UNIVERSIDADE DA BEIRA INTERIOR
Engenharia

RobotChain: Artificial Intelligence on a Blockchain using Tezos Technology

Vasco Ferrinho Lopes

Dissertação para obtenção do Grau de Mestre em
Engenharia Informática
(2º ciclo de estudos)

Orientador: Prof. Doutor Luís Filipe Barbosa de Almeida Alexandre

Covilhã, Junho de 2019

Acknowledgments

Reaching the end of this dissertation, I would like to express my deepest gratitude for those that made this possible.

First, and foremost, I would like to thank Professor Luís Alexandre for his constant support, guidance, sincerity, patience, understanding and so much more. Thank you for teaching me so much and for passing on to me important values, such as honesty and hard work. Your motivation, expertise, availability, insights, and help made the conclusion of this dissertation possible. I am beholden for the opportunity of working with such an amazing person and professional that is always available to help his students and to guide them. My deepest thank you for trusting me and giving me the opportunity to work with you.

To my parents, brother and grandparents, I express my utmost gratitude. Their support was unconditional, not only to make this chapter of my life possible but for every moment that they put up with me, for the knowledge they impart on me and for always being present when I need.

To my beloved Diana, I am deeply grateful for all your love, support, help and above all, your understanding for the time I was absent and for the thousands of times I had to cancel our plans due to this work. I truly appreciate you, what you represent and your entire support. I am, beyond question, happy and grateful for having you in my life.

No less important, I thank all my friends that helped me through not only the dissertation but my academic path. Especially everyone in Socialab that actively participated in discussions and helped me when I needed. Namely, Nuno Pereira, João Neves, António Gaspar, Abel Zacarias, Bruno Silva, Miguel Fernandes, and many others. I would also like to thank the guys from Segal and C4G: José Manteigueiro, Fábio Pereira, André Rodrigues, Gonçalo Henriques, João Antunes and Luís Carvalho for bearing with me when I was tired of working. To Acácio Correia and José Ribeiro for helping me grow both as a person and as a professional in the uPato project and ever since. To “O Colestrol Não Pode Vencer” guys that always made sure I spent some time with friends, and to everyone that was present throughout these 5 years and helped me with this dissertation, namely André Rodrigues, José Manteigueiro, Mário Pereira, Nelson Fonseca, João Amaral, Eliana Pereira and many others that are not mentioned here, a huge thank you.

Last but not least, I would like to thank Tezos for the financial support that was given to RobotChain project, which made this project possible.

Resumo

Hoje em dia é possível ver que a blockchain não está apenas a crescer a um ritmo exponencial, mas que é também uma tecnologia disruptiva que mudou a forma como trabalhamos com transações financeiras. Ao fornecer uma maneira eficiente de confiar numa rede desconhecida e de permitir realizar transações sem a necessidade de uma autoridade central, a blockchain cresceu rapidamente. Além disso, a blockchain fornece também descentralização de dados, imutabilidade, acessibilidade, não-repúdio e irreversibilidade, o que torna esta tecnologia indispensável em muitos setores. Mas, mesmo fornecendo propriedades interessantes, a blockchain não tem sido amplamente utilizada fora do âmbito financeiro. Da mesma forma, os robôs têm sido cada vez mais utilizados em fábricas para automatizar tarefas que vão desde pegar objetos, transportá-los e colaborar com humanos para realizar tarefas complexas. Porém, é importante impor que os robôs atuem entre certos limites legais e morais e que seus eventos e dados são armazenados com segurança e que estes possam ser auditáveis. O problema é que isso raramente acontece. Os robôs são programados para executar uma tarefa específica sem se ter total certeza de que essa tarefa irá ser executada sempre de maneira correta, e os seus dados são armazenados localmente, desconsiderando a segurança dos dados. Sendo que em muitas ocasiões, não existe qualquer segurança. Isso significa que os dados, especialmente os *logs*, podem ser alterados, o que pode resultar em que os robôs e, pela mesma linha de pensamento, os fabricantes, possam ser acusados de problemas que não causaram. Tendo isto em consideração, neste trabalho, procuramos integrar a blockchain com a robótica, com o objetivo de proporcionar maior segurança aos robôs e aos dados que geram e potenciar ainda a utilização de algoritmos de inteligência artificial. Fazendo uma visão abrangente dos métodos que propõem integrar a blockchain e inteligência artificial ou robótica, descobrimos que este é um campo em crescimento, mas que há uma falta de propostas que tentem melhorar os sistemas robóticos utilizando a blockchain. Ficou também claro que a maioria das propostas existentes que integram inteligência artificial e blockchain estão focadas na construção de *marketplaces* e só utilizam a blockchain para armazenar a informação sobre as transações que foram executadas. Assim, neste documento, propomos três métodos que utilizam a blockchain para resolver diferentes problemas associados a robôs. O primeiro é um método para armazenar, com segurança, *logs* de robôs dentro de uma blockchain, utilizando para isso *smart-contracts* como armazenamento. Neste método foi também proposta uma maneira de detetar anomalias em robôs automaticamente, utilizando para isso os dados contidos na blockchain e *smart-contracts* para definir a lógica do algoritmo. Ao utilizar *smart-contracts*, é garantido que os dados são seguros e imutáveis, desde que a blockchain contenha nós suficientes a participar no algoritmo de consenso. O segundo método vai além de registar eventos, para registar também informações sobre sensores externos, como uma câmara, e utilizando *smart-contracts* para permitir que Óraculos interajam com a blockchain, foi possível utilizar algoritmos de análise de imagens, que podem detetar a presença de material para ser recolhido. Esta informação é então inserida num *smart-contract* que define automaticamente o movimento que um robô deve ter, tendo em consideração a quantidade de material à espera para ser recolhida. A terceira proposta é um método que utiliza a blockchain para armazenar informações sobre robôs, e imagens provenientes de uma Kinect. Esta informação é então utilizada por Óraculos que verificam se existe alguma pessoa dentro do um espaço de trabalho de um robô. Se existir alguém, essa informação é armazenada e diferentes Óraculos tentam identificar a pessoa. No fim, um *smart-contract* age apropriadamente, mudando ou até mesmo parando o robô, dependendo da identidade da

pessoa e se a pessoa está localizada dentro da zona crítica ou de aviso.

Com este trabalho, mostramos como a blockchain pode ser utilizada em ambientes onde existam robôs e como esta pode ser benéfica em contextos onde a cooperação entre várias entidades, a segurança e a descentralização dos dados são essenciais. Mostramos também como Óraculos podem interagir com a blockchain e cooperar de forma distribuída, para alavancar algoritmos de inteligência artificial de forma a realizar análises nos dados, o que nos permite detetar anomalias robóticas, material para ser recolhido e a presença de pessoas em imagens. Mostramos também que os *smart-contracts* podem ser utilizados para executar mais tarefas do que servir o propósito de fazer transações monetárias de forma automática. As arquiteturas propostas neste trabalho são modulares e podem ser utilizadas em vários contextos, como no fabrico de peças, controle de robô e outras. Devido ao facto de que as arquiteturas propostas, são fáceis de integrar, adaptar, manter e estender a novos domínios. A nossa opinião é que a interseção entre a blockchain e a robótica irá moldar parte do futuro da robótica moderna assim que a blockchain seja mais utilizada e fácil de integrar em sistemas robóticos. Esta integração será muito proeminente em tarefas onde os robôs precisam de se comportar sob certas restrições, em enxames de robôs, devido ao fato de que a blockchain fornece informação global sobre o estado da rede, e também em fábricas, porque as ações realizadas por um robô podem ser facilmente estendidas ao resto dos robôs, e porque fornece um mecanismo extra de segurança aos dados e a todas as ações que são efetuadas com ajuda de *smart-contracts*.

Palavras-chave

Análise de Imagens, Blockchain, Controlo de robôs, Descentralização, Deteção de Anomalias, Deteção de Caras, Deteção de Pessoas, Imagens 3D, Inteligência Artificial, Monitorização de Espaços de Trabalho, Óraculos, Registo de Eventos, RobotChain, Smart-Contracts, Tezos

Resumo Alargado

Hoje em dia é possível ver que a blockchain não está apenas a crescer a um ritmo exponencial, mas que é também uma tecnologia disruptiva que mudou a forma como trabalhamos com transações financeiras. Ao fornecer uma maneira eficiente de confiar numa rede desconhecida e de permitir realizar transações sem a necessidade de uma autoridade central, a blockchain cresceu rapidamente. Além disso, a blockchain fornece também descentralização de dados, imutabilidade, acessibilidade, não repúdio e irreversibilidade, o que torna esta tecnologia indispensável em muitos setores. Mas, mesmo fornecendo propriedades interessantes, a blockchain não tem sido amplamente utilizada fora do âmbito financeiro. Da mesma forma, os robôs têm sido cada vez mais utilizados em fábricas para automatizar tarefas que vão desde pegar objetos, transportá-los e colaborar com humanos com humanos para realizar tarefas complexas. Porém, é importante impor que os robôs atuem entre certos limites legais e morais e que os seus eventos e dados são armazenados com segurança e que estes possam ser auditáveis. O problema é que isso raramente acontece. Os robôs são programados para executar uma tarefa específica sem se ter total certeza de que essa tarefa irá ser executada sempre de maneira correta, e os seus dados são armazenados localmente, desconsiderando a segurança dos dados. Sendo que em muitas ocasiões, não existe qualquer segurança. Isso significa que os dados, especialmente os *logs*, podem ser alterados, o que pode resultar em que os robôs e, pela mesma linha de pensamento, os fabricantes, possam ser acusados de problemas que não causaram.

Esta dissertação tem como principal objetivo estender a aplicabilidade da blockchain a ambientes robóticos. De forma mais específica, pretende-se: 1) perceber como se pode utilizar a blockchain fora de ambientes financeiros, com especial foco em integrar a blockchain com sistemas robóticos e inteligência artificial, incluindo uma revisão detalhada do estado-da-arte; 2) desenvolver um método que permita algoritmos de inteligência artificial terem acesso às informações registadas na blockchain; 3) utilizar os dados que estão contidos na blockchain num algoritmo de inteligência artificial de forma a melhorar este; e por último, 4), permitir que algoritmos de inteligência artificial e robôs registem os seus eventos numa blockchain. De forma a alcançar estes objetivos, esta dissertação apresenta várias contribuições descritas ao longo de oito capítulos.

O primeiro capítulo define o âmbito e o problema onde esta dissertação se enquadra. Para além disso, são também descritos os principais objetivos do presente trabalho de investigação, assim como as principais contribuições da investigação feita neste trabalho de forma a melhorar o estado-da-arte tanto na integração da blockchain com inteligência artificial como com robôs. No fim do capítulo, é definida a estrutura deste documento.

O segundo capítulo apresenta uma revisão da literatura nas áreas de investigação necessárias ao desenvolvimento de um sistema que integra robótica e blockchain. Este divide-se em secções onde é: 1) introduzida a blockchain em detalhe, explicando os seus componentes e como a informação é registada; 2) explicado em detalhe as propostas que integram algoritmos de inteligência artificial com blockchains e com sistemas robóticos; e 3) métodos de deteção de anomalias robóticas. No que diz respeito à análise de métodos que integram algoritmos de inteligência artificial com blockchain, foi necessário verificar métodos existentes que não se

baseiam apenas em contexto académico, pois, uma parte significativa das propostas existentes, foca-se em desenvolver estes métodos industrialmente. Com esta revisão, foi possível concluir que existe uma grande falha neste campo de investigação. Esta falha é maioritariamente existente porque quase todos os métodos propostos fazem uma integração destas duas tecnologias apenas para utilizar a blockchain como registo de transações financeiras nas aplicações que são desenvolvidas fora da blockchain. Sendo que numa grande parte, as aplicações desenvolvidas consistem em mercados para venda de algoritmos. De seguida, foi feito um estudo de todos os métodos existentes até à data que tentam integrar sistemas robóticos com blockchain. Daqui pode-se concluir que, apesar da quantidade de métodos propostos não ser tão grande quando comparada com outros campos de investigação, esta área está em crescimento. O estado-da-arte atual foca-se particularmente em utilizar a blockchain para fornecer informação global sobre uma rede de robôs e para mitigar problemas associados ao aparecimento de *byzantine* robôs nesses sistemas. Finalmente, foi também efetuado uma leitura e explicação detalhada dos métodos na literatura que se propõem a detetar anomalias em robôs. Desta leitura, foi possível ver que a maioria destes métodos foca-se na utilização de algoritmos de inteligência artificial, tais como redes neuronais convolucionais e sistemas imunológicos artificiais, para resolver o problema de detetar anomalias. Destes foi possível concluir duas coisas: a primeira é que os métodos propostos, ao utilizarem este tipo de algoritmos, dificilmente podem ser estendidos para novos robôs e novos ambientes, sendo que se for possível fazer-se essa mudança, os métodos teriam de ser retreinados, o que tem um custo temporal elevado e envolve também a aquisição de novos dados e o tratamento destes. A segunda conclusão é que os métodos não podem ser facilmente comparados, pois são criados para resolver problemas específicos e não existe uma base de dados pública com anomalias robóticas, que sirva como maneira de comparar os métodos. Após esta revisão detalhada do estado-da-arte, foi possível perceber que existem falhas fundamentais nestas áreas de investigação, sendo que focámos o nosso trabalho em melhorar o estado-da-arte referente à integração de sistemas robóticos com blockchains, possibilitando a integração com algoritmos de inteligência artificial.

O terceiro capítulo descreve a blockchain utilizada neste trabalho - RobotChain. Para além de a descrever em detalhe, descreve também em detalhe todos os componentes que fazem parte da blockchain da Tezos, tais como o algoritmo de consenso e como é possível fazer alterações no núcleo desta blockchain sem ser necessário criar uma blockchain nova que esteja em concordância com as novas regras. Esta descrição detalhada é feita porque a RobotChain é baseada nesta blockchain pública e contém as suas características principais. Neste capítulo foi também descrito o processo de criação da RobotChain e as alterações que foram feitas, tais como adicionar novos campos às transações e a criação de mecanismos que de n em n transações repartem a blockchain de forma a reduzir o seu tamanho e por consequente, o custo, em termos de tempo, para validar transações. Neste capítulo é também explicado como funcionam os *smart-contracts*, e é apresentado como estes podem ser desenvolvidos em Liquidity para funcionarem tanto em redes da Tezos como na RobotChain. Por fim, é apresentado um exemplo de um *smart-contract* em Liquidity e em Michelson que serve o propósito de armazenar registos de robôs dentro da blockchain de forma imutável. Este *smart-contract*, faz parte do método proposto no capítulo 5 e foi utilizado ao longo do projeto para armazenar informação na blockchain.

O quarto capítulo serve o propósito de introduzir os robôs colaborativos, focando-se especialmente no UR3, e também explicar as opções tomadas neste projeto de forma a se conseguir

trabalhar com um robô real. Neste capítulo, começou-se por explicar o que são os robôs colaborativos, introduzindo a sua necessidade em fábricas para automatização de processos e para haver colaboração entre humanos e máquinas de forma a haver um aumento de produtividade. Logo a seguir explicou-se como se pode configurar um robô UR3 de forma a permitir interação com o *Robot Operating System* e com componentes externos ao robô. Depois, explicou-se o método criado para haver comunicação entre o UR 3 e um computador externo, sendo que esta comunicação foi feita por TCP/IP na forma de *sockets*, permitindo que sejam enviados comandos para o robô a partir de um computador, ao invés de se utilizar o dispositivo acoplado ao robô e, ao mesmo tempo receber informação em tempo real sobre o estado dos motores do robô. Por fim, explicou-se as tomadas de decisão relativas a estas escolhas para comunicação com o UR 3. Em suma, este capítulo explica o robô UR3 e as decisões feitas em termos de comunicação deste com componentes externos, sendo que estas decisões definiram o percurso deste projeto no que tem a ver com a utilização do robô.

Os três capítulos que se seguem introduzem os três métodos propostos neste documento, focando-se na resolução de problemas encontrados ao estudar o estado atual da tecnologia no âmbito da robótica e da sua integração com sistemas como a blockchain. Sendo que o quinto capítulo apresenta um método para deteção de anomalias em robôs e, ao mesmo tempo, um mecanismo que permite o registo de eventos e *logs* de robôs na RobotChain de forma segura e imutável. Este método foi proposto de forma a colmatar o problema do armazenamento de dados provenientes de robôs em ambientes sensíveis, como fábricas, e, ao mesmo tempo utilizar esses dados para demonstrar que é possível integrar a blockchain com a robótica de forma a resolver problemas de forma eficiente. Para se resolver esses problemas, propusemos uma arquitetura que utiliza a RobotChain como um mecanismo de armazenamento descentralizado e *smart-contracts* que não só armazenam a informação proveniente de robôs, mas também automaticamente conseguem detetar anomalias. Com esta arquitetura foi possível reforçar a ideia que apenas robôs que tenham permissão para tal, possam inserir dados na blockchain e, ao mesmo tempo que esses dados são guardados de forma imutável, auditável, transparente e não repudiável. Este método foi testado com um robô UR3 real, onde foi possível detetar automaticamente anomalias que foram forçadas no robô, mostrando que a integração entre robôs e blockchain fornece mecanismos de segurança que muitas vezes são difíceis de ter em sistemas tradicionais, e mais importante, que esta integração é possível. Baseado no método proposto no capítulo quatro, o capítulo cinco apresenta uma segunda proposta que tem como intenção aprofundar mais a integração entre a blockchain e a robótica e, ao mesmo tempo permitir que mais intervenientes possam interagir com o sistema, focando-se em propor um método inovador para o problema de controlar robôs. Isto porque detetamos que a maioria dos ambientes que envolvem robôs colaborativos requerem que exista sempre pelo menos uma pessoa a controlar o robô ou, no melhor dos casos, a supervisionar os meus movimentos. Para resolver esse problema, este capítulo apresenta uma arquitetura que utiliza o método apresentado anteriormente, mas também componentes que permitem um controlo automático de robôs e processamento de imagens provenientes de sensores externos. Este método utiliza *smart-contracts* para efetuar o controlo dos robôs, o que reforça que a movimentação do robô está bem definida e é sempre enviada para o robô da mesma forma. Os Óraculos entram aqui para processar informação que seja proveniente dos robôs ou de sensores externos, de forma a fornecer mais informação aos *smart-contracts* encarregues de definir a lógica de controlo. Para ilustrar este método, criámos um cenário onde um braço robótico UR3 executa uma tarefa de pegar em objetos e deixá-los noutra posição. Para isto, colocámos uma câmara a olhar para a linha de transporte do material e a

enviar as imagens para um *smart-contract*. Estas imagens são então processadas por Óraculos que tenham permissão para tal, que detetam a quantidade de material que existe para ser apanhado e inserem essa informação num *smart-contract*. Com esta informação, é então definida automaticamente a velocidade do robô - reduzindo-a se existirem poucos materiais para serem recolhidos, aumentando-a se existirem muitos ou parando o robô se não existir nenhum material para ser recolhido, até que exista. Esta proposta tem a vantagem de não requerer nenhuma intervenção humana, que todas as decisões e a informação pode ser facilmente verificada e que a lógica de um robô pode ser facilmente estendida a outros robôs devido à utilização de *smart-contracts*.

