms proposed both in the literature and in the industry that try to tackle problems regarding energy and time consumption spent on validating blocks. In short, Blockchain properties are: data replication, decentralization, accessibility, non-repudiation and transaction time-stamping, irreversibility and (pseudo) anonymity. These properties enhance security, by ensuring that, no centralized party controls the data, as it is decentralized and available to the network, which information is immutable and can't be changed by a user. More, blockchain also forces a user to sign the information sent to the chain, meaning that at a given time, is possible to see who sent a specific information to the blockchain. Also, every transaction sent to the network is added to a block, and if accepted into the blockchain, it is ensured that it is valid and secure.

B. RobotChain

Our approach uses RobotChain as blockchain [18]. RobotChain is a system to register robotic events in a trust and secure way and it's based on the blockchain developed by Tezos. It incorporates characteristics that are important for systems that are intended to work in sensitive environments. The first characteristic is the support of easier formal verification of smart-contracts, which are written in Michelson, a stack-based language, or in Liquidity and then compiled to the former. This is very important when working with robots, as it provides security that the code will do what it was defined to do. The second characteristic is the self-amending property that allows changes to be performed on the blockchain by voting on-chain, without the need to conduct hard-forks when a core feature needs to be changed. Finally, the last characteristic is the consensus algorithm, Delegated Proof-of-Work, where the energy and time needed to validate transactions are far more efficient than PoW, which is critical when dealing with huge amounts of data. RobotChain is a consortium blockchain, consisting of a private blockchain where it's unknown if the nodes can be trusted, which is useful when dealing with factories with multiple robots from different manufacturers. For this type of blockchain, the self-amendment process is useful when there is a need to change the protocol without the need for major changes. The need for registering robotic events in such a way is due to the fact that a human could alter logs, trying to blame an innocent robot for actions that it did not performed.

IV. Proposed Method

A. System Description

In this paper, we propose a general architecture that leverages the power of Blockchain and smart-contract technologies to allow its use in robotic contexts, where it can be used for virtually any robotic task, allows untrusted robots to communicate with external parties, and uses smart-contracts to register information in a secure way. The proposed method is composed of three components, the RobotChain, the Robot Stations and the Oracles, which can be seen in Figure 1. The first component - RobotChain, ensures that all the data inserted into it is decentralized, replicated through peers, accessible, immutable and it can't be repudiated, since every

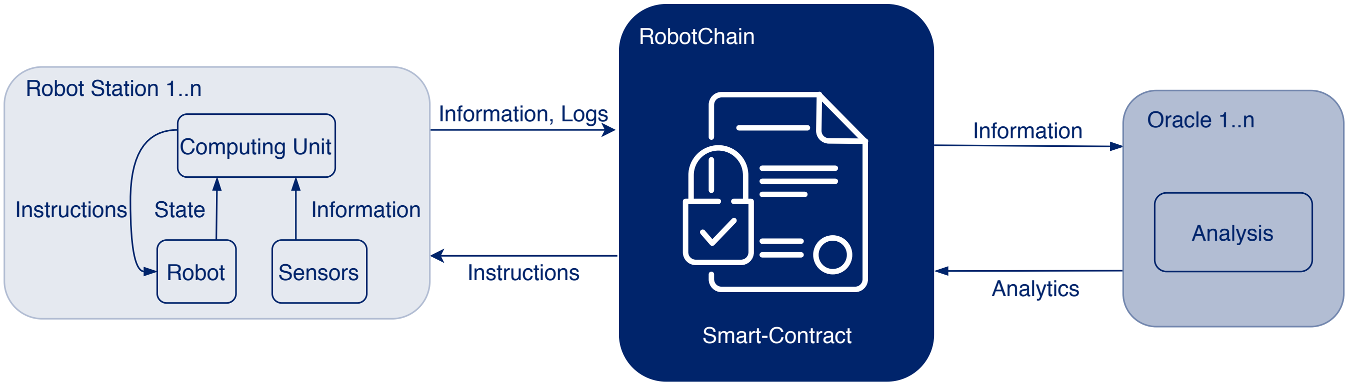

Fig. 1: Architecture of the proposed method.