O capítulo sete apresenta o terceiro e último método proposto neste documento. Este método baseia-se nos dois anteriores de forma a propor uma forma inovadora de resolver o problema em que por vezes, há a necessidade de um robô numa fábrica estar isolado para se garantir que não se danifica nem magoa terceiros. O método proposto utiliza a blockchain para registar eventos de robôs e informação sobre o espaço de trabalho do dado robô. Esta informação é extraída a partir de uma Kinect no formato de imagens 3D. Estas imagens depois de armazenadas na blockchain a partir de *smart-contracts*, são processadas por Óraculos para detetar a presença, ou não, de pessoas dentro do espaço de trabalho do robô. Mais especificamente, dentro da zona de aviso ou da zona crítica que são criadas à volta do robô. Se existir alguém dentro de uma dessas zonas, essa informação é registada e um Óraculo diferente conduz o processo de tentar identificar quem a pessoa, ou pessoas são. Se a pessoa for conhecida e estiver dentro da zona de aviso, nada acontece com a exceção de haver um registo na blockchain desta informação. Caso a pessoa entre dentro da zona crítica, a informação não é só registada, como a velocidade do robô é automaticamente reduzida para 10 segundos por movimento, sendo isto feito autonomamente por *smart-contracts*. Por outro lado, se a pessoa for desconhecida e estiver presente dentro da zona de aviso, o robô tem a sua velocidade reduzida para 10 segundos por movimento. Caso a pessoa entre na zona crítica, o robô é parado até novas ordens. Este método serve para demonstrar que é possível fazer a integração de blockchain com robótica e, ao mesmo tempo ter múltiplas entidades a processar os dados inseridos na blockchain. Mais ainda, demonstra também que é possível utilizar algoritmos computacionalmente exigentes, como os utilizados no processamento de imagens.

Por último, os principais resultados deste trabalho de investigação são resumidos no capítulo oito. Para além disso, discutem-se também as principais contribuições para o desenvolvimento da integração ubíqua entre a blockchain e sistemas robóticos. Finalmente, faz-se um resumo dos métodos propostos, explicando as suas vantagens e demonstrando que as propostas são benéficas, pois providenciam arquiteturas de sistemas que são auditáveis e que permitem não só guardar informação de forma segura e imutável, mas também a interação de múltiplas entidades para processamento de dados. No fim deste capítulo faz-se também uma perspetiva de trabalho futuro, que permita uma integração entre a blockchain e sistemas robóticos que forneçam mecanismos úteis a qualquer ambiente de trabalho, tanto em fábricas como em casas domésticas.

Abstract

Blockchain technology is not only growing everyday at a fast-passed rhythm, but it is also a disruptive technology that has changed how we look at financial transactions. By providing a way to trust an unknown network and by allowing us to conduct transactions without the need for a central authority, blockchain has grown exponentially. Moreover, blockchain also provides decentralization of the data, immutability, accessibility, non-repudiation and irreversibility properties that makes this technology a must in many industries. But, even thought blockchain provides interesting properties, it has not been extensively used outside the financial scope. Similarly, robots have been increasingly used in factories to automate tasks that range from picking objects, to transporting them and also to work collaboratively with humans to perform complex tasks. It is important to enforce that robots act between legal and moral boundaries and that their events and data are securely stored and auditable. This rarely happens, as robots are programmed to do a specific task without certainty that that task will always be performed correctly and their data is either locally stored, without security measures, or disregarded. This means that the data, especially logs, can be altered, which means that robots and manufacturers can be accused of problems that they did not cause. Henceforth, in this work, we sought to integrate blockchain with robotics with the goal to provide enhanced security to robots, to the data and to leverage artificial intelligence algorithms. By doing an extensive overview of the methods that integrate blockchain and artificial intelligence or robotics, we found that this is a growing field but there is a lack of proposals that try to improve robotic systems by using blockchain. It was also clear that most of the existing proposals that integrate artificial intelligence and blockchain, are focused on building marketplaces and only use the latter to storage transactions. So, in this document, we proposed three different methods that use blockchain to solve different problems associated with robots. The first one is a method to securely store robot logs in a blockchain by using smart-contracts as storage and automatically detect when anomalies occur in a robot by using the data contained in the blockchain and a smart-contract. By using smart-contracts, it is assured that the data is secure and immutable as long as the blockchain has enough peers to participate in the consensus process. The second method goes beyond registering events to also register information about external sensors, like a camera, and by using smart-contracts to allow Oracles to interact with the blockchain, it was possible to leverage image analysis algorithms that can detect the presence of material to be picked. This information is then inserted into a smart-contract that automatically defines the movement that a robot should have, regarding the number of materials present to be picked. The third proposal is a method that uses blockchain to store information about the robots and the images derived from a Kinect. This information is then used by Oracles that check if there is any person located inside a robot workspace. If there is any, this information is stored and different Oracles try to identify the person. Then, a smart-contract acts appropriately by changing or even stopping the robot depending on the identity of the person and if the person is located inside the warning or the critical zone surrounding the robot.

With this work, we show how blockchain can be used in robotic environments and how it can beneficial in contexts where multi-party cooperation, security, and decentralization of the data is essential. We also show how Oracles can interact with the blockchain and distributively cooperate to leverage artificial intelligence algorithms to perform analysis in the data that allow us to detect robotic anomalies, material in images and the presence of people. We also

show that smart-contracts can be used to perform more tasks than just serve the purpose of automatically do monetary transactions. The proposed architectures are modular and can be used in multiple contexts such as in manufacturing, network control, robot control, and others since they are easy to integrate, adapt, maintain and extend to new domains. We expect that the intersection of blockchain and robotics will shape part of the future of robotics once blockchain is more widely used and easy to integrate. This integration will be very prominent in tasks where robots need to behave under certain constraints, in swarm robotics due to the fact that blockchain offers global information and in factories because the actions undertaken by a robot can easily be extended to the rest of the robots by using smart-contracts.

Keywords

3D Images, Anomaly Detection, Artificial Intelligence, Blockchain, Decentralization, Event Registration, Face Detection, Image Analysis, Oracles, People Detection, Robot Control, RobotChain, Smart-Contracts, Tezos, Workspace Monitoring.

Contents

List of Figures

List of Tables

Acronyms

AGI Artificial General Intelligence

AIS Artificial Immune System

AI Artificial Intelligence

API Application Programming Interface

CNN Convolutional Neural Network

Cobot Collaborative Robot

DCGAN Deep Convolutional Generative Adversarial Network

DPoS Delegated Proof-of-Stake

DT Decision Tree

EVM Ethereum Virtual Machine

GAN Generative Adversarial Network

GPU Graphics Processing Unit

IoT Internet of Things

MLP Multilayer perceptron

ML Machine Learning

MSE Mean Squared Error

NN Neural Network

PaaS Platform As a Service

PoB Proof-of-Burn

PoS Proof-of-Stake

PoW Proof-of-Work

RF Random Forest

ROS Robot Operating System

RPC Remote Procedure Call

SDAE Stacked Denoising Autoenconder

SOM Self-Organizing Map

SVM Support Vector Machine

k-NN K-Nearest Neighbor

Chapter 1

Introduction

Blockchain technology has been used to allow monetary transactions to be placed digitally without the need for a central authority. News about how cryptocurrencies are growing in use and value are extremely common. The same happens with new approaches of blockchain to do something more with the cryptocurrencies, either increase its privacy, the inner components of the blockchain or to store more transactions in a cheaper way. This happens because blockchain is a revolutionary technology that has several properties among which, immutability, accessibility, non-repudiation, decentralization of the data and the existence of smart-contracts stand out. Blockchain introduced a way to make transactions over a network of peers that might be untrustworthy and without a central authority that enforces the centralization of all the data. This is done because of the consensus algorithm, which makes every peer work to validate the transactions, and in return, they receive a small reward, usually in the form of cryptocurrencies. However, blockchain utility outside of the financial scope and to serve as a ledger has not been deeply studied. There are some proposals that try to use blockchain to improve systems, such as transportation [5, 6] and healthcare [7, 8, 9], but almost all rely on the blockchain only to store information about transactions.

Similarly, robots are being used more than ever, especially in factories. The advent of affordable and high-performance robots, with incredible power, makes them the perfect fit to automate repetitive tasks, such as assembly lines in factories, and to collaboratively work with Humans. This is due to the increase in research on Artificial Intelligence (AI) with focus on robotics. However, robots either need to be supervised or placed inside a cage to make sure that they don't damage other objects or people. A specific type of robots, which are called Collaborative Robots (Cobots), try to mitigate some of those problems by having sensors to stop upon collisions or methods to collaboratively work with humans. However, these solutions come short when we think in robots that autonomously interact with people in factories and must abide to legal and moral principles. As a result, we are still performing mundane tasks every day. Most of the time, these are boring and unproductive tasks that could be automated, but as there are no perfectly secure robot systems, we are still far from it. Moreover, if we think in sensitive environments like factories, data security is crucial. However, most robots don't have systems to securely store their information, such as logs and events, and the factories usually ignore this because implementing such systems is time-consuming and not an easy task. This makes the robots susceptible to be tampered with, making the information they hold in case of accidents useless. We find that blockchain could be very useful not only in this case, since it provides immutability of the data and decentralization, which solves this issue, but in many robotic contexts. This technology could be very beneficial in contexts where multi-party cooperation, security, and decentralization of the data is essential. Properties such as immutability, accessibility and non-repudiation and the existence of smart-contracts make blockchain technology very interesting in robotic contexts that require event registration, global information of the network, or integration with Artificial Intelligence.

In this work, we seek to understand how blockchain can be used outside the financial scope, by doing an extensive overview of the proposed methods available, and how can we use it in robotic contexts, possibly with AI algorithms levering the data that is inserted into the blockchain.

1.1 Motivation and Objectives

In this work, we are particularly interested in integrating robotics with blockchain and AI. The main objective of this dissertation is to perceive how this integration is possible, how is it possible to give AI algorithms access to the information stored in the blockchain and use that data to conduct analytics and allow external parties to insert those analytics into the blockchain. These goals aim to supply AI algorithms with data, which turns into valuable information that can be used to make the blockchain induce changes in the robots. The motivation behind these goals is to show that such integration is possible and useful both to enhance robots' capabilities as well as AI algorithms. Blockchain is normally used to store information about monetary transactions, but it has very powerful characteristics that we find important for robotic systems. The most important ones are: replication, decentralization of the data, irreversibility, accessibility, non-repudiation, time-stamping of transactions and (pseudo) anonymity. Therefore, blockchain can ensure that robot logs and events are stored in the blockchain and that no one can alter them. In this project, we are also interested in using smart-contracts to solve the aforementioned goals, since smart-contracts can have action-triggers autonomously. For those purposes, several methods that try to use blockchains outside the financial scope will be analyzed.

1.2 Main Contributions

The overall contribution of our work is a set of novel methods to control and monitor robots and monitor robot workspaces. The results attained by our approach can also be considered an achievement, since they not only are novel, but they also improve the state-of-the-art in a field that is still its infancy. This work shows that the integration of blockchain and robots is possible and can be used to improve the current systems. The following paragraphs briefly describe the individual contributions of this dissertation to advance the state-of-the-art in blockchain, robotics and AI.

The first contribution is a comprehensive review of the state-of-the-art both regarding the integration of blockchain and robotics as well the integration of blockchain and AI. This work resulted in a survey published as a pre-print in Arxiv [10] and later, a shorter version with focus on the intersection of robotics and blockchain was published in the Ledger Journal - Proceedings of the First Symposium on Blockchain and Robotics, MIT Media Lab, 5 December 2018 [11].

The second contribution regards the proposal of the robotic anomaly detection using block-chain as a ledger, resulting in a paper published in the 2019 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC) [12].

The third contribution is a novel approach to the well-studied problem of controlling robots. Our solution uses smart-contracts to automatically change the behaviour of robots depending on environment information that is processed by Oracles. This work was submitted to 2019

IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) and is still in the reviewing process. Meanwhile, a pre-print was published in Arxiv [13].

The fourth contribution regards the first robot workspace monitoring approach that uses blockchain to store and distribute information and smart-contracts to have actions on the robots depending if a person is detected inside the robot workspace by an Oracle. This work was accepted for publication in the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW) [14] under the workshop "Blockchain Meets Computer Vision and AI".

The fifth and final contribution was the dissemination of this work in the European Robotics Forum 2019. We were invited by two workshops to present what blockchain is and how we were leveraging it to improve robotics field. The first workshop was "Cloud robotics: Deployment prospects and future needs" where we explained blockchain and talked about how Cloud can be used as Oracles in blockchain systems and showcased our work. The second workshop was "Blockchain in Robotic Applications" where we focused the presentation on explaining our proposals.

1.3 Thesis Organization

The remainder of this thesis is organized as follow: chapter 2 gives a short introduction about blockchain as well as the most relevant works that advanced blockchain technology. Moreover, we present a detailed overview of the proposals that integrate AI algorithms and robotics with blockchain and methods that focus on detecting anomalies, with the focal point on robots. Chapter 3 describes RobotChain in detail as it is the blockchain used in this work. Tezos' blockchain is also explained in this chapter as it served as the basis for RobotChain, and we also explain how smart-contracts are created and used in RobotChain. Chapter 4 describes the UR3 robotic arm and explains the process of configuring it and creating the system used throughout this work to acquire data from the robot and to manipulate it in different tasks. Chapter 5 describes the method proposed to store robotic events inside the blockchain and use that data to detect robotic anomalies. Chapter 6 proposes and describes a method that uses smart-contracts both as a communication gate between the blockchain and the Oracles and as action-triggers to control robots autonomously. Chapter 7 explains and proposes the last method of this dissertation. This method uses blockchain as a decentralized ledger, Oracles to process 3D images and smart-contracts to control a robot depending on the information that is inserted into the blockchain. Finally, conclusions are drawn in chapter 8 and the future work is outlined.

Chapter 2

Fundamental Concepts and Related Work

In this chapter, we review the concepts related to blockchain technology, how blockchain has been integrated with other technologies, such as AI, as well as the most prominent work regarding blockchain integration with robotics and AI and proposals that try to detect anomalies in robotic systems. Section 2.1 introduces blockchain technology, and presents the advances that this technology had since its first idealization. Section 2.2 reviews the methods that integrate blockchain with AI and presents a short comparison of the studied methods. Similarly, section 2.3 introduces the methods that integrate blockchain with robotics systems and does a comparative analysis. Section 2.4 introduces the state-of-the-art regarding anomaly detection, namely the methods and architectures of the proposals that try to detect robotic anomalies. Finally, section 2.5 summarizes the most relevant conclusions of this chapter.

2.1 Blockchain

Blockchain is a recent technology that has been disruptive in many industrial and academic fields. The core idea behind blockchain was first introduced by Leslie Lamport [15] where he described a model to reach consensus over a network of computers in which the computers may be unreliable. This work served as a base for the technology proposed by Satoshi Nakamoto in 2008 [16], called Bitcoin. Bitcoin introduced the concept of cryptocurrencies transactions in a decentralized network with rewards to the peers that validate the transactions. But, the biggest and most exciting thing about Bitcoin isn't Bitcoin at all, it's the blockchain protocol. The blockchain protocol is, in its essence, a digital distributed and decentralized ledger that is composed of digital transactions and shared through a network. This protocol is based on a Peer-to-Peer architecture, with every participant forming a node in the network. The transactions performed in this network are stored in blocks that are coupled to the blockchain. These blocks hold information, most likely, information about lists of monetary transactions if a cryptocurrency is being used in that blockchain. The information contained within the blocks make up a database where adding or changing the information in that database comes in the form of appending new blocks to the blockchain. This property and the consensus algorithm inherent to the blockchain make up for the security that comes with blockchains.

The most usual proprieties that a block has, based on the blockchain overview conducted by Zibin Zheng et al. [17], are:

- Block Version: parameter that indicates the rules to be followed in order to validate the block;
- Merkle tree root hash: the hash value that represents all the transactions in the block;
- Timestamp: the time, in seconds, that represents when a peer validated the block;
- NBits: the target for a hash (not used in every consensus methods);

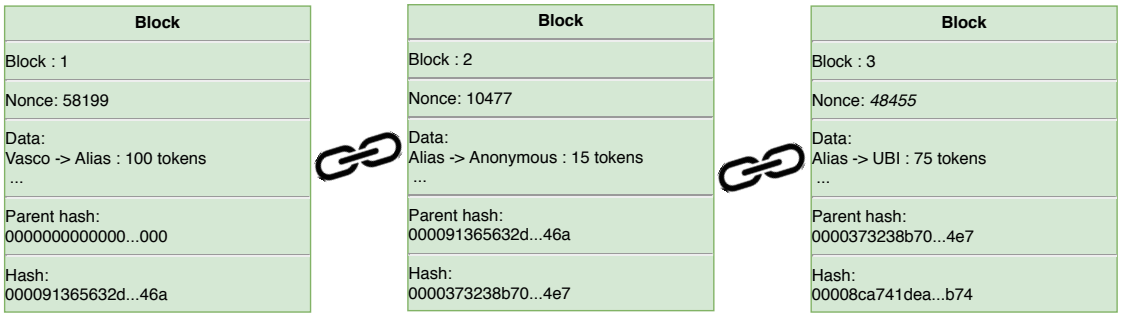


Figure 2.1: Blockchain stored in a peer. This blockchain is correct and complies to the rules.

- Nonce: a random number that the peer increments to achieve a hash for the block that complies with the target;
- Parent block hash: a hash value that points to the predecessor block.