transaction is signed. For a factory environment where multiple robots from multiple manufacturers are present, these properties are essential, as neither the robots are completely trustworthy, nor the manufacturers trust one another. Here, the blockchain serves the purpose of: 1) storing information in a secure way, 2) providing smart-contracts that are used to receive and send information from the robots and oracles, allowing a single system to perform multiple tasks, requiring no human intervention and that are validated (different smart-contracts can have different behaviours e.g., control robots, provide information, coordinate tasks, among others), and 3) give all the stakeholders a way to trust the network, wherein this, one or more stakeholders might not be trustworthy. The second component are the Robot Stations, where each one may contain sensors, and the robot coupled to a computing unit, which can be any device as long as it allows communication with the robot, the RobotChain and possible external sensors. These Robot Stations can have different settings depending on hardware, environment and tasks, but the core is the same, they feed information about the robot and sensors into the RobotChain and receive instructions from the smart-contracts, which depends on the desired behaviour. The third and final component is the oracles, which are external parties to the blockchain that, if allowed, interact with smart-contracts and use the information contained in them to perform analysis and then feed that analysis into the blockchain. Oracles can be virtually anywhere, such as in cloud systems, and can perform any type of analysis that the data allows them to, such as AI algorithms, and are allowed to interact with one or more smart-contracts if, inside them, this permission is specified.

This is a general approach that can be used with any robot, as long as the robots have external communication to send their logs and receive instructions. The smart-contracts are useful because they contain all the logic that controls the possible changes to be made on the robots and they are not controlled by any party, in the sense that once published into the blockchain, it is impossible to change the code contained on them, and, trigger events without the need for human intervention. The proposed method supports the simultaneous deployment of different smart-contracts.

B. *Example Application*

We demonstrate the proposed method by creating a scenario that simulates a factory environment in which a robotic arm needs to pick a raw material from one conveyor and place it on another one. In the created scenario, the robotic arm is a UR3, the raw material are orange balls. The robot needs to pick the material from the end of a track-line and place it at the beginning. We incorporated a small motor that blocks the balls from reaching the end of the track for n seconds to represent different arrivals at the robot workspace. This is useful to test how the system performs under different raw material arrival intervals. This scenario and the complete sequence of movements in this task can be seen in Figure 2, where the robot picks one ball, places it at the beginning of the track and stops the movement because it is informed to do so since there are no more balls to be picked. The final image represents the home position to which the robot returns if there is nothing more to pick.

For this task, the proposed architecture consisted of RobotChain as a ledger, which contains smart-contracts that execute code to define a robot state. The information contained in the RobotChain is sent by a controlling unit that receives information from the UR3 arm state, which includes position, velocity and effort and an image from a USB camera, placed on a tripod, looking to the place where the balls need to get picked. Both are connected to the same computing unit, but this does not influence the processing or the blockchain, as they are treated separately. As the blockchain can rapidly increase in size [19], we only store on the blockchain, the robotic events (logs) and a tuple comprised of a Hash and an ID that represents the image. The image itself is stored in a database. The information on the blockchain is then used by a trusted external oracle. In our approach, the oracle processes the images to detect how many balls are present. This information is then sent to a smart-contract where it is defined if the robot should stop at the home position, which means that there is no material to transport, or either slow down or speed up, meaning that there are few or many materials to transport, respectively.

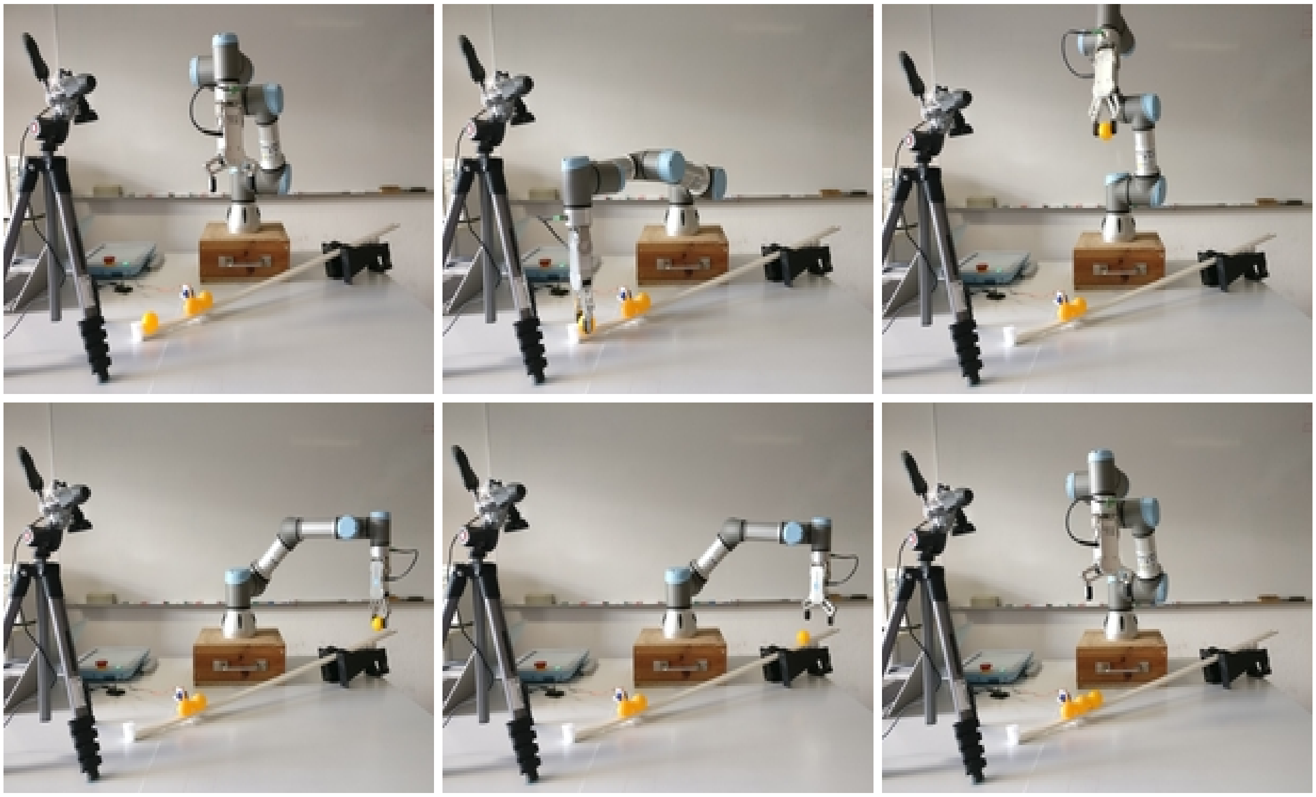

Fig. 2: Pick and place task. UR3 arm picks and places a ping-pong on the track and then goes to the home position because the is nothing more to pick.

C. The Oracle - Image Analysis

As explained before, we use external parties to the blockchain, oracles, to process the images captured. The oracles have read-only access to the database where the images are stored and use the hashes that are in the blockchain to validate that the images were not altered.

In order to detect the raw material in the images captured, we built a detector that detects how many orange balls are present in the image. The algorithm created to perform this task can be divided into 7 distinct steps:

1) Image Pre-processing - to remove noise by using a Gaussian Filter with a kernel size of $3 * 3$;
2) Color Space Transformation - transforms the result of the first step from an RGB color space to an HSV space. This is done to search for the presence of yellow to orange colors and HSV space allow us to do so in a more natural way, whilst being less sensitive to the illumination conditions.
3) Pixel Search - search for pixels that are in the range of yellow to orange, removing all the pixels that fall outside this range;
4) Per-element Bitwise AND - projects the result from the third step into the input image, removing all points that were not in the color range. The result is an RGB image;
5) Color Space Transformation - the result of the last step is transformed from RGB to a gray image;
6) Canny Edge Detection - detect the edges on the gray image, in order to detect the silhouette of the material;
7) Circle Hough Transform - extraction of features from the silhouette in the form of circles.