The way blocks are usually constructed (different blockchains may have different proprieties) enforces that a block is more secure as more blocks come after it. This security occurs because of the parent block hash. Cryptographic hash functions work by receiving an input and giving as output a "fingerprint" of the input. These functions ensure that the same input has always the same output and that different inputs have a very low probability of having the same hash value [18]. When a block is valid and a hash value is inserted into it, a change will lead the block to become invalid, as the values do not represent the hash that it has and all the blocks that are after the said block become invalid (wrong parent block hash). Malicious users could try to cheat the network by changing hashes, but the since blockchain is a distributed network, other peers will have the correct blocks and will propagate them through the whole network, leading to the isolation of the malicious user. Although all blocks have the "Parent block hash" value, the first block of the entire blockchain will have n zeros (0) (n being the number of bits in the hash used, 256 for SHA256) since there is no predecessor block. This initial block is called genesis block. In Figure 2.1 an example is shown of a stored blockchain in a peer. This example is correct, as it complies to the rule that a hash of a block needs to start with four zeros. This blockchain is the same for all the network and all the blocks are validated. In Figure 2.2 we present an attack attempt. In this attack, someone tried to cheat the network by changing the block 1, by making "Alias" receive more tokens than it was supposed. With this change, the attacker had to find another "nonce", which grouped with the information on the block, makes the hash function generate a new hash that complies with the imposed rule that a hash needs to start with four zeros. Despite the attacker being able to find a new hash, all the blocks after the one that he changed became invalid, since the parent hash of block 2 changed and the hash of that block no longer represents the information in it. One can say that he could calculate a new hash for all the blocks, and indeed he could, even though this is a very time-consuming task. But the security of the blockchain doesn't come only from the blocks themselves, but also from the whole network. In this case, as the attacker tries to propagate the changes to the network, he will get isolated since the majority of the network has different values and will not accept this propagation [19].

Transactions in blockchain are normally pseudo-anonymous. This may change in different blockchains, but usually, blockchains use asymmetric cryptography for communications. With

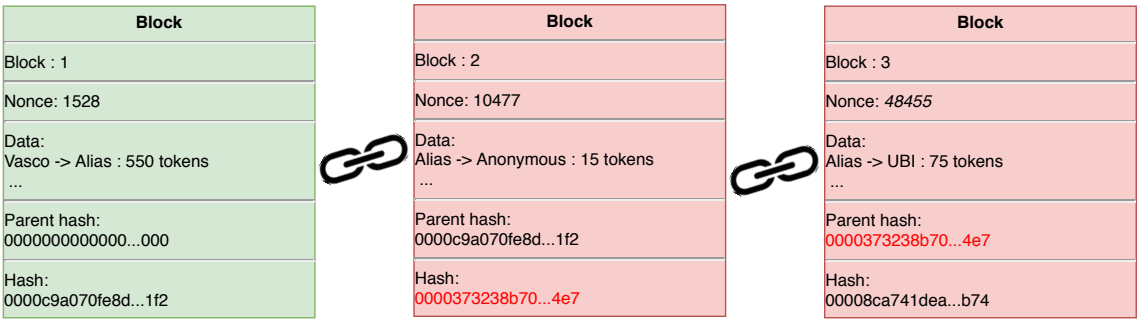


Figure 2.2: Blockchain stored in a peer. This blockchain shows an attack attempt and is incorrect.

this, a transaction has only the public key of the sender and the receiver, giving both secrecy of who they really are.

Bitcoin also introduced the Proof-of-Work (PoW) consensus mechanism. Consensus algorithms in blockchain are used to maintain consistency in the distributed network. The one used by bitcoin consists of making each node work, by calculating a hash value of the block, and changing the nonce, to achieve a hash value smaller than a target [20]. This is called a "cryptographic puzzle". When one node reaches a value that complies with the rules, that node broadcasts the block and the hash value to the network that will confirm the correctness of the hash value. In short, Bitcoin made it possible to solve many problems associated with digital currencies, such as the ability to trust unknown peers in a network, without the need for a trusted authority. This innovation behind Bitcoin inspired thousands of other applications over the past years.

Blockchain technology has suffered many developments, changing from the initial use focused on cryptocurrencies to decentralized applications. The literature indicates Ethereum [21] as the second generation of blockchain. Ethereum conducted a major shift in the blockchain technology, evolving from the simple use of blockchain for monetary transactions in form of currencies to a stage where the transactions take the form of smart-contracts that enable the value that is transacted to be programmable. The smart-contract functionality is based on the work done by Nick Szabo [22] and with it, Ethereum has grown exponentially since it was first described in the white-paper by Vitalik Buterin in 2013. Smart-contracts are pieces of code that are inserted into the blockchain, and they are executed when the programmed criteria are met. This allows users to exchange values other than cryptocurrencies (tokens), ranging from real assets to algorithms and to program smart-contracts to do repetitive work, for example, user $\mathtt{A}$ sends user $\mathtt{B}$, 1 token each time the user $\mathtt{A}$ reaches a multiple of 10 tokens in his wallet.

Ethereum design marked the beginning of the development of blockchain focused on applications, allowing developers to build decentralized applications over it. The platform is public, open-source and as said before, is based on blockchain that features smart-contracts. It also provides a Turing-complete virtual machine that enables the programs to be executed in a global network. Although these features are very desirable for real-world applications, the speed that a block is validated is low, mainly because of the consensus mechanism used, PoW. This means that the costs associated with validating blocks are high, due to the fact that every node tries to solve a cryptographic puzzle. This means that the way this platform is built, it will hardly be capable of supporting some specific applications, such as networks made of robots.

Since the problems of the biggest blockchains, like Ethereum and Bitcoin, became constraints for decentralized applications, the third generation of blockchain emerged. This third generation is still a working process, as there is not a complete released platform that solves the problems stated. But, there are many platforms that are indicated in the literature as being in the front line for the third generation, such as IOTA [23], EOS [24], Skynet [25], Tezos [26, 27] and Ethereum if the promise of changing the consensus algorithm from PoW to Proof-of-Stake (PoS) is fulfilled. These approaches propose solutions to the constraints of the previous generations and their approaches allow for the growth of new markets and opportunities since the restrictions to development are lower. The most common change in these new approaches is the introduction of new models for the consensus mechanism, which hugely contributes to lower energy consumption and to improve the number of transactions validated per second, which are the biggest problems of blockchain. Currently, bitcoin alone is using 73.12 terawatts per hour [28] and can only process, on average, 7 transactions per second.

Due to the fact that blockchain offers properties such as non-repudiation, replication and decentralization of the data, irreversibility, accessibility, (pseudo) anonymity and it can achieve consensus within an untrusted network [29], it increased exponentially in usage and as started to be used outside the financial scope. One of the most studied applications of blockchain outside the financial scope is in healthcare, where it has been used to privately store medical data with guarantees that only allowed people can see the registries [7], and to improve and accelerate healthcare research [8]. Blockchain has also been used in transportation systems [5], and extensively studied to integrate Internet-of-Thing (IoT) devices [30, 31]. One of the most promising uses for blockchain is its integration with AI. Both blockchain and AI are disruptive technologies that have immense utility and that have shown they can solve real problems by themselves. The integration of these two can solve problems that are inherent to blockchain [32], such as sustainability and efficiency by improving the energy consumption [33] and by providing dedicated mining [34, 35]. AI can also be used to improve blockchain and smart-contracts security [36, 32], by aiding in formal verification of the code. There are proposals that integrate the two in the form of decentralized applications that use AI for processing data and blockchain as a ledger. However, most of the proposals consist of using blockchain as a lower-level ledger to store monetary transactions and an upper-level application that contains a marketplace for selling AI algorithms and datasets [36]. There have been some initial proposals that try to integrate this to enhance robot characteristics and to conceptualize how a swarm of robots can be controlled over a blockchain and the benefits of such a system [36]. But, currently, there are barriers regarding the integrating of AI, robotics and blockchain together, which are slowing the development of proposals that integrate these technologies. These barriers are most prominent in the confirmation time of a transaction since most of robotic networks working or AI algorithms can't wait much time to decide the next task to do, and the high energy consumption. In systems that use heavy algorithms like the ones this dissertation will address, such high costs of energy are not acceptable, and in industries and factories that would use blockchain with AI to automate their processes, it is unthinkable that they will lend time and energy from the robots for blockchain computation, since it'll affect the robot productivity. We've identified that there was a gap in the field since there hasn't been enough development in algorithms that use the information of blockchain to change its state, do statistics or monitor its state. Despite the incipient development, there has been some initial work to reduce the problems and create an environment where AI can be used alongside blockchain, which could help massify robotic systems. The advantages that could be achieved from the combination of

these technologies include the decentralization of AI, more accuracy of the algorithms since the data is validated and it's immutable once inserted in the blockchain and a huge boost of the development of industry 4.0, since it will lower costs, increase productivity and aggregate software and hardware into one single platform.

In this work, we use a consortium blockchain to mitigate some barriers to integration of blockchain and robotics, and we leverage AI algorithms and the data contained inside the blockchain to create novel solutions to monitor and control robots and also to monitor robot workspaces.

In the next sections we present a detailed overview of the proposals that use blockchain and AI or robotics. We also dedicate a section to overview the state-of-the-art in detecting robotic anomalies.

2.2 Blockchain and Artificial Intelligence

In the White paper by Namahe [37], a real-world application of how blockchain and AI can work together towards a more transparent chain in the fashion world is shown. Their goal is to create a platform, based on the blockchain technology to aggregate all the different tiers and stakeholders of this industry into one single platform. This idea allows them to reduce costs to companies since they have everything they need in a single platform, ranging from sellers to buyers, and to enforce the transparency of all trades, ensuring that the workers are paid at least the minimum wage and also contribute to reducing child labour. The AI coupled to this blockchain has the job of acting as an invisible tracking machine that proactively identifies issues before they happen and suggests mitigation's to keep the supply chain running efficiently.

SingularityNET [38] is building a platform with the objective of having a decentralised platform that allows anyone to create, share, and monetise AI services at scale. They intend to democratise AI by creating a marketplace, over a blockchain technology based on ERC20 standard [39] over Ethereum [40]. This allows the authors to create and use a specific token for their platform, called AGI (a reference to, but not the same as Artificial General Intelligence (AGI)), that can be bought and traded with Ether (Ethereum token) or other ERC20 tokens. More concisely, the idea behind SingularityNET is both the creation of a platform where developers can monetise the algorithms they export to the SingularityNET's blockchain and also the integration of a high-level Application Programming Interface (API) that should allow everyone to easily build and deploy systems with numerous algorithms that are allocated in the blockchain. The API system works on the basis that a user is willing to pay to have an algorithm do a specific job. The algorithm chosen to conduct the routine may need other sub-routines, which can be done by other algorithms in the blockchain. These sub-routines are paid by the API that is handling the principal routine, which deducts the earnings of the first algorithm but allows it to perform an extensive work, even though it may not know how to do it. For the sake of exemplification, imagine that you want to create a software capable of detect profanity words in the news, you use the SingularityNET API and select one of the suggested algorithms for that job, but most certainly, some news will be in a foreign language and require translation. This is where the API handles the subcontracting of other algorithms to handle that sub-problem. The goal of the developers is that this will allow every user to easily use AI, despite the fact that the algorithms might be a black-box for the user. SingularityNET wants to provide a mechanism to create a

full network of different algorithms where which one of them is only built to solve one specific problem. This has as a foundation previous work conducted by some of the same authors [41]. This foundation allied with the decentralised network of algorithms allows the creation of some sort of Artificial General Intelligence (AGI), where a unique system can perform multiple and complex tasks. The authors like to reference the expressive robot Sophia [42] as one of the cases that serve both as a test and as a client of this AGI idea.

On a more economic side, Numerai [43] has built a unique concept over the blockchain. Their concept allows data scientists to make predictions on one of the toughest and more competitive markets, the stock market. They built a unique system that tries to nullify the fact that the stock market has very slow progress in the fields of Machine Learning (ML). This slow progress happens because the data is very sensitive and only a few data scientists have access to the raw data. Numerai overcomes this challenge using an abstraction of the data. This abstraction is built with a unique algorithm of ciphering the stock market data without losing the prediction aspect inherent to the data. With this new way of presenting the data, they are able to share it with their community and allow everyone to participate with his or her prediction algorithms onto that data. This, in the developer's words, is a way of overcoming the human bias and overfitting of the data. With the ensemble of prediction algorithms, Numerai holds a hedge fund that uses the prediction algorithms (tested on new data that developers don't have access to, to reduce the bias) to invest the capital, depending on the results and the direction that the algorithms take them. The blockchain is present in this project so that they can have a decentralised system of predicting algorithms and to have a specific ERC-20 token, called Numeraire. This token is used by the users to participate in the weekly competitions, where they bet Numeraires in their algorithm's performance. This serves the purpose of defining the "confidence" that the data scientist has on his algorithm, and higher confidence can lead to higher earnings if the algorithm passes the tests.

DeepBrain Chain [34] launched a platform where they aim to create a cloud of graphics cards. With the recent developments in the deep learning area, a large amount of data is required to train the most proficient models, but this implies that the developers have access to multiple graphics cards. But, even if developers have access to some cards, the time taken to train a Neural Network (NN) such as a Convolutional Neural Network (CNN) can take days. So, DeepBrain chain created a way of allowing the miners of cryptocurrencies to mine the blockchain they created on their platform, but, instead of doing a typical proof-of-work, they are actually allowing others to use their graphics cards. This way, researchers and developers can have access to thousands of graphics cards. The blockchain ensures that all the parties comply with their agreement and that the data that is being used to train the NNs is not accessed by people without permission. This is all done using smart-contracts over the blockchain with an ERC-20 compliance token. With this platform the miners get more profit for their smart-contract mining and graphics card renting, the users spend less money because they have no need for expensive physical equipment. This platform can easily be used to build an AI ecosystem dedicated to robotics.

Another project that has some similarities with DeepBrain Chain is Neuromation [35]. The underlying idea is the same, they try to leverage the distributed computational power associated with the blockchain to allow developers to train NNs. The main difference is that the Neuroma-

tion Market Platform unites the scientific community, market resources and commercial and private entities in one single place. This market allows anyone to buy or sell datasets, models and AI services, like labelling of data. The decentralised computing power also allows this marketplace to sell AI models or train a brand-new one. Neuromation has practical applications already in place by their partners, mainly in medical detection problems and industrial robots by training models with synthesised data.

Aitheon [44] is a platform that has been developed for over a decade and has recently adopted a blockchain technology based on the ERC-20 standard. Their goal is to have a complete platform that can reduce the number of time-consuming tasks that developers have to endure, like organising documents. Their solution is a platform with both AI and Robotics that can provide automation for specified business processes. The platform they built has 5 modules. The first, AI module, tries to retrieve information about frequently performed tasks and takes actions to automate those tasks. The second module, called Digibots, is very similar to the first one, but this module focus on automating programming tasks, like back-end solutions and data-driven problems. The third module, Mechbots, is focused on helping businesses integrate robotic automation to increase efficiency and productivity. The last two modules, Aitheon Specialists and Pilots are human specialists that conclude the tasks that can't be fully automated and also provide robot supervision. In short, Aitheon provides a Platform As a Service (PaaS) that provides a platform easy to be used by a wider audience (even with no-tech skills), that aims to automate time-consuming tasks, that is capable of integrating new robots without effort and at the same time, reduce the payment latency and increase security by using an ERC-20 token that can be used by an AI algorithm to do payments.

Eligma [45] is a company that aims to transform the way people interact with products, allowing them to easily buy, track and sell all kinds of products. They developed a platform that has two cores. The AI core, consists of a chatbot, that is designed to help users to interact with the platform, a product-matching algorithm, that depending on user inputs and preferences matches them with different products that are on the platform and finally, a value-prediction algorithm that does a present and future estimation of all products that are on sale. The second core is the blockchain, where they use the Ethereum blockchain for protection and as a side-chain for data storage, reducing the costs of maintaining the main chain with all the data. The blockchain also allows the platform to have a decentralised sales point, where two parties have a mutual agreement on some product and this becomes a smart-contract over the blockchain, introducing more security and ensuring that both parties comply with the contract. The blockchain is based on the ERC-20 standard, which allows the developers to introduce the idea of making a payment over the blockchain possible with multiple currencies. This means that a user can pay for a product with both Bitcoin, Ethereum token, ELI (Eligma token), united states dollar or make a crypto-based loan.

The work done by TraDove [46] is very similar to the one done by Eligma but focused on businesses. It aims to provide a way for different businesses to promote their products and match with sellers and buyers that have similar interests and profiles. This matching is done by AI technologies. With the blockchain layer, they introduce a payment that lowers the costs regarding taxes when sending money to multiple countries, but, they aim to use this layer as a way to induce transparency about the exchanges and allowing the sellers to advertise their

products, with the B2BCoin (TraDove token), to specific targets.

A completely different platform is the one created by AdHive [47], where they focus their product only to the advertisement market. The platform is heavily based on AI algorithms and blockchain. It has two types of users. The first type is the advertisers, that submit videos or other pieces of information that they want to be advertised on social media and blogs and specify a maximum budget for the advertisement. This request for advertisement enters the blockchain as a smart-contract that retains the budget until all the conditions are met or until the contract is cancelled. The second type of users are the influencers, that have the means to advertise the information. The AI algorithms verify if the influencers put the correct information on their dissemination media and apply the information they acquire into the blockchain so that the smart-contracts are fulfilled.

Matrix [48] is a project that wishes to solve four problems with the blockchain technologies: the barriers associated with smart contracts due to the fact that these are usually done in specialised programming languages, the lack of security associated with the smart contracts, the slow transaction speed and the inflexibility in managing and updating blockchains. The first problem is solved in this platform by introducing a NN that is able to automatically convert simple scripts that contain information regarding the transaction conditions and the inputs and outputs to a smart contract. This enables users to create smart contracts. The second problem is created by having smart-contracts calling external functions and by having an open and decentralised approach to the smart-contracts. Matrix solves this problem by providing 4 extra components, a rule-based semantic and syntactic analysis engine for the smart contracts, a formal verification toolkit to provide the security properties of a smart contract, an AI-based engine that detects problems in the transaction models and checks their security and lastly, a deep learning based platform for dynamic security verification and enhancement. The third problem is a well discussed one, given the fact that Bitcoin takes about 60 minutes to confirm a transaction and Ethereum can achieve about 15 transactions per second, major sellers and some real-world problems can't leverage the blockchain technology. Matrix accelerates the transfer per second by allocating the PoW processing in a delegation network, which incurs smaller latency because the number of nodes is drastically reduced. The developers select the delegation network in a random way in the sense that the probability of a node to be selected is proportional to its PoS. The platform claims to have achieved 50 thousand transactions per second on their testnet. The fourth problem presented by the developers is overcome by giving access control and routing services to allow integration of private chains into a common public chain. This integration allows information to flow between the chains to meet the user criteria. Matrix also uses a reinforcement-learning algorithm to optimise the parameters regarding the transactions. The last thing that Matrix introduced on their platform is a new mining mechanism in which miners perform Markov Chain Monte Carlo computing, reducing both the time spent on each transaction and the waste of energy.