The detector built for this problem is extremely fast - requiring, on average, 42 milliseconds, with a standard deviation of 2.3 milliseconds (values obtained by testing this method on 10000 images on a AMD Ryzen7 2700 computer).

D. Robot Control

Procedure 1 Smart-Contract Structure in Liquidity

1: Version Number
2: Definition of data structures (type)
3: Definition of the storage structure
4: Init method
5: Entry points

Despite the fact that smart-contracts are usually used to do legal contracts or currency transactions, they can be useful to allow the interaction of oracles with the blockchain, to automatically perform actions and to help nodes of a network obtain global information.

In procedure 1, we present the typical structure of smart-contracts written in Liquidity. First, it is required to define the version of Liquidity used, for example: `[%% version 0.5]`. Then, it is possible to define data structures by using the 'type' syntax. An example of this is: `type oracles = key_hash list`. This step is optional, as one can decide that it's not useful to have data structures defined. The third step is the definition of the storage variable. This step is done similarly to step 2) but it's required that the structure has the name 'storage', for example: `type storage = oracles`. The fourth step is optional, as it serves as an initializer for the storage. This method is defined by the `init` keyword and it is called only once - when the smart-contract is deployed. The way this method initializes the storage is by returning the values to initialize the storage, example: `let%init init_oracles (id : key_hash) = ([id]).` The final requirement in a typical Liquidity structure are the entry points. These methods are used to perform some logic and have actions inside of the smart-contract or on the blockchain. It is possible to have multiple entry points, and when working with oracles, it is advised to create an entry point for each type of task or user that is allowed to call that smart-contract. The signature of an entry point is defined by the `entry` keyword, the parameters received and the name of the storage variable in that context, for example: `let%entry main ((log : string), (oracle : key_hash)) storage = ( [], storage ).` Note that more options can be used, but in the context of integration between Robotics and blockchain, the steps presented are enough to perform rather complex tasks with smart-contracts.

Our approach uses smart-contracts to: 1) register robotic events, 2) to define the logic for robotic control. Specifically to the scenario created, the information of how much material is waiting to be picked up and the robot state, from which it is possible to infer if the robot is transporting anything or not, is used by smart-contracts that define the robot velocity. This velocity is defined in seconds per movement. We defined a simple function on the smart-contract to demonstrate the proposed method. The smart-contract to define the number of balls to be transported is:

$$x = ImageAnalysis() + Transporting() \qquad (1)$$

where $ImageAnalysis()$ is the information that the trusted oracle (trusted in the sense that only it can insert information onto the smart-contract) inserts into the contract and $Transporting()$ is a function inside the contract that adds 1 if the robot is transporting a ball and 0 otherwise.

The velocity is defined by:

$$v = \frac{maxSpeed + meanSpeed}{x} \qquad (2)$$

where $maxSpeed = 2$ and $meanSpeed = 4$. This enforces that the slowest and the fastest velocities correspond to six and two seconds per movement, respectively, when working with a maximum number of balls to be picked equal to three.

V. Discussion

With the architecture proposed we show how it is possible to integrate robotics and blockchain. Even though we showcased a rather simplistic velocity control, it is a novel way of controlling robots that can be adapted to different situations and scenarios and that can also be tuned to ensure that the temperature and other values of the robot don't exceed a threshold in order to avoid failures. This approach is robust to changes due to the fact that no one can alter the control logic without permission, which is imposed by the smart-contract. The blockchain acts as a ledger that securely stores data across a possible untrusted network and, with the capability of interacting with oracles, external process of the data can be performed, which can integrate many forms, but more importantly, it can bring external processing power, such as AI algorithms, to the blockchain. With our proposal, it is possible to have multiple robots working in different tasks and have a unified system to control them which also presents the benefit of having global information of the whole network that can be used upon request. It is also possible to do other kinds of tasks, as it is a modular approach and can be extended to other tasks and even to multiple oracles with ease. There can also be multiple smart-contracts to control different robots and different tasks for the same robot.