Marketplaces are one of the most common applications of blockchain technology. The AI Technology Network (ATN) [49] is one of those that leverage the blockchain to propagate AI services over a marketplace. ATN is built over a smart-contract enabled blockchain so it can provide datasets and AI algorithms in a secure and trustworthy way. The developers introduced an open interface blockchain platform that enables to solve some interoperability issues between AI and

smart-contracts that complies with the major standards. ATN makes it possible for users, developers and sellers to share services without concerns about security using a robust system that integrates cross-chain (multiple blockchains, especially Ethereum, Qtum, RSK and others) compatibility and a platform design that ensures the interoperability among the services on the marketplace. The ATN platform encrypts all the services that are integrated into the platform so they can ensure that the developers that integrate services on their marketplace are remunerated for their work and have the assurance that their product is secured. This platform is open-source to encourage the development of new applications.

Cortex [50] propose a solution for the problems associated with smart-contracts. The fundamental idea is a blockchain where users can augment their smart-contracts with the AI system proposed by Cortex. This enables the creation of decentralised applications that can react to both external and internal variables in the blockchain. Cortex built this system over a public blockchain of their own but in the early stages, they had integration with an ERC-20 token. The system also has other layers upon the blockchain. One of those layers is a platform that encourages developers, through token rewards, to upload ML models and to optimise existing ones. This collective effort may lead to achieving AGI over the Cortex platform, allowing decentralised AI applications to be very robust and highly complex. Cortex has its own virtual machine which is compatible with the Ethereum Virtual Machine (EVM), allowing it to work with Ethereum smart-contracts. Users can pay Cortex tokens to use the Cortex virtual machine in order to use AI algorithms that are under the Cortex platform. As consensus over the Cortex smart-contracts, all nodes must agree on the outcome of the inferred result.

Application of blockchain and AI in the medical field can lead to great improvements both for patients and for doctors. Some work on this has been conducted by NAM [51]. NAM is trying to revolutionise the state of medical care, with a focus on Japan. It's a project that integrates these technologies to try to reduce medical costs, helping patients improve their health and allowing doctors to grasp the patient's information regarding a disease or a treatment. This platform, called NAM Chain, intends to build several AI services. The first one is a consulting bot that can be used by any person. It works by introducing the symptoms on the app and the bot will say how urgent the symptoms are. The second one is a set of prediction models to detect diseases by scanning medical reports. The third one is a service that serves as a nutritionist, recommending healthy food based on the user's lifestyle and constitution. The fourth one is the basis of the platform, where all the services work, which the developers call "a system of medical records with deep learning and blockchain" [51]. This blockchain is based on the Ethereum network and blockchain but the developers want to migrate to a private blockchain in the future. The way to fight the resistance of integrating medical information, namely large images, in the blockchain is by giving the miners NAM Coins (NAM Token), which will be usable in clinics and in the AI services that the platform provides. They state that they developed a Top-K shortest path distance from sender to receiver to speed up the package transmission. The developers also assure that they work in security and that they want every report to be secure but they are very vague on explanations and don't have a clear explanation on how they will prevent people from accessing other peoples' information and how they will assure that the information on the blockchain can't be seen by those who should not have permission to see it.

Deeraj Nagothu et al. [52] propose a novel approach to tackle the problem of security in traditional surveillance systems. As Smart Cities develop in today's world, the Internet of Things (IoT) technology becomes more important, however, it also compromises the security of the data. Traditional surveillance systems normally work with huge amounts of data and the tasks associated are growing. These tasks include people identification, aggression detection and others. Normally, these systems are designed with a centralised architecture with many servers to process the data but these designs are vulnerable to single point of failure and usually, data leaks because of the lack of protection associated with the surveillance feed. The solution proposed by the authors to tackle these problems is a system based on microservices architecture and blockchain technology. The microservices serves to isolate the video feed into different sectors, improving the system availability and robustness because the operations become decentralised. The blockchain serves as a base to synchronise the video analysis information to the microservices, proving security in a trustless network. They introduce smart contracts in their system to increase security, in order to prevent any unauthorised user from accessing the data they do not own. This offers a scalable and decentralised access control solution to the problem. By having the system distributed by microservices has the advantage of allowing continuous development and continuous delivery without interrupting the whole service.

In the work done by Jianwen Chen et al. [33] a novel method to solve the problems associated with consensus on the blockchain is proposed. In most of the working systems based on blockchain technology, the consensus is reached by PoW, where every node receives a cryptopuzzle that needs to be solved or PoS that takes into account the amount of stake each node has. Both can take a long time to reach consensus and some approaches may even lead the whole blockchain to a centralised system and to high energy consumption to conduct the consensus process. The proposed framework to solve these problems is based on AI technology. The framework works as follow: first, a CNN calculates the average transaction of each node, then, statistics about the threshold values of the average transaction nodes are created. These values serve to categorise the nodes into three categories, the super nodes, random nodes and validator nodes. The super nodes are the ones that have lower latency and more computational capability. The random nodes guarantee the fairness of the network. The CNN used to classify each node is based on the AlexNet [53]. This CNN receives information about the state of each node, which includes the computing power, online time, payoff and latency, and with this information it outputs the node classification. The flow of the process goes by receiving a new block data and calculating the super nodes and random nodes that create a block to be coupled to the blockchain and send that block to the nodes pool to be verified. The authors conducted many experiences with this method that show that it can be compared with PoS and PoW in terms of security, latency and costs and still has reasonable results.

Tshilidzi Marwala and Bo Xing conducted a study about blockchain and AI [32] and present some of the problems associated with smart-contracts. The authors say that AI is the core of the new industrial revolution and that blockchain is a technology that can be integrated with AI to make it even more powerful. Based on their research, the authors infer that the blockchain is not decentralised in the sense that the underlying development is attributed to a cluster of developers and show-case the smart-contracts as an example of how this can be tragic. In the past, many attacks were conducted that targeted smart-contract vulnerabilities [54]. These vulnerabilities were mostly introduced because the smart-contracts are essentially a collection of functions and data that are programmed by different human programmers. This makes the

smart-contracts likely to have flaws, so, the authors show a brief overview of how AI can be used to reduce the number of errors and flaws in the smart-contracts so that they can be delivered to the blockchain with increased security. They propose an idea that can solve this problem: an AI algorithm to automatically perform formal verification of the smart-contracts to ensure correct execution, in other words, that the smart-contracts do what they are supposed to do. The authors also introduce the concept of using computational intelligence to improve security, for example, the use of evolutionary computation to improve cryptanalysis, which can lead to the creation of more robust cyphers and consequently improving blockchain system's resilience. The authors also highlight the possible solution to the inefficiency of AI for consensus and suggest that specialised computer components to run NNs could be the solution to improve both the speed and the efficiency of this process.

ABBC foundation proposed a method that uses computer vision to improve the security of wallets that hold tokens by using face recognition algorithms to identify the owner of an account [55]. This approach uses AI not to improve the blockchain itself or to develop methods that use it, but rather to improve the security of the software that is used to manage cryptocurrencies.

2.3 Blockchain and Robotics

The work conducted by Bruno Degardin and Luís A. Alexandre (as supervisor) [56], shows how it is possible to create a blockchain and use it to store robotic events. The idea is to ultimately allow the creation of smart-contracts that use information acquired on the wild by different robots (possibly from different manufacturers) and have action-triggers based on the contracts that are stored and verified on the blockchain. This can ultimately improve productivity in a factory and reduce the time spent on doing tasks like refilling the screws, in which the robot uses the blockchain to indicate that it needs more screws to continue its work. As a follow-up to this work, and now using Tezos' technology [2], the authors propose the creation of a blockchain for robotic event registration that takes advantage of the improved security provided by the formal verification embedded into Tezos. This on-going work plans to adapt the blockchain to support many more events per second than the current specification enables, and to allow for the system to deal with a large number of interacting robots.

Eduardo Castelló Ferrer presented in [57] possible benefits of combining blockchain technology with robotics, especially swarm robotics and robotic hardware. The advantages of robotic swarms are how easy these networks can scale and their robustness to failure. These advantages come from the fact that members of these swarms are distributed. In the industrial sector we can also see how this market is growing and allowing companies to achieve higher productivity, which is the case of AmazonRobotics [58] that has been showcasing its army of robots that co-operatively work to manage their warehouses [59]. Most of the robotic swarms only use local information, this means that a robot only has information about itself and, in rare cases, information regarding robots that are close to it. The integration of blockchain in these systems can give robots global information, which can be useful for different applications. The blockchain can also improve the speed of how the system changes the behavior since having global information allows the whole system to quickly change behavior to address specific robot needs. This can also be done by a controller robot that evaluates the system state by using the information inserted into the blockchain to define the changes and commit them to the blockchain to be

Table 2.1: Short overview of the methods presented in section 2.2. Acronyms used in the table: Artificial Intelligence (AI); Blockchain (BC); Byzantine-fault Tolerant (BFT); Delegated Proof-of-Stake (DPoS); Proof-of-Concept (PoC); Proof-of-Importance (PoI); Proof-of-Stake (PoS); Proof-of-Work (PoW); Smart-Contract (SC).

Name	Problem	Solution	Consensus	Token	Compatibility
Namahe [37]	The separation of the different stakeholders of the fashion world (sellers, buyers, retailers and producers).	A unified marketplace that uses AI services to make it more transparent and that automatically identify problems.	BFT	NMH	ERC-20
SingularityNET [38]	Integration of different AI services so they can work together seamlessly.	Marketplace for algorithms. API that automatically call third-party algorithms to solve defined problems.	PoW	AGI	ERC-20
Numerai [43]	The confidentiality associated with data from the stock market.	Platform where anyone can download encrypted data regarding stock market and participate in competitions with AI algorithms that serves as guide to the Numerai Hedge-fund.	PoW	NMR	ERC-20
DeepBrain Chain [34]	The lack of resources to build AI services.	Miners get paid to lend their graphics cards to train neural networks instead of the traditional mining.	DPoS and PoI	DBC	ERC-20
Neuromation [35]	Dispersion of models, datasets and AI services; Lack of computing resources.	A marketplace and ”lend your GPU” mining process.	PoW	NTK	ERC-20
Aitheon [44]	Time-consuming tasks that developers endure.	Automation with AI and Robotics.	Multi-Blind	AIC	ERC-20
Eligma [45]	The difficulty of people to sell, purchase and resell products.	A marketplace based on AI services with product-matching and chatbots to help users.	PoW	ELI	ERC-20
TraDove [46]	The difficulty of business-to-business contact.	Marketplace that introduces businesses to each others by using AI.	-	BBC	-
AdHive [47]	High expenses on the marketing market.	Advertisers insert advertisements as SCs and AI services matches them with influencers.	PoC	ADH	-
Matrix [48]	Slow transaction speed; Specific languages for SCs; Inflexibility of the BC.	A NN to convert scripts to SCs; Increased security by having rules associated with SC; By using AI and different consensus.	PoW and PoS Hybrid	MAN	-
ATN [49]	Security concerns about selling AI services.	SC based marketplace to provide datasets and AI algorithms in a trustworthy way.	-	ATN	ERC-20; ERC-223; Qtum; RSK
Cortex [50]	Difficulty in integrating services in SCs.	An AI system that allows users to augment their SCs.	Cuckoo Cycle PoW	CTXC	ERC-20
NAM [51]	Low-quality healthcare.	Store all medical records in a BC and a platform with AI services to give suggestions and to help users access their data.	PoS	NAM	ERC-20
D. Nagothu et al. [52]	Security in traditional surveillance systems.	System based on microservices architecture and BC technology.	-	-	-
J. Chen et al. [33]	Validation speed.	CNNs that classifies nodes as super, random and unknown, to speed up transactions lower the energy consumption.	AI-based	-	-
T. Marwala and B. Xing [32]	Conducted a research about BC and AI.	AI services that provide automatic formal verification of SCs and use of computational intelligence to improve security.	-	-	-
ABBC foundation [55]	Wallet’s security.	Face recognition to recognize the owner of the account.	-	-	-

conducted automatically. These improvements and the global information of the system can lead to higher productivity and easier maintenance both in robotic swarm environments and in environments where robots are working individually but require changes to their behavior.

In [60], the authors present Robochain, which is a conceptualization of a method to share critical data among robots in a secure way. This is presented as a framework to tackle privacy

issues regarding using personal data by robots during a Human-Robot interaction. This method uses MIT OPAL [61] to provide a layer of security and ensure the privacy of the information represented on the data, and with this, the robots can improve Machine Learning (ML) locally with the information they acquired and publish it on the network. The blockchain technology is a part of this framework for transparency and ledger, storing events and to validate the published models. The authors also propose a consensus mechanism so that every node can vote on which model to accept by taking advantage of the smart-contracts technology within the blockchain.

The authors of [62] proposed a methodology to create a coalition of robots, sensors and actuators. The coalitions are created by having all the information passing by a knowledge processor and then, insert it into a blockchain. This allows every node of the coalition to have the same information and allows the use of smart-contracts to adjust robot actions and reallocate resources. Citing the authors, this systems provides immutable distributed storage that is crucial to negotiate separated tasks among different participants (robots).

T. L. Basegio et al. proposed an architecture to allocate tasks within multi-agent networks using a private blockchain [63]. This is built on the idea of having communication and coordination throughout the whole system. This proposal uses blockchain as a ledger that only registers the events to be securely stored and has a platform built outside of it to allocate tasks to the robots.

There has been work done in integrating robotics and blockchain to improve the security of robot swarms, which is the case of [64], where Strobel and Dorigo introduced a method that uses a reputation system to manage robots inside a robotic swarm, which serves the purpose of mitigating the propagation of bad robots actions to the whole swarm. This method uses blockchain to provide a mechanism to have a shared knowledge system. The other proposal that aims to improve security is [65], in which the authors proposed a method based on smart-contracts to reach a consensus in robot swarms that can have byzantine agents. This proposal makes it possible to achieve consensus over an untrusted network of robots by having the robots voting.

Eduardo C. Ferrer proposed a method to provide secure and secret cooperation of robot swarms [66]. The method does not use a blockchain, but instead it uses one of the fundamental components of blockchain technology, the Merkle Trees. The idea is that cooperative robotic missions are encapsulated in a Merkle Tree and individual robots share cryptographic proofs of their work without disclosing the high-level mission. This method was demonstrated by having a swarm of robots creating a maze and by collecting pieces and returning them to a predefined position. In these, the robots didn't know the mission itself but knew that the action they were conducting was part of it.

Applications that integrate robotics and blockchain are also being developed in the industry, which is the case of Kambria and Robonomics. Kambria [67] is building a platform similar to Android but with focus on robotics, in which high-level libraries, modular hardware and software are available in order to speed up robotic development. Robonomics [68] is working on a platform that aims to soften the human-to-robot and robot-to-human interaction by having a marketplace of liability contracts.

Table 2.2: Short overview of the methods presented in section 2.3. Acronyms used in the table: Blockchain (BC); Convolutional Neural Network (CNN); Machine Learning (ML); Proof-of-Work (PoW); Smart-Contract (SC).

Name	Problem	Solution	Consensus
B. Degardin [56]	Robotic Event Recognition.	Proprietary BC for faster block validation.	PoW
RobotChain [2]	Integration of robotics and BC; transaction validation speed; how to control such a system over a BC.	Tezos technology, SCs and AI to improve performance and quality of distributed robotic systems.	-
E. C. Ferrer [57]	Integration of BC with Swarm Robotics.	Explains how BC can help solve issues in Swarm Robotics. Presents a set of problems that needs to be overcome.	-
Robochain [60]	Privacy issues regarding the use of sensitive and protected data.	Interconnected Robotic Swarm that change behaviour depending on the data, and is able to improve ML algorithms and propagate them to a BC without compromising the privacy of the data.	SC based
N. Teslya and A. Smirnov [62]	Creation of coalitions of intelligent robots.	The use of knowledge processors to insert information acquired by the robots into a blockchain and SCs to manage the system.	-
T. L. Basegio et al. [63]	Allocation of tasks in multi-agent networks.	Blockchain to register the decisions made and a platform outside the blockchain to allocate tasks to each robot.	-
V. Strobel and M. Dorigo [64]	Shared knowledge in robot swarms.	Use of blockchain to have a shared ledger and a reputation system to manage the swarm.	-
V. Strobel et al. [65]	Byzantine robots in robot swarms.	Smart-contract consensus where robots vote for the propagation of information. Byzantine robots get isolated from the network.	SC based
Eduardo C. Ferrer [66]	Security and secretness of robot swarm cooperative missions.	Use of Merkle Trees to store the hash of the robots' actions and the entire mission to be executed.	-
Kambria [67]	Development of robot applications that are time-consuming.	Platform in which high-level libraries, modular hardware and software are available to speed up robotic development.	-
Robonomics [62]	Human-to-robot and robot-to-human interaction.	BC approach to have a marketplace of smart-contracts that allow easier interactions.	-

2.4 Robotic Anomaly Detection

Bojan Jakimovski and Erik Maehle proposed RADE, a method inspired in the Natural Immune System for Robot Anomaly Detection, with focus in detecting failures in autonomous robot systems [69]. The system was tested with the aid of the robot OSCAR [70], which has 6 legs with 3 motors per leg. In this method, the authors use clonal selection, combined with fuzzy logic for representing the information. Fuzzy logic is used to allow the method to categorize the anomaly detected, depending on a range of values. The method works by defining two sets of rules: one set defines all the rules that detect when there is some anomaly present while the other defines the rules that represent a state where an anomaly is not present. These rules also have a "weight", which is the part of the Artificial Immune System (AIS) in the method. This weight serves the purpose of incrementing or decrementing the value of the anomaly presence, to adjust the range of values used by the fuzzy logic. With this method, the authors were capable of detecting anomalous situations in their experiments against the normal behavior of the robot OSCAR in tasks that involved walking with abnormal conditions, such as collisions and joints getting disconnected.

Huimin Lu et al. presented a method that, based on reinforcement learning, is capable of detecting motor anomalies in drones [71]. The algorithm works by analyzing the information about the temperature change and deciding if the change is critical or not, indicating the presence of an anomaly. This analysis is complemented by using the information about the speed within the reinforcement learning algorithm to adjust the threshold temperature. This threshold indicates whether the drone should continue with the same speed or if it should decelerate or stop to cool down and check if the anomaly persists. The authors validated this method by conducting experiences in which a temperature sensor and a raspberry pi 2 were coupled to the drone. Despite that the experiences were conducted in a small time-frame, the method was capable of keeping the drone under the temperature threshold by conducting speed deceleration.

Dong Zang et al. proposed a method to detect anomalies in industrial applications [72]. The method uses Markov Chains to extract an array of features, which they entitled "Markov Feature". This consists on a one dimension vector with all the probabilities of state transitions. The state transition frequency was extracted from the dataset, where the authors defined 4 regions of interest in order to detect abnormal occurrences. The dataset used to build and test their method consists of time series information of the pressure from experimental pipelines in the city of Beijing. From the data, the authors also extract the mean and the variance and use them as features. With these features and data, the authors used 4 different algorithms to classify the data: K-Nearest Neighbor (k-NN), Decision Tree (DT), Random Forest (RF), and Support Vector Machine (SVM). All of the algorithms surpassed the classic methods of feature extraction, i.e., Statistic-based, Wavelet-based and time and frequency domain method, in the task of detecting anomalies.

The work presented by Wallace Lawnson et al. [73] uses Generative Adversarial Networks (GANs) [74] to identify indoor environment anomalies. The GAN was trained by teleoperating a robot within an indoor environment in which the images were split into patches and then used to train the algorithm. By doing so, the authors were able to create a method capable of locating anomalies in indoor environments without the need to store images for comparative analysis. This method was evaluated in the same indoor environment as the training phase, in a scenario where some objects were missing, e.g. a door handle, and was capable of detecting the anomalies. However, due to the fact that it uses a GAN approach to check for anomalies, this method is not scalable since it only works in the environment where the algorithm was trained.

HuiKeng Lau et al. developed an immuno-engineering approach to detect anomalies in robotic swarms [75]. The work focus on creating a simulation of a robotic swarm containing 8 robots in which their task is a collective work to pick food. The authors simulated both static and dynamic environments in which they induced anomalies to evaluate the proposed method. When no anomalies occurred in the whole simulation, the energy consumption and the traveled distance of the robots is very similar. However, when different anomalies were induced in the swarm system, the individual performance of the robots decreased immensely (either for a period of time, while the swarm recovers from the anomaly, or until the robots stop completely) and the productivity of the whole swarm has a momentary fall. This work shows us that Artificial Immune System (AIS) approaches used in combination with simple features such as mean and variance can detect robot anomalies.

Fabio González and Dipankar Dasgupta [76] compared two methods in the task of detecting anomalies in syntethic data. The dataset used was generated using the Mackey-Glass equation:

$$\frac{dx}{dt} = \frac{ax(t-\tau)}{1+x^c(t-\tau)} - bx(t) \quad (2.1)$$

where $a = 0.2$, $b = 0.1$ and $c = 10$.
The equation was solved using the fourth-order Runge-Kutta method. With this, the authors generated 1500 normal samples, from which they discarded the first 1000 to eliminate the "initial value effect" and conducted the same method to generate abnormal samples by changing one of the equation parameters in the final 100 samples. Then, compared a simple Multi-layer Perceptron approach against a Self-organizing Map trained only on the normal samples and concluded that both methods achieve similar results in the detection of the abnormal samples.

In [77], the authors propose a method based on Stacked Denoising Autoenconder (SDAE) to denoise and extract valuable features from rotating machinery vibration signals in order to detect anomalies. This method is compared against a Deep Belief Network because the authors found no previous work that they can compare to. The SDAE achieved higher accuracy and has the advantage that it allows the extraction of features from raw signals and also has a high generalization ability because it is pre-trained using unsupervised learning. To evaluate the methods and compare them, the authors used the rolling bearing dataset and gearbox dataset from [78].

Huo et al. presented a short overview of methods for fault diagnosis of rotating machinery that use entropy techniques [79]. In the paper is stated that the data flow of the diagnosis in the methods studied usually contains either 4 or 5 steps: Data Acquisition, Signal Processing and/or Feature Analysis, Feature extraction and Fault Pattern (classification). From this survey, we extracted information that of the 20 studied papers, 11 papers focused on detecting faults on bearings, 4 on motors, 3 on gearboxes, 2 on shafts and 1 on gear and bearing parts. Most of those methods monitor vibration signals, however, some stand out: [80, 81] that use current signals and [82, 83, 84] that monitor acoustic emission signals.

In [85], a method based on a deep neural network architecture that is capable of fault diagnosis without the need for previous feature extraction is proposed. The deep network, based on stacked autoencoders that are pre-trained individually, is capable of detecting and classifying faults in data obtained from bearings [86]. This work has the advantage of removing the need for manual feature extraction because the network is capable of acquiring the features from raw signals by itself. The problem with this approach is that it is only capable of detecting failures in data regarding bearings. Generalization implies the need to train a new deep network, which could imply changes in the architecture itself, which can be time-consuming tasks. [87] shows a similar process that compares a traditional, manual feature extraction, against a feature learning method in the form of a CNN. The CNN model is applied to raw amplitudes of the frequency spectrum of vibration data (acquired from two accelerometers). This work shows that an end-to-end deep learning system can out perform the traditional feature extraction plus classification, improving the fault detection accuracy and removing the need for human analysis on the data to extract features.

O. Janssens et al. proposed a method that uses infra-red images to conduct condition monitoring, specifically to detect rotor imbalance and at the same time, perform fault detection [88]. The authors use a private dataset of five different bearings in which they use an SVM and a Random Decision Forest to classify 5 different faults and detect rotor imbalance with 88.25% accuracy. A similar approach, using CNNs [89], show how it's possible to use infrared thermal images acquired from videos to detect faults. In this work, the authors use transfer learning to mitigate the problem of the small size of data, which can be an issue when working with deep learning because if there are a small number of data to train, it might not be enough to train a deep network, because the input values are diluted throughout the network and may not be enough to adjust all the network weights.

In [90], a method to detect anomalies based on summarization and data storage is proposed. By storing large amounts of data in an organized way, by using metadata to know the localization of the data and by compressing it with summarization techniques, the authors were capable of reducing the space required to store large amounts of data and improve the efficiency when searching for specific contents when compared to traditional methods. Then, by using representations of the data, the authors cluster them to detect anomalies. The clustering process is done in two steps, the first one uses the Clustream algorithm [91] in order to obtain representations of the acquired data. The second step is the application of the X-means algorithm [92] to cluster the data incoming from step one. Finally, a threshold is compared with the results to define if they represent anomalies or not.

2.5 Conclusion

This chapter presented a comprehensive review of the concepts behind blockchain, the integration of this technology with AI and robotics and an overview regarding anomaly detection. First, we reviewed the most relevant approaches in each one of the aforementioned topics. With regard to blockchain, we explain the inner components of blockchain, how consensus is achieved and how the information is stored in a decentralized manner. It was interesting to note that the number of proposals that integrate blockchain and AI are increasing but are fundamentally focusing on the same, developing applications that use blockchain as a ledger, which are in most cases marketplaces for algorithms. Similarly, we saw an increasing interest in the integration of blockchain with robotics, with swarm robotics as the most sought after robotic field to be integrated in blockchain because it allows the entire swarm to have global information and properties like security and network management that are usually harder to achieve in traditional methodologies, due to implementation complexity and resources consumption. Finally, we reviewed the state-of-the-art methods in detecting robot anomalies, from which we detected that most of them use private databases which makes it harder to compare methods and that there are two major approaches to detect anomalies: the first focus on using AI algorithms, such as CNNs and AIS, while the other focus on using entropy techniques.

Chapter 3

RobotChain

This chapter serves the purpose of explaining RobotChain, which is a blockchain based on Tezos technology. First, we introduce the Tezos blockchain, namely, its properties and components. With this introduction, it is possible to present RobotChain and the changes that were conducted in the Tezos blockchain to create this new blockchain that is dedicated to robotic environments. In this chapter, we also present the history behind smart-contracts and how can they be built on RobotChain. This is done because the work presented in this document uses RobotChain as a decentralized ledger and smart-contracts as communication gate and as an action-trigger software that acts upon the blockchain and the robots autonomously.

The structure of this chapter is as follows: section 3.1 presents a detailed explanation about the Tezos blockchain, its software and its differentiating characteristics when compared to other blockchains. Section 3.2 explains the creation of RobotChain and the changes that were made to the Tezos blockchain. Finally, section 3.3, explains how smart-contracts work and shows an example of a smart-contract written in two different languages that work in RobotChain.

3.1 Tezos Blockchain

L.M. Goodman presented in 2014 his positional paper [26] where he proposed Tezos and later presented a white paper [27] that explained in detail all the advantages that his proposal could achieve. Tezos' base goal is to address four problems in Bitcoin, which were detected by the author. The first problem is the inability for Bitcoin to dynamically innovate, which requires that a new blockchain is created when changes to the core technology need to be done (the "hard fork" problem). The second problem is the cost and centralization issues of the PoW consensus algorithm, which is computationally expensive and pools of miners can concentrate enough computational power to control the mining of the entire network. The third problem is the limited expressiveness of the Bitcoin transaction language that limits smart-contracts. The last one is the security concerns regarding the implementation of a crypto-currency and the developer-induced errors in smart-contracts. So, Tezos is a platform for smart-contracts and decentralized applications that aims to have on-chain governance in which the stakeholders can govern the protocol and decide future changes. Tezos also has improved security because it is designed to facilitate formal verification, which can mitigate many flaws in the code. This is achieved by having the code written in OCaml and by presenting a new language, Michelson [93], for smart-contracts so those can be formally verified. The seed consensus is Delegated Proof-of-Stake (DPoS) based on Slasher [94], chain-of-activity [95] and Proof-of-Burn (PoB) [96]. With this consensus, a stakeholder can delegate its Tezzies (Tezos token) to other stakeholders so they can "bake" (miners are called bakers in the Tezos chain) with his currency. By having a modular system that combines the transaction and the consensus protocols into a "blockchain protocol", the blocks in the blockchain are defined as operators that can act on the state of the chain, allowing the blockchain protocol to become introspective, allowing the blocks to act

on the protocol itself. This is a major advantage of Tezos technology because it enables the stakeholders to vote directly on protocol upgrades to the system and avoids hard-forks. A baker is selected to create a new block when one of their Tezzies is selected. This act improves the security of the system because Tezos forces the bakers to deposit Tezzies to validate their work and if they act honestly (don't attempt double spends or propagation of blocks from different branches), they are rewarded, otherwise, they are punished. Tezos also has endorsers, which are stakeholders that are asked to witness the creation of a block and verify its validity. In short, the miners are called bakers in Tezos, and they can earn the right to include a block into the blockchain through a lottery. As opposed to normal lotteries, in this, the chances of winning the lottery depends on how much tokens the delegate has access to. A major change from the normal approach is that with this, not everyone needs to be a baker, as one can delegate their tokens to a baker. Individuals that delegate their tokens are called delegators. This whole system can be very interesting towards improvements in robotics and AI because the self-amendment allows for changes in crucial parts of the blockchain without requiring hard-forks, the formal verification can increase the level of confidence in other participant's information, the faster consensus allows for more users to register their events onto the blockchain and it allows the creation of new AI algorithms that leverage the formal verification or the changes of the blockchain protocol.

As aforementioned, Tezos formalizes blockchain governance by allowing the holders of tokens to vote for changes regarding the blockchain. The amendment process is divided into 4 different periods, each one containing 32768 blocks, which translates in approximately 3 weeks per period, taken into account the current baking cycles. This means that from the proposal to activation, the time elapsed is roughly 3 months. During any of the periods, any problem that leads to failure in proceeding to the subsequent period forces the whole amendment process to revert and a new one starts, in other words, it is restarted. The first period is the proposal, anyone with at least 10000 tokens (their own or delegated) can submit proposals within the blockchain, using the proposal operation. Each individual can submit a maximum of 20 proposals per Proposal Period and when submitting a proposal, his vote is automatically counted. Each proposal has an invoice attached, which is a compensation for the work (if the proposal is accepted). This compensation structure adds incentives for developers to improve Tezos. At the end of the proposal period, the most favorably voted proposals advance to the next period. The second period is the Voting period, where bakers/delegates can vote on the proposals that reached this period. The possible votes are: "Yay", "Nay", "Abstain", which represent upvote, downvote, and abstention respectively. These votes are conducted to determinate which of the proposals advances for testing. If the Quorum is met, the proposals that advance to the next period need to meet Super-majority, which consists on having 80% of "Yay" votes when evaluated against the total of "Yay" plus "Nay" votes. This can formally be defined as:

$$SuperMajority = \begin{cases} Yes, & \text{if } \frac{V_{Yay}}{V_{Yay}+V_{Nay}} \geq 0.8 \\ No, & \text{otherwise} \end{cases} \quad (3.1)$$

The Quorum for the next vote period is adjusted taking into account the participation in the concluded one and the current Quorum requirement. As a seed, the Quorum for the first cycle was defined as 80%. The Tezos formula to update it is as follows:

$$Q_{t+1} = 0.8 * Q_t + 0.2 * q_t \quad (3.2)$$

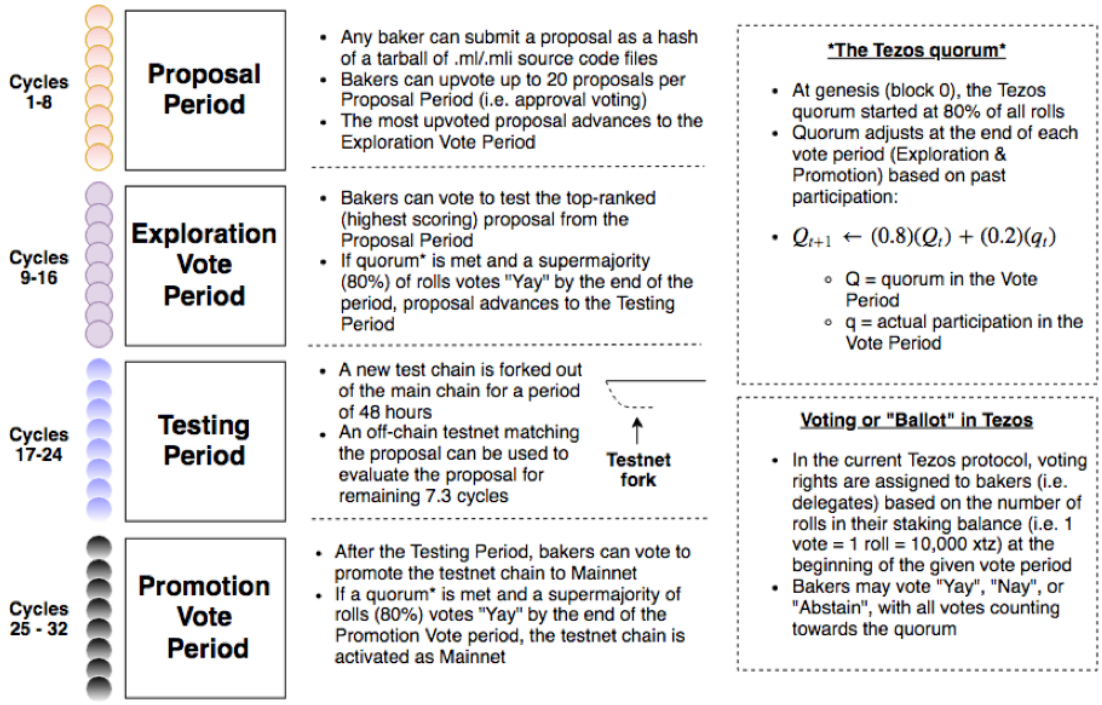


Figure 3.1: Tezos governance mechanism overview [1].

Where: Q = Quorum in the Vote Period; q = Participation in Vote Period.
With the conclusion of the Vote period, the amendment process proceeds to the third one, if all the requirements were met. This is called the Testing period and it consists in the creation of a test chain that is run for 48 hours in order to reduce the risk of implementing the changes directly on the main chain. This test chain is a sandbox version of the main one and it has no access to system calls, it serves only the purpose of evaluating whether a proposal is a worthy amendment or not. At the end of the testing period, the process moves to the Promotion Vote period where individuals vote for the adoption of the amendment. This period is very similar to the second period, Exploration Vote period, with the need for Quorom and Super Majority so that the proposals can be accepted, but with one major difference: at the end of this period, the accepted proposals are promoted to the main chain. All the changes to be promoted to the main chain need to pass formal tests, which include formal verification. Another interesting characteristic is that the stake of each individual is computed at the beginning of each period, which means that the amount of Tezzies owned at the beginning of the period determinate how much ones vote values (1 roll = 1 vote). This governance mechanism is illustrated in Figure 3.1, which was taken from the Tezos documentation [1, 97]. The volume of amendment proposals and the current and previous protocols of Tezos can be seen in [98] and [99] respectively.

As it has been introduced before, Tezos has more than just one public blockchain network. There are three different main networks running without counting the possible test chains that are generated in the amendment process. These blockchain networks are:

- Mainnet: This is the live network that is used to transact real assets. The interaction with this network costs real Tezzies.
- Alphanet: It is a test network that is mostly used to test smart-contracts and it serves the purpose of testing Tezos software before it is deployed to the Mainnet. All Tezzies are

fictitious.

- Zeronet: A test network that contains code that may not yet be tested. This network is mostly used by the core developers of the blockchain.

It is also possible to have a sandbox version of the blockchain network running on a local private network. This is very useful when it is required to have a private blockchain, similarly to our case, where only stakeholders inside the facilities (imagine robots and machinery in a factory) are able to interact with it.

Tezos' software is divided into four main parts which serve different purposes. These components are worth understanding as they give an explanation of the blockchain by themselves. The components are as follows:
Tezos Node: It is the main component of the whole system, it serves the purpose of synchronising with the whole network, by connecting the device that runs it to other peers and continuously sends and receives blocks and updates the node state.
Tezos Client: It is the library used to communicate with the node through Remote Procedure Call (RPC), which ultimately is used to deploy and interact with contracts on the Blockchain and to get information about blocks. This also contains a built-in software to manage wallets (accounts).
Tezos Admin: This component serves the purpose of having commands that allow the user to interact with the peer-to-peer layer so one can check the connections and interact with them.
Delegate: It is the component related with baking and it is divided into 3 daemons:

- Baker: computes the baking rights for a given account, collects transactions information and bakes blocks;
- Endorser: computes the endorsing rights for a given account. When receiving a new block, it verifies the validity of the block and emits an endorsement operation;
- Accuser: continuously monitors all the blocks emitted to the blockchain to detect double-baking and double-endorser operations in order to denounce them.

Tezos has two different kinds of accounts, the implicit and originates ones. The implicit ones start with the identifier tz1 (e.g. tz1iRrRdWu4ACDh4oUX2E1GeLngrhmYENE7r), and for each implicit account, there is a public and a private key associated. These type of accounts have also the owner and the balance field. The originated accounts have the identifier starting with KT1 (e.g. KT1TvQ8HhuQnckCLeqGTFfcUajh65VjJSfYh). These accounts can have Michelson code, turning them into contracts. There are four main fields in this type of account:

- Manager: the private key of the account;
- Amount: tokens in the account;
- Delegatable: indicator if the tokens can be delegated for baking;
- Delegate fields: information about who received tokens to delegate from this account.

To interact with the accounts there are wallets. Wallets are pieces of software or hardware used to manage accounts and some can provide additional features to help users manage their assets. The hardware wallets have the benefit of not being connected to the internet so, the risk of being hacked is reduced. Software wallets can be installed as plugins to the browsers or as external applications. Tezos' client has a built-in wallet that can be used with a command line interface.

It is also important to understand the concept of gas and its limits even though that in a private blockchain the concept of payment might not be used. The gas is the fee that is paid to the baker for his computational resources cost when baking the transaction. There is also the storage fee, that is the fee that one must pay when deploying or using one contract that needs to store some data in the blockchain. This fee is only paid when increasing the storage size. If 5 bytes are replaced by another 5 bytes, no fee is paid. When deploying a contract to the blockchain, one must declare the limits of both gas and storage fees. This serves to indicate the maximum that he's willing to pay to see his transaction validated. However, Tezos' software defines a hard-cap for both limits to assure that no one tries to make this value too high in order to avoid dead-locks on the network. It is worth mentioning that bakers are specified to work in their best interest, they always prefer a transaction that has lower limits because those are the transactions that take less time to validate.

3.2 RobotChain

RobotChain is a blockchain designed for robotic systems [2]. By having a blockchain as a ledger, it is ensured that every event sent to the network is validated and if inserted in the blockchain, it is valid and secure. This blockchain is based on the one developed by Tezos [26, 27] as it incorporates characteristics that are important for systems that are intended to work in sensitive environments like RobotChain. The major properties of this blockchain technology are the support for formal verification of the code and the self-amending property that allow performing changes on the blockchain by voting on-chain, without the need to conduct hard-forks when a core change needs to be executed. The formal verification of code ensures that it always does what it is supposed to do, regarding the specifications. This blockchain uses as a consensus algorithm the Delegated Proof-of-Stake, which is more energy efficient than Proof-of-Work and it also as the important property that not everyone needs to be a baker, which is important when we think on heavy energy and time-dependent environment like a production line. RobotChain also allows the creation of modules upon it, as it is a modular approach, decentralized applications and other systems can be built using the information that is on the blockchain.

RobotChain is a consortium blockchain [100], which consists of a private blockchain where the nodes are from multiple manufacturers, which is the case of factories where robots were manufactured by different companies. This type of use for blockchain also leverages from one important characteristic present in Tezos' blockchain, which is the self-amendment process and the ability to change rules, like the consensus algorithm, without the need for hard-forks, which ultimately adds a layer of security [101]. Being a consortium blockchain, it also allows the creation of a reputation system, where manufacturers gain reputation depending if the robots that communicate with RobotChain act honestly and lose reputation if they misbehave or if the robots try to cheat the network by propagating blocks to the blockchain that contain false information.

The creation of the private blockchain - RobotChain, was done by cloning the main net of Tezos and by conducting several changes to it. These changes were made in an attempt to understand inner workings and improve it in the context of a consortium blockchain. The main changes were the addition of a transaction parameter named "Transaction Description", providing a field to document the transaction or a smart-contract call, allowing texts on transactions. The

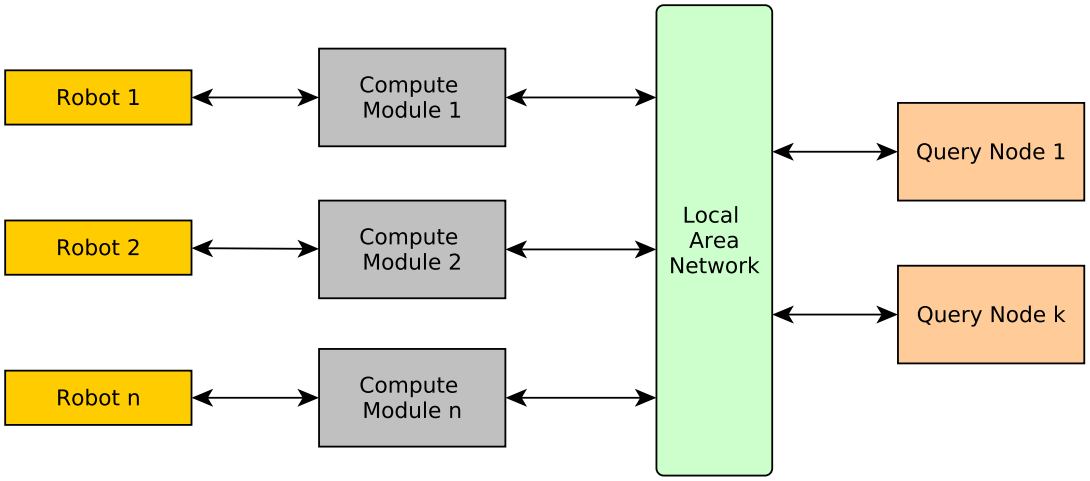


Figure 3.2: Description of how the RobotChain works [2].

second change was the parameters of smart-contracts storage, which were hard-coded to have a small limit and were increased in order to store more information on the smart-contracts. The third modification made to the network was the change of the genesis public key, allowing the creation of a private network in a controlled environment. The fourth modification was the removal of the token, since it is not required in robotic environments, removing a time-consuming step from the validation process. The final and biggest modification was the property of time-segmenting the blockchain [102]. This property consists of segmenting the blockchain every n blocks and generating a new blockchain in which the first block contains all the smart-contracts that were inserted before the segmentation. The information that is contained in the segmented part is then inserted into cold storage. This property allows RobotChain to scale both in terms of speed and space since it has less information to be stored on the devices (cold storage can be stored elsewhere) and since it requires less space, it can be run on smaller devices that focus entirely on validating incoming transactions.

In this work we used and show how a blockchain - RobotChain, can be used to store robotic events, such as logs, and how the smart-contracts can be used to process this information contained in the RobotChain and how can they interact with Oracles and robots. A high-level representation of how RobotChain and the communications work is presented in Figure 3.2, where a blockchain is running over the Local Area Network is shown and Robots are coupled to computing devices that insert values into the blockchain. These computing modules can also receive information from the blockchain, either by querying it or by receiving it directly from smart-contracts. The same idea is also applied for Oracles (external parties), which can query the blockchain to receive information and can insert analytics into smart-contracts if there are any that allow it.

3.3 Smart-Contracts

Smart-contracts are software programs that bind two or more parties involved in a transaction to an agreement without any third party being involved or without requiring intervention to trigger the contract. The smart-contracts on the blockchain guarantees that nothing can be

changed about the agreement without the consent of all the parties because blockchain provides immutability and because smart-contracts automatically trigger when the conditions defined on it are met. In short, smart-contracts are protocols intended to facilitate audits and enforce the agreement stipulated on the contract. Smart-contracts allow credible transactions without third parties that facilitate the exchange of money, property, shares, or anything of value in a transparent, conflict-free way while avoiding the services of a middleman. This technology was first envisioned by Nick Szabo [22], who realized that traditional contracts could be converted to computer code, stored and replicated in digital systems. Later, Ethereum [21] implemented this technology over the Ethereum Blockchain and with it, gained immense exposure and the value of the tokens in Ethereum rapidly increase. This blockchain marked the beginning of the second generation of blockchain, which focused on applications, allowing developers to build decentralized applications over the blockchain network. These decentralized applications are created by using smart-contracts that handle transactions, which are usually payments, and that also allows the interaction with Oracles (external parties to the blockchain) that define the logic of the application. We can refer to this as using blockchain as a ledger, to register the transactions, smart-contracts to handle transactions and Oracles that have the application logic (which can be anywhere and centralized, since it is external to the Blockchain). Smart-contracts are created differently for different blockchains. The official programming language to write smart-contracts in Ethereum is called Solidity [103]. Solidity is compiled to EVM byte code, which is run in the Turing-complete Ethereum virtual machine. Tezos has a similar approach in terms of building smart-contracts, having a Tezos Virtual Machine that directly interprets smart-contracts. As mentioned before, the official language of Tezos for building smart-contracts is Michelson [93], which is a strongly typed, stack-based language that facilitates the formal verification and reduces the developer induced errors by performing a type checking procedure before compilation. But due to the fact that Michelson has a steep learning curve, which is a consequence of being a stack-based language, Liquidity was built [104]. Liquidity is a high-level typed smart-contract language based on OCaml. Not only Liquidity is easier to learn when compared to Michelson because it is a high-level language, but it also has good documentation and provides both a compiler and a decompiler to compile smart-contracts from Liquidity to Michelson and vice-versa. This makes Liquidity a perfect choice to both build smart-contracts in Tezos and to audit them (by decompiling Michelson to Liquidity).

Despite the fact that smart-contracts are usually used to do legal contracts or monetary transactions, they can be useful to allow the interaction of Oracles with the blockchain, automatically performing actions and to help nodes of a network obtain global information. One of the focus of this project is to show that smart-contracts can be used in applications outside the financial scope, with focus on integrating robotics with blockchain by using smart-contracts to store information and to serve as a communication gate between the blockchain itself, robots and AI algorithms. In this project, we focused on building smart-contracts in Liquidity for the reasons mentioned before.

In listing 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 1.0]`. 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 is 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 is required that the structure has the name 'storage', for example:

Listing 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
6: Auxiliary methods
```

`type storage = oracles`. The fourth step is optional, even though it serves as an initializer for the storage. This method is defined by the `init` keyword and it is called only one time which is 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 fifth requirement in a typical Liquidity structure are the entry points. These points are methods that 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. Entry points are defined with the `entry` keyword and return a list of operations and the storage. 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 )`. The final requirement is the optional step of creating auxiliary methods. These methods, if created, serve to aid different operations. An example is a method that receives a list of integers and calculates the sum of the list. This method can be created in two different ways: with the same syntax of OCaml functions: `let find_sum lst = let sum = List.fold (fun (elt, acc) -> elt + acc) lst 0 in sum`, or by using the `[@inline]` keyword in the method signature: `let[@inline] add_zero_list lst = 0::lst`. Note that there are more options that can be used, but in the context of integration between Robotics and blockchain, the presented steps are enough to perform rather complex tasks with smart-contracts.

An example of a smart-contract written in Liquidity is shown in listing 2 and its correspondence in Michelson in listing 3. These smart-contracts were among the ones created for this project and these two were used throughout the entire work to store all the logs of the robots. The idea was to use the storage of a smart-contract to store all the events of a robot so that they were safely stored, immutable and easily audited. The smart-contract written in Liquidity goes accordingly with the structure explained. First, it defines the `version` of liquidity being used, then, the storage, where it is defined a `big_map` to store the robot logs and the `address` of the robot that is allowed to use that smart-contract. The `init` method initializes the storage with an empty `big_map` and with the `address` of the robot. Finally, the `main` method defines the only `entry` method on this smart-contract and receives both a string with the robot log and timestamp of that log. In this, it is verified if the address that is calling the contract is the allowed robot and if the log inserted does not exceed a predefined length. If any of these conditions is not verified, the contract does not perform any action. If they are verified, the contract couples the log to the storage and concludes the work. The corresponding Michelson code does the same logic with a completely different syntax and the difference that is required to define the parameter and the storage before the method that will handle all the logic.

Listing 2 Liquidity smart-contract to store robot logs.

```
[%%version 1.0]

type storage = {
  information: (int,string) big_map;
  identifier: address;
}

(* Deployment call has the  address of the robot to be used *)
let%init storage (id : address) =
  { information = (BigMap : (int, string) big_map);
  identifier = id  }

(* Function called to insert values into the storage *)
let%entry main ( (timestamp : int), (param : string) ) storage =
  (* Check if caller is authorized *)
  if Current.source() <> storage.identifier then
    Current.failwith ("Identifier not compatible.");

  (* Check if param length matches the constraints *)
  if String.length param > 256p then
    Current.failwith ("Parameter too long.", param);

  let storage = storage.information <- Map.add timestamp param
  storage.information
  in
  ( ([]: operation list), storage )
```

To build and deploy smart-contracts into a Tezos network, either public or private, it is required to have the Michelson code verified. The first step is to install Liquidity (if it is the chosen language) by doing the following steps:

1. Install OPAM: `sudo apt-get install opam`
2. Create an OPAM switch: `opam switch create liquidity 4.06.1`
3. Clone Liquidity repository: `git clone https://github.com/OCamlPro/liquidity && cd liquidity`
4. Make files: `make clone-tezos`
5. Install dependencies: `make build-deps`
6. Build and install: `make && make install`

The second step is to compile a Liquidity file to Michelson. To develop a Liquidity smart-contract any text editor works, but Liquidity-lang [104] offers an online text editor that offers an appealing color scheme to differentiate the reserved words and warnings in writing time. This text editor also offers an easy way to compile Liquidity to Michelson and vice-versa, but it can be done using the Liquidity compiler by doing as follows in the terminal where it was installed:

- Compiling Liquidity to Michelson: $ liquidity filename.liq
- Decompiling Michelson to Liquidity: $ liquidity filename.tz

Listing 3 Michelson smart-contract to store robot logs.

```
parameter (pair int string);
storage (pair :storage (big_map int string) address);
code { DUP ;
       DIP { CDR @storage_slash_1 } ;
       CAR @_timestamp_param_slash_2 ;
       DUP ;
       CDR @param ;
       DUUUP @storage ;
       CDR %identifier ;
       SOURCE ;
       COMPARE ;
       NEQ ;
       IF { PUSH string "Identifier not compatible." ; FAILWITH } { UNIT } ;
       DROP ;
       PUSH nat 256 ;
       DUUP @param ;
       SIZE ;
       COMPARE ;
       GT ;
       IF { DUP @param ; PUSH string "Parameter too long." ; PAIR ; FAILWITH }
          { UNIT } ;
       DROP ;
       DUUUP @storage ;
       CDR %identifier ;
       DUUUUP @storage ;
       CAR %information ;
       DUUUP @param ;
       DUUUUUP ;
       CAR @timestamp ;
       DIP { SOME } ;
       DIIIIP { DROP ; DROP ; DROP } ;
       UPDATE ;
       PAIR @storage %information %identifier ;
       NIL operation ;
       PAIR };
```

The above commands generate a file, either with the extension `.tz` or `.liq`, which represents a Michelson and a Liquidity smart-contract respectively. After generating the Michelson code, it is required to type-check it in order to search for possible insecurities that may be present due to wrong variable types or excessive memory in a variable. This can be done by executing the following command: `tezos-client typecheck script filename.tz`. Finally, to push the contract to the blockchain, it is required to execute `tezos-client originate contract [CONTRACT_NAME] for [IDENTITY] transferring [STARTING_BALANCE] from [SOURCE_OF_FUNDS] running [PATH_TO_PROGRAM]` with the correspondent name for the contract, the identity of the contract creator, the balance of Tezzies that will be sent to the contract, the address of the sender of such Tezzies and the smart-contract path.

3.4 Conclusion

In this chapter, we described how Tezos differentiates itself by providing properties that are unusual in blockchains, but that are important to have increased data security. In short, these

properties ensure that there is a lower energy consumption and faster consensus algorithm (Delegated Proof-of-Stake), when compared to traditional ones, e.g., Proof-of-Work, that the blockchain contains self-amending properties that allow changes to be conducted in the blockchain core without the need for hard-forks and the support to formally verify the smart-contracts. We have also shown how these properties are present in RobotChain and which changes were conducted on this private blockchain to have features that are useful in robotic environments. Lastly, we explained how smart-contracts emerged and how can they be used in RobotChain, by presenting an example of a smart-contract to register robotic logs.

Chapter 4

The Cobot UR3

Robots, specially Cobots, have grown exponentially in terms of their use due to the Industry 4.0 paradigm because properties such as sensors to stop movement upon collision are essential in factories and in clustered environments. This growth made Cobots reach a stage of maturity that makes them essential in factories, both to increase productivity and to work in collaboration with Humans. The main purpose of Cobots is to work closely with someone to produce or create something. There are different types of Cobots, but almost all of them need to be either supervised or caged to ensure that their actions will not hurt anyone or itself. Even though that any robot can be considered collaborative and used as so, Cobots have special features that differentiate them. These features are: very strict safety measures, force limited manipulators that constrain the robot to certain limits and these are specially designed to be able to interact with human co-workers. These robots usually have a rounded design, which spreads the force of an impact over a larger area and normally contain pressure sensors that detect slight collisions in order to stop the robot before damaging anything. The problem of doing passive recognition of collisions is that off-the-shelf Cobot solutions can't actively avoid collisions without external hardware or software, meaning that the robot will stop its movement when it enters in contact with an object, instead of recalculating a new path that avoids collisions.

Cobots are a welcomed improvement regarding the traditional robots that served the purpose of automation. It is expected that this new era of robots will eventually be used in all kinds of environments, such as streets, houses and public places where they will interact with people. However, this is still a remote scenario, since it is required to have certainty that robots actuate within certain ethical and legal conditions and that they can not only be fully autonomous but have a dynamic autonomy that allows them to react differently to similar situations.

In this project, we worked with a real UR3 robotic arm [105]. The UR3 robot is produced by Universal Robots and it has an onboard controller. This robot can be controlled manually by a user through an interface or it can run autonomously if programmed to do so. It can be programmed to move a tool, and communicate with other machines using electrical signals. It is an arm composed of extruded aluminum tubes and joints. UR3 comes with a graphical interface called PolyScope, that allows the robot to be programmed to move the tool along the desired trajectory. The main UR3 specifications are:

- Weight: 11 kg;
- Payload: 3 kg;
- Reach: 50 cm;
- Degrees of Freedom: 6;
- Rotation: 360-degree rotation on all wrist joints, infinite rotation on end joint; 360 degrees in 1 second;

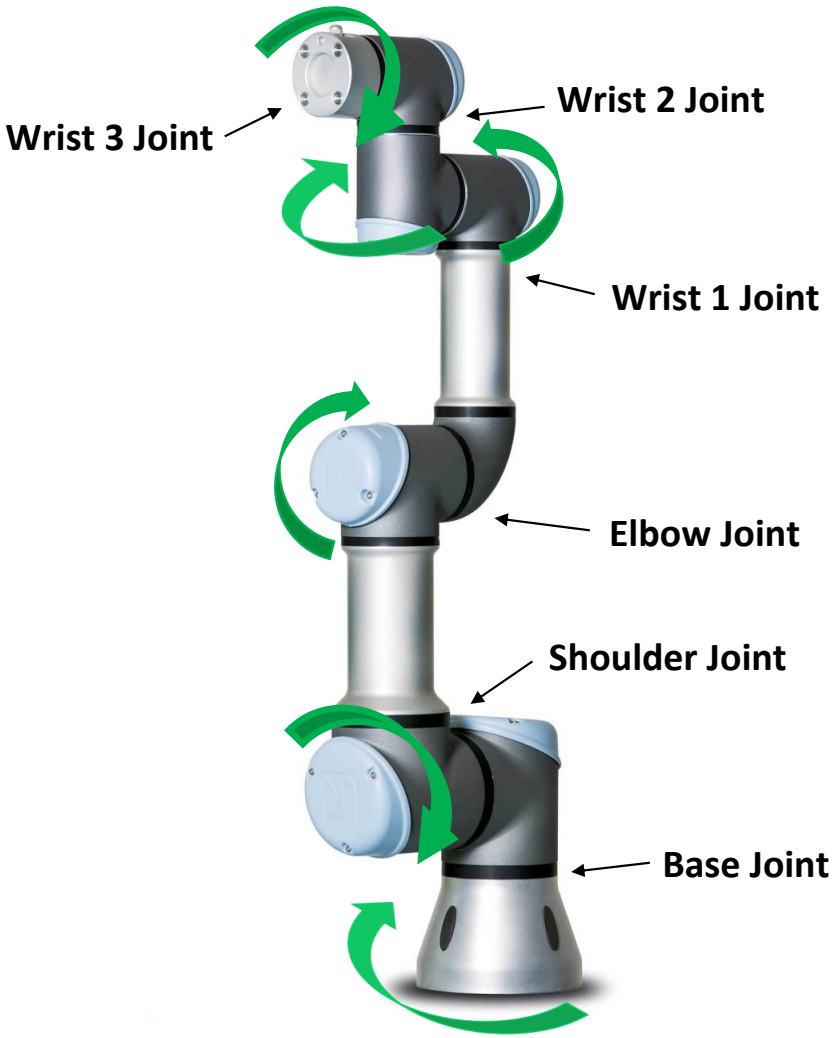


Figure 4.1: UR3 robotic arm, joint rotations and its descriptions.

- Repeatability: ±0.1 mm;
- Safety: 15 adjustable settings (force limit default is 150 N; can be adjusted down to 50 N).

In Figure 4.1, we show how the UR3 joints rotate and where they are located. Throughout the rest of this document, we will refer to the joints from 1 to 6, starting from the base joint to the wrist 3 joint. We do this because it is easier to explain the joints this way and because this nomenclature is usually found in the literature. From the robot joints, we extract information about the position of the joint, how much effort is being done and the velocity of the joints. As the UR3 does not come with a gripper, we used the RG2 gripper [106]. This gripper is easy to integrate, as it is only required to have a serial connection between the robot and the gripper and because Universal Robots also have embedded functions to control the gripper inside PolyScope. The UR3 robot, as briefly explained, can run on manual control, where a worker moves the robot arm or in automated control. For this, the programming can be done at two levels, the Script Level and the API Level. In the case of Script Level programming, the arm is only controlled by a program written in URScript. This is a language that was developed by Universal Robots and it

is very similar to python. URScript can provide the necessary commands for communication and motion control, but it has not extended libraries for motion planning and the communication between a server (a computer) and the client (the robot) using URScript limits the functionality of the robot, as the pendant does not work during the execution of scripts. This makes it impossible to have dynamic changes in useful time. If such computations are necessary, it is appropriate to write a program in a higher level, the API Level. There are some implementations for this, the most common one is the integration with the Robot Operating System (ROS) and then use `ur_modern_driver` [3] to allow MoveIt! [107] to control the robot. Even though the UR3 arm can be controlled using PolyScope, it is not enough to solve the problems we had in this project. Shortly, PolyScope didn't allow us to control the robot dynamically, to change its speed and movements in execution time. So, in the next sections, we will explain how we configured UR3 to work with ROS and the scripts created to both receive values from the robot and send commands to it.

4.1 Configuration

The first step to configure a computer and the UR3 to communicate is to install ROS. In this project, we worked with ROS-Kinetic. To install it, it is only required to have an operative system that is compatible, usually, either Ubuntu or Debian. Using Ubuntu, it is necessary to follow these steps:

1. Setup the computer to accept software from packages.ros.org: `$ sudo sh -c 'echo "deb http://packages.ros.org/ros/ubuntu $ (lsb_release -sc) main" > /etc/apt/sources.list.d/ros-latest.list';`
2. Setup the keys: `$ sudo apt-key adv --keyserver hkp://ha.pool.sks-keyservers.net :80 --recv-key 421C365BD9FF1F717815A3895523BAEEB01FA116;`
3. Update packages: `$ sudo apt-get update;`
4. Install ROS: `$ sudo apt-get install ros-kinetic-desktop-full;`
5. Initialize rosdep and update it: `$ sudo rosdep init && rosdep update.`

After ROS is successfully installed, it is only required to install the package `ur_moder_driver` by cloning it into a catkin workspace and the package `universal_robots` by executing the following command: `$sudo apt-get install ros-kinetic-universal-robots`. With this, we have everything we need to control the robot from the computer, now it is just needed to configure the UR3 hardware to enable networking. To do so, it is possible to use UR's teach-pendant. It can be done by using it to navigate to the Setup Robot -> Setup Network Menu and select DHCP and an IP for the robot. This can be seen in Figure 4.2. With this configuration completed, the robot can be started from the computer by executing `$roslaunch ur_bringup ur3_bringup.launch robot_ip:=192.168.1.102`.

4.2 Communication

Universal Robots directly communicate with the tech-pendant that runs PolyScope. But it is possible to communicate with external devices through the TCP/IP protocol. The UR3 arm has

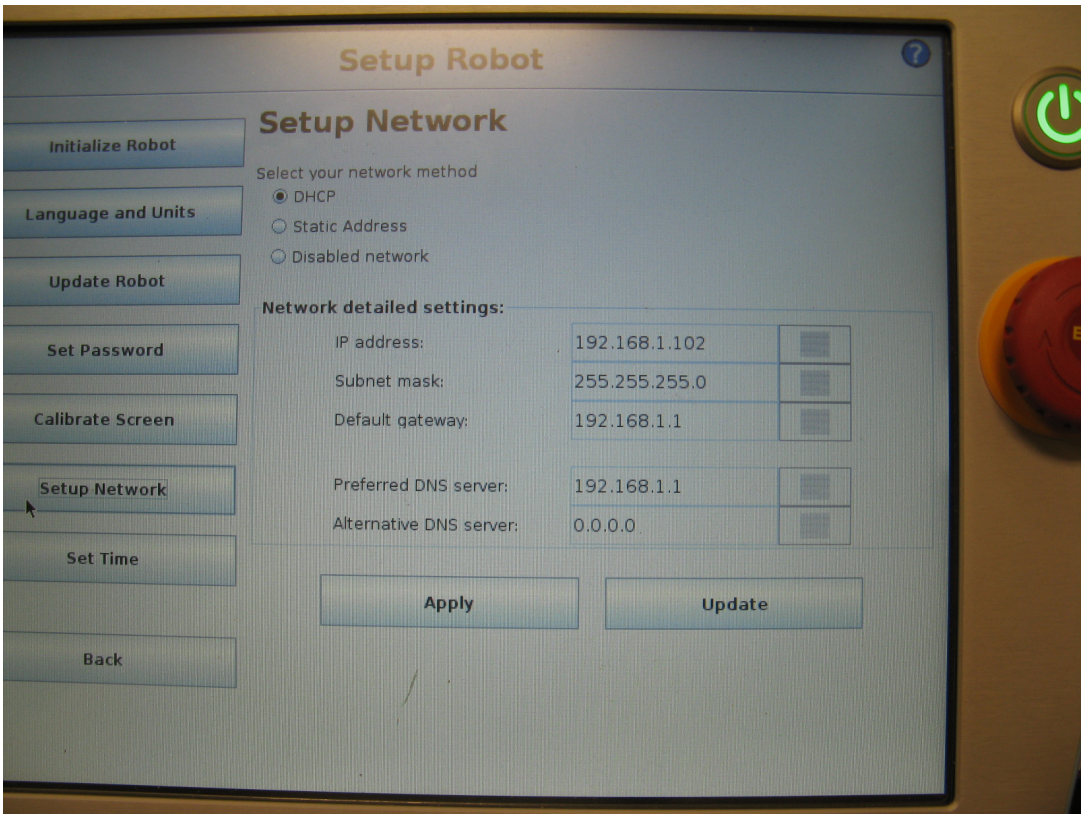


Figure 4.2: Configuration of the UR3 access to network [3].

two specific ports for this communication. The port 30003 is dedicated to sending information about the robot (such as effort, position, and velocity of the joints) and the port 30004 is used to receive targets (commands or changes to the behavior). When connected, the UR3 computer broadcasts a stream of data that contains internal values of the robot. It is also possible to use ROS `ur_modern_driver` to receive these values in the form of a ROS topic, which is streamed at a rate of $125Hz$. The structure of the streamed data is presented in Table 4.1. In the scope of this project, the interest is only in "Q actual", since it is the part of the message that contains the joint values.

To send commands to the robot we discarded the option of using ROS `ur_modern_driver` required to either use MoveIt! or to create specific ROS messages and publish them with a frequency of $125Hz$, which was not ideal for us. So, we used the TCP/IP protocol to communicate over the port 30002 with sockets. This communication requires that the commands that are sent to the robot are formatted with URScript syntax. The commands should be transformed into strings before going to the robot. An example of a command to tell the robot to move to a specific target is:

- "movej(get_inverse_kin(p[x, y, z, rx,ry,rz], qnear=[q1, q2, q3, q4, q5, q5]), t = speed".

Where (x, y, z) represent a target point; (rx, ry, rz) represent the orientation of the end-effector, $qnear$ represents a set of points used in the calculation of the inverse kinematics and $speed$ is the value in seconds that the robot should take to perform that movement.

The $movej$ command is sufficient to make the robot perform any desired movement that is physically possible to the robot. The only problem is that there is no way to control the RG2

Table 4.1: UR3 data stream structure [4].

Meaning	Type	Number of Values	Size in Bytes
Message Size	Integer	1	4
Time	double	1	8
Q target	double	6	48
Qd target	double	6	48
Qdd target	double	6	48
I target	double	6	48
M target	double	6	48
Q actual	double	6	48
Qd actual	double	6	48
I actual	double	6	48
I control	double	6	48
Tool vector actual	double	6	48
TCP speed actual	double	6	48
TCP force	double	6	48
Tool vector target	double	6	48
TCP speed target	double	6	48
Digital input bits	double	1	8
Motor temperatures	double	6	48
Controller timer	double	1	8
Test value	double	1	8
Robot mode	double	1	8
Joint modes	double	6	48
Safety mode	double	1	8
	double	6	48
Tool accelerometer values	double	3	24
	double	6	48
Speed scaling	double	1	8
Linear momentum norm	double	1	8
	double	1	8
	double	1	8
V main	double	1	8
V robot	double	1	8
I robot	double	1	8
V actual	double	6	48
TOTAL		131	1044

gripper through TCP/IP communication. In other others, there is no solution to control the gripper from external devices. To solve this problem, we created a URScript in the PolyScope software where we only defined two commands, one to open the gripper and one to close it. Then, we extracted the URScript from the tech-pendant to a computer and removed everything that was not necessary and split the commands into two separate files, one file containing the definition of the gripper and the commands to close it and the other one containing the definition of the gripper and the commands to open it. Then, whenever it was required to either open or close the gripper, we read the file to a string and sent the whole string through the socket communication. With this, we had a complete system that allowed us to control the robot and read his inner values from an external device.

4.3 Conclusion

In this chapter, we briefly introduced Cobots and its usefulness in factories and to do automation. Cobots are deeply connected with the Industry 4.0 paradigm, as they are used to automating processes that were usually done manually by Human workers. These type of robots can also work closely together with human workers due to features, such as pressure sensors, that not only increase the robot security, but also give tools to workers to manipulate the robots with ease. We also described how a UR3 arm can be configured to accept commands from a computer and how we built a method to receive and send data to real UR3 arm.

Chapter 5

Detecting Robotic Anomalies using RobotChain

Robotic events can provide notable amounts of information regarding a robot's status, which can be extrapolated to detect productivity, anomalies, malfunctions and used for monitorization. But, registering robotic events with proof that they can't be tampered with is often disregarded because of the complexity of such systems, which leads to slow adoption. Thus, insecurity breaches and file tampering can be done to protect robot manufacturers when their robots have abnormal behavior and jeopardize a factory production line. Which leads to discarded robotic logs because they are susceptible to chances and malicious intents. We tackle this problem by inserting robotic logs into RobotChain by using smart-contracts. This enforces that only an authorized entity can use that smart-contract (the robot itself) and that every log registered in RobotChain is immutable and safely stored. We also propose a method to detect robotic anomalies that use the information registered by the robots. Detection of anomalies is an important capability to have as it can increase the life-span of robots but can also increase productivity in factories, as it can provide early information about anomalies that need to be solved, and protect human operators that work close to robots that can be damaged and act improperly. In short, in this chapter, we propose a method to solve the problem of registering robotic events with proof that they can't be tampered with and we propose a method that uses those robotic events to detect anomalies. This method works by having an Oracle interacting with the smart-contracts that use the information that is inserted into the blockchain to detect anomalies. This proposal is useful as it provides a ledger that can register information from the robots in a way that it cannot be tampered with and a way of leveraging Artificial Intelligence methods over it, as it can have new modules integrated into the system without the need to redesign it. Our proposal integrates three technologies that are disrupting our daily life's in a unique system, Robotics, AI, and Blockchain. We show how this proposal can be used to store information about a UR3 arm and detect anomalies on it.

The remainder of this chapter is structured as follows: section 5.1 presents and explains the proposed methods to register robotic events and to detect robotic anomalies. Section 5.2 shows the experiences conducted to validate the method and discusses the results obtained. Section 5.3 describes how we built a web application to read values from the blockchain and present them online, and lastly, section 5.4 contains our conclusions.

5.1 Proposed Method

To register robotic events we used the storage of a simple smart-contract, which is called by the computer coupled to a robot that adds to the storage the new information about the robot joints. The smart-contract used for this was written in Liquidity, which is a high-level language that is compiled to Michelson, the Tezos main smart-contract language. As explained in 3.3, Liquidity is useful as it is statically-typed, which reduces the number of induced errors by the programmer and as it has a compiler to Michelson and a decompiler that translates Michelson to

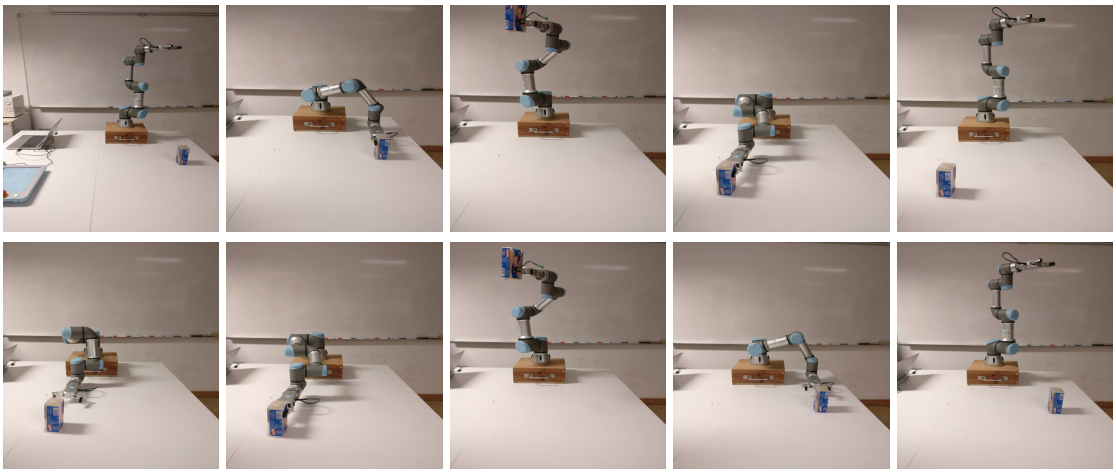
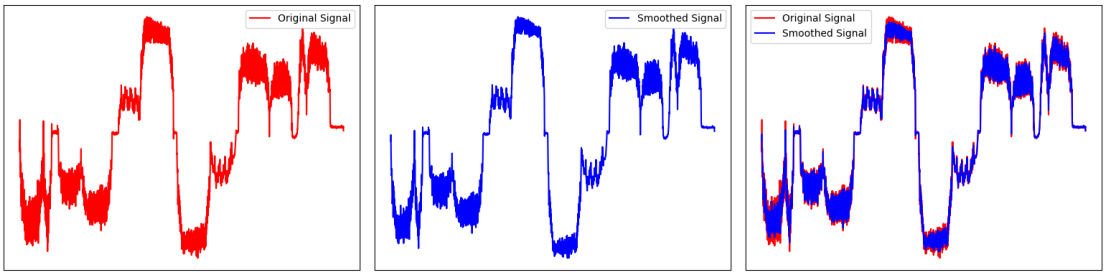


Figure 5.1: Pick and place task. Robot start in the home position picks the object of position 1, leaves it on position 2, returns to the home position and then does the inverse, picking the object of position 2, dropping it on position 1 and returning to home position.

Liquidity, it becomes easier to audit smart-contracts. The smart-contract used to store robotic logs was explained in detail in 3.3. But shortly, the smart-contract contains two variables in the storage: an address, to register the robot identifier, and a string list, to register the logs. In terms of methods, it has an init method to initialize the storage with the robot address and an empty list and the main method. The main method is where all the work of this smart-contract happens and this method receives a string and a timestamp. Initially, the logic of the method verifies if the one who called the contract was the owner of it (by comparing addresses) and if the string received does not exceed a predefined length. If all the conditions hold true, the method finishes by coupling the new string to the storage.

Using RobotChain and a real UR3 robot [105], we created a scenario where it is possible to detect anomalies. The scenario created was a pick and place task performed by the UR3, where the robot has to pick the soap, place it in a different position, return to the home position and then do the reverse by putting the soap in the initial place and returning to the home position. The main reason to illustrate this method on such a task is due to the fact that this type of task is one of the most common in factory environments, where robots are used to transport materials from one place to another, e.g. pick a door from a conveyor and place it on a car. The chain of events of the described task is presented in Fig. 5.1, where the images represent parts of the movement the robot does. We connected the robot to the ROS and with the `ur_modern_driver` and the `universal_driver` packages which are compatible with ROS, we were able to receive information at a rate of $125Hz$ regarding effort, velocity, and position of the 6 joints the robot has.

With this system, we acquired information about the robot for 25 complete sequences tasks. In the last one, we induced anomalies by holding the robot for brief seconds, counteracting its movement, enforcing the need to adjust the robot joints to correct its behavior. This forced anomaly is used to simulate when a robot suffers from some internal failure, whether by faulty components or by power issues or by colliding with another object.



(a) Original Signal (b) Smoothed Signal (c) Comparison

Figure 5.2: Smoothing process and comparison. Image (a) represents the original signal that contains noise, (b) is the same signal but after being smoothed with the proposed method and (c) is the comparison between the original and smoothed signals.

The acquired information consists of multiple signals that contain noise, as it is acquired using a real robot, so, to remove part of the noise, we smoothed every data point with the following filter:

$$g(i) = \begin{cases} 1/3 \sum_{k=i-1}^{i+1} f(k), & i \in \{2, \ldots, n-1\} \\ f(i), & \text{otherwise} \end{cases} \quad (5.1)$$

Where $i = 1, ..., n$, n is the number of samples on each signal, $f(i)$ represents sample i from one of the collected signals and $g(i)$ its smoothed version.

As the dataset contains multiple repetitions of the same task, we used Autocorrelation to find the period of each signal in order to create a model of it. We then divided the dataset into two separate ones, the initial 80% of the data for the train set and the rest for the validation set. Using the train set, we created a model for each signal, where the model is the mean of each point for the different periods. This ensures that we have a representation of the signal that is robust and at the same time it is a low computation cost approach. For each point, we also determined the standard deviation. In the end, the model to tackle the anomaly detection consisted of a new signal for the type of signals contained in the dataset and a vector with length equal to the period size that contains the values of the standard deviation for each point of a period. The creation of this model is an unsupervised task, as we don't label the data. In Figure 5.2 we present three images to showcase the smoothing process. The first image (a), shows the original signal, the second one (b) shows the signal after being smoothed by the aforementioned process, and finally, (c), shows a comparison of both signals to show that the smoothed signal has less noise.

To detect anomalies, we compare a period with the model signals, and we use the following equation for anomaly detection:

$$\mathbf{A}_{j,i} = \begin{cases} 1, & \text{if } |\mathbf{S}_{j,i} - \boldsymbol{\mu}_i| > k\boldsymbol{\sigma}_i \\ 0, & \text{otherwise} \end{cases} \quad (5.2)$$

Where j represents the columns (different signals), i represents the indexes (rows), $\mathbf{A}$ is a variable that stores the existence of anomalies, $\mathbf{S}$ contains the values of the signal that is being checked for anomalies, $\boldsymbol{\mu}$ contains the model signals, k is a constant and $\boldsymbol{\sigma}$ is a vector that contains the standard deviations for each point.

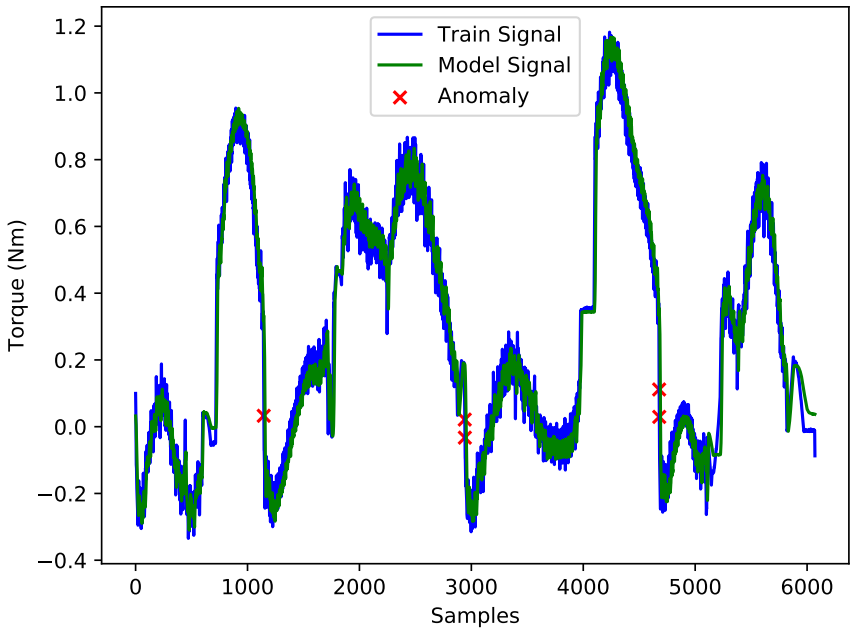


Figure 5.3: False-positive anomaly detection example by comparing the effort signal of joint 4 with the model signal.

Equation (5.2) considers as anomaly each point that deviates from the mean k times. This constant is useful to fine-tune the model to reduce the false-positives detected, as a higher k defines that points are considered anomalous only if they deviate from the mean with higher values than with a smaller k. In the following section, we explain in detail the experiences conducted to adjust the k and to validate the method.

5.2 Experiments and Discussion

By using the equation (5.2) with the train set and by validating it with the validation set, we concluded that $k = 16$ was the best value to detect the anomalies and at the same time assure the smallest number of false positives as possible, which was done by evaluating the differences between the different signals and experimenting different k values. By using the train set we also perceived that some points on the model have a standard deviation close to zero, which implies that a small variation on a new signal in that point has a high chance of being detected as anomalous. By focusing on the effort signals of the joints, we can see an example of this problem occurring in Figure 5.3. To solve this problem, we defined that an anomaly only occurs if there are more than 5 detections in a range of 30 points. This is defined as so because of the rate at which the robot publishes the information, $125Hz$, so an anomaly appears in multiple sequential points, as the robot adjusts to compensate for the occurrence.

Then, after adjusting the model, we used the validation set to verify its efficiency. The validation set contains 20% of the initial data, which translates into 5 periods of the signal, and as the data acquired represents a time series data, none of the points present in the validation set were used to create the model. In this set, the first periods contains no anomalies, and the

Table 5.1: MSE of the different periods contained in the validation set with the model signal. Only the effort for each joint signal is considered.

Period	Joint 1	Joint 2	Joint 3	Joint 4	Joint 5	Joint 6
1	0.0018	0.0037	0.0039	0.0013	0.0008	**0.0041**
2	0.0023	0.0048	0.0045	0.0017	0.0013	0.0021
3	0.0022	0.0048	0.0043	0.0019	0.0014	0.0018
4	0.0026	0.0061	0.0053	0.0023	0.0013	0.0032
5	**0.0181**	**0.2626**	**0.1470**	**0.0420**	**0.0036**	0.0039

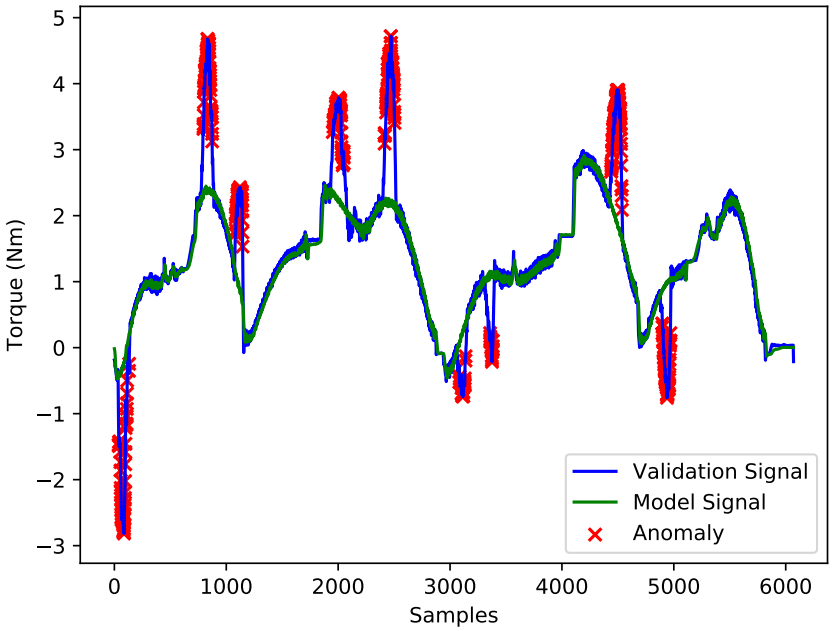


Figure 5.4: Detection of anomalies in joint 2 of the validation set where anomalies are present.

last one contains anomalies that were induced as mentioned before. This can be seen in Table 5.1, where each one of the values represents the MSE of the signal model against the signals in the validation set. In this, we can see that the MSE is higher in the last period, where the anomalies are located, except for the last joint, which has similar MSE for all periods because this joint is the closest one to the gripper and does not suffer from the anomalies induced. By using equation 5.2 with $k = 16$, the method was capable of detecting every anomaly present in the validation set and had 0 false-positive, which means that the method only detected anomalies in the last period of the validation set, which is the one containing anomalies. The detection of the anomalies is shown in Figure 5.4 for the joint 2.

To ensure that the method proposed is suitable for the problem, we acquired two new datasets with the system aforementioned and used it to test the model created for detecting anomalies. The first test dataset contains no anomalies, while the second one contains anomalies in every period. We used cross-correlation to ensure that when comparing the signals, we are comparing

Table 5.2: MSE for the different period of the two test datasets regarding the effort signal. The dataset A represents the test set with no anomalies and the dataset B represents the test set with anomalies.

Dataset	Period	Joint 1	Joint 2	Joint 3	Joint 4	Joint 5	Joint 6
A	**1**	0.0011	0.0020	0.0024	0.0013	0.0017	0.0011
	2	0.0021	0.0030	**0.0037**	0.0017	0.0019	0.0023
	3	**0.0040**	**0.0090**	0.0100	**0.0047**	**0.0030**	**0.0071**
B	**1**	0.0145	0.4037	0.1203	0.0034	0.0022	**0.0043**
	2	0.0121	0.5355	0.1590	0.0044	0.0020	0.0021
	3	**0.0236**	**0.6131**	**0.2180**	**0.0618**	**0.0056**	0.0042

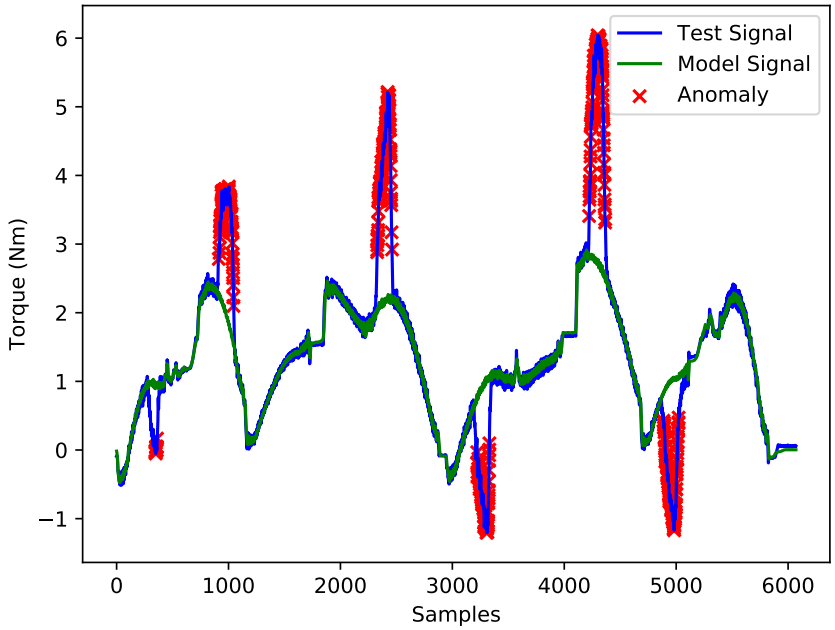


Figure 5.5: Detection of anomalies in effort signal of joint 2 in period 1 of the test dataset that contains anomalies.

the right points. So, with the correlation of both signals, we know how many data to remove at the beginning of the new datasets so that every signal starts at the same point as the signal models. In table 5.2 we present the values of the MSE per period per test dataset, A represents the test set with no anomalies and the dataset B represents the test set with anomalies. The MSE value is useful to check if there are anomalies in our experiments, but it is not a robust method and for other anomalies, since it can dilute the anomalies becoming impossible to perceive if there are any. Using equation 5.2 in both these datasets, no anomalies were found in the first one and in the second one the method was capable of detecting the anomalies in all off the periods. An illustrative example of this detection can be seen in Figure 5.5.

We conducted one more experience in which the goal was to have the UR3 conduct the pick and place task only one time. While the UR3 was performing this task, we counteracted his movement four different times. The counteraction was performed in the middle joints and in

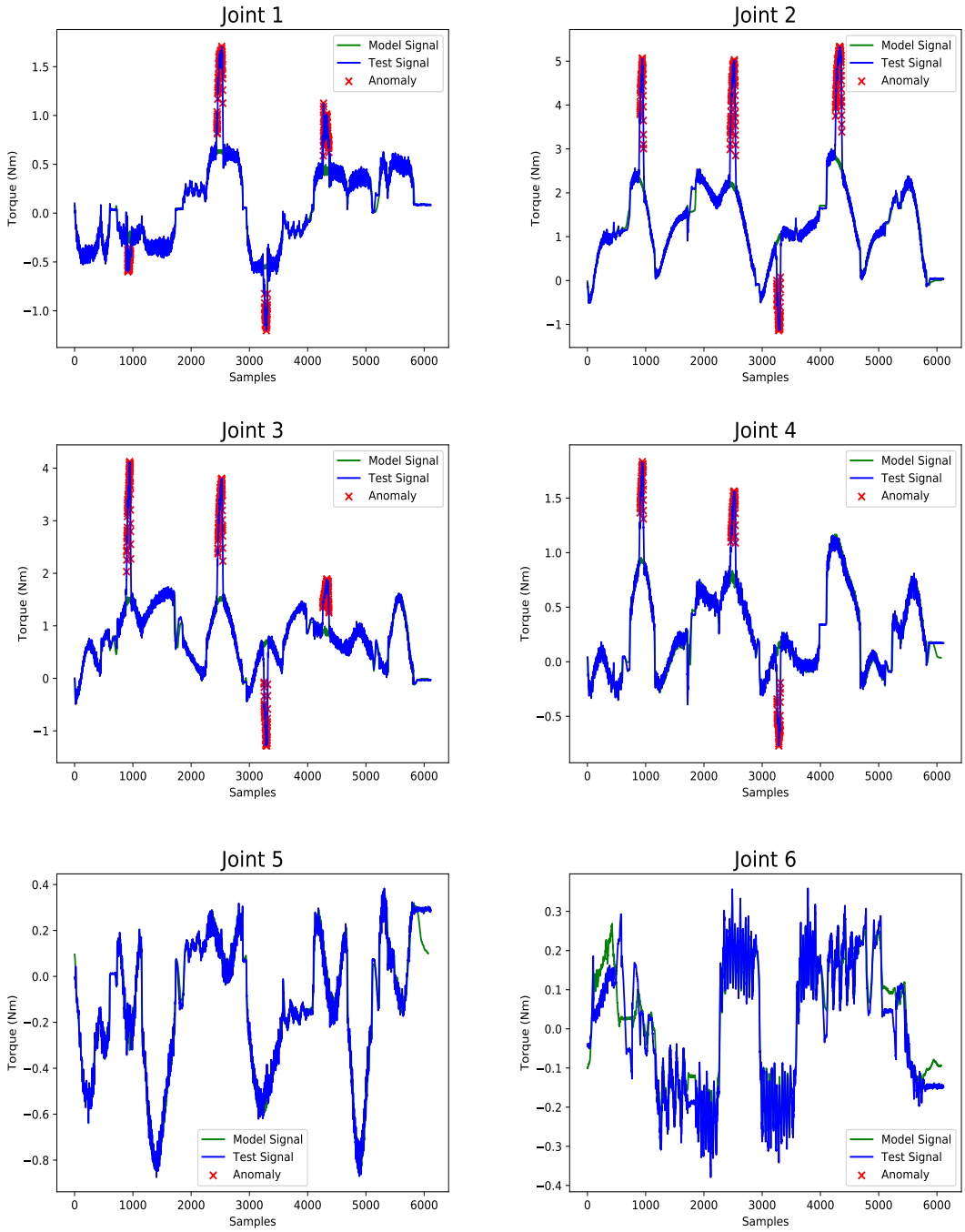


Figure 5.6: Anomaly detection for each joint of the UR3 arm regarding the effort. There were 4 anomalies induced in this experiment by counteracting the robot movement