The constraints of our approach are: first, how easy it is for the blockchain size to grow, and that is why we store an image representation instead of the image itself on the blockchain. However, this can be further improved by removing unnecessary information in transactions, and applying off-chain mechanisms. Second, the validation speed, which is the greatest bottleneck of most blockchain approaches. If there are thousands of transactions per second, the time to get an answer to change a robot behaviour can slow down. This can be improved by changing the consensus algorithm to one that removes the need for computation in order to validate blocks. In our case, RobotChain is already a high-performant blockchain that allows a high number of transactions per second [18], [20]. More, if the environment where the blockchain is deployed is a private one, such as a factory, the connection speed will further improve the speed. In this setting, to allow the blockchain to scale to more robots, it requires fast block validations, and also a growing number of nodes to aid in the validating the blocks. This means that as the number of robots grow, the computing power associated with the blockchain should also increase.

In table I, we present the velocities of the robot at different times from 4 different experiences, where the velocities are in seconds per movement (speed of the robot) and were retrieved at different times (Time column). The velocity values equal to zero mean that the robot is stopped and waiting for material to pick. The experiments started with the same scenario, 3 balls to be picked in the beginning, which means that the initial velocity of the robot is defined by the aforementioned method to 2 seconds per movement. What differs in the experiences is the time, n, that the motor takes to open and allow one ball to pass, representing the arrival of a new material to be picked by the robot. The experiences conducted were: A: $n = 5s$, B: $n = 15s$, C: $n = 40s$, D: initial $n = 20s$ and incremented by $5s$ every iteration (each time a ball arrives at the picking position).

TABLE I: Values of the velocity, in seconds per movement, for the four conducted experiments.

Time (s)	A	B	C	D	Time (s)	A	B	C	D
10	2	3	3	3	150	3	2	0	0
20	3	3	3	3	175	3	3	6	0
30	3	3	6	3	200	2	3	0	6
50	3	3	6	6	225	2	6	3	0
75	3	3	6	6	250	2	3	6	6
100	3	3	6	6	275	2	6	0	0
125	3	3	6	6	300	2	6	6	0

The values shown are within a 5 minute frame because in all the experiments the values repeat themselves, with exception of experiment D, in which the robot tends to stop more often due to the fact that n keeps on being incremented. The table shows that the proposed method is indeed capable of controlling the robot velocities, which happens in real-time. One property that can be seen in the table, is that due to the fact that the method adjusts the velocities depending on the rhythm of incoming materials, the velocities tend to suffer few changes. For instance, in experience A, even though the velocity stays constant for large periods of time, the method keeps enforcing that the robot should be at a determined velocity, depending on the number of balls to be transported. This is important to keep logs of the robot's and to enable it to adjust to sudden changes.

VI. Conclusions

This paper describes how blockchain can be integrated with robotics by using RobotChain as a decentralized ledger. The proposed architecture is capable of registering robotic events and use oracles for processing acquired data (in our example, images) and controls robots by using smart-contracts. By using such a modular architecture, it is possible to insert new modules that can either serve as oracles and do any type of data processing, such as image or log analysis, or new smart-contracts to act upon the system in a secure way. This system ensures that no one can change past states that were inserted into the blockchain and that the output of the smart-contracts is always free of human changes, meaning that no one can alter a smart-contract logic once it is on the blockchain. Our proposal shows that it is possible to integrate blockchain with robotics and that this integration has advantages beyond having only a way to register and transmit information. Even though we demonstrated the proposed method on a mechanism to control a robot, it can be used in other contexts, such as: distributing tasks to a network of robots; having a mechanism where robots can ask for help in their task if they can't perform it or if they have no information about a specific requirement, for example, identifying humans or objects in images (one robot may not know what are the objects but other robots may); detecting productivity and problems in robots, which can be useful in factories or to monitor other aspects of the stations connected to the blockchain.

References

[1] J. Kober, J. A. Bagnell, and J. Peters, "Reinforcement learning in robotics: A survey," *The International Journal of Robotics Research*, 2013.

[2] T. B. Sheridan, "Human-Robot Interaction: Status and Challenges," *Human Factors*, vol. 58, no. 4, pp. 525–532, 2016, pMID: 27098262.

[3] T. Balch and R. C. Arkin, "Behavior-based formation control for multirobot teams," *IEEE Transactions on Robotics and Automation*, Dec 1998.

[4] D. Nguyen-Tuong and J. Peters, "Model learning for robot control: a survey," *Cognitive processing*, 2011.

[5] C. C. de Wit, B. Siciliano, and G. Bastin, *Theory of robot control*. Springer Science & Business Media, 2012.

[6] G.-Z. Yang *et al.*, "The grand challenges of Science Robotics," *Science Robotics*, 2018.

[7] K. Salah, M. H. U. Rehman, N. Nizamuddin, and A. Al-Fuqaha, "Blockchain for AI: Review and Open Research Challenges," *IEEE Access*, vol. 7, 2019.

[8] T. Marwala and B. Xing, "Blockchain and Artificial Intelligence," *CoRR*, vol. abs/1802.04451, 2018.

[9] V. Lopes and L. A. Alexandre, "An overview of blockchain integration with robotics and artificial intelligence," *Ledger*, vol. 4, 2018.

[10] E. Castelló Ferrer, "The blockchain: A new framework for robotic swarm systems," in *Proceedings of the Future Technologies Conference (FTC)*, K. Arai, R. Bhatia, and S. Kapoor, Eds., 2019.

[11] E. C. Ferrer, O. Rudovic, T. Hardjono, and A. Pentland, "RoboChain: A Secure Data-Sharing Framework for Human-Robot Interaction," *eTELEMED conference*, 2018.

[12] V. Lopes and L. A. Alexandre, "Detecting Robotic Anomalies using RobotChain," in *2019 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC)*, 2019.

[13] V. Lopes, N. Pereira, and L. A. Alexandre, "Robot Workspace Monitoring using a Blockchain-based 3D Vision Approach," in *CVPRW*, 2019.

[14] T. L. Basegio, R. A. Michelin, A. F. Zorzo, and R. H. Bordini, "A Decentralised Approach to Task Allocation Using Blockchain," in *International Workshop on Engineering Multi-Agent Systems, 2017*. Springer, 2017, pp. 75–91.

[15] Teslya, Nikolay and Smirnov, Alexander, "Blockchain-based framework for ontology-oriented robots' coalition formation in cyberphysical systems," *MATEC Web Conf.*, 2018.

[16] V. Strobel and M. Dorigo, "Blockchain technology for robot swarms: A shared knowledge and reputation management system for collective estimation," in *ANTS 2018*, 2018.

[17] V. Strobel, E. Castelló Ferrer, and M. Dorigo, "Managing byzantine robots via blockchain technology in a swarm robotics collective decision making scenario," in *AAMAS*, 2018.

[18] M. Fernandes and L. A. Alexandre, "RobotChain: Using Tezos Technology for Robot Event Management," in *Symposium on Blockchain for Robotic Systems, MIT Media Lab*, 2018.

[19] Y. Nishida, K. Kaneko, S. Sharma, and K. Sakurai, "Suppressing Chain Size of Blockchain-Based Information Sharing for Swarm Robotic Systems," in *CANDARW, 2018*, Nov 2018.

[20] V. Lopes, M. Fernandes, N. Pereira, and L. A. Alexandre, "A time-segmented consortium blockchain for robotic event registration," in *ICBCT 2021*, 2021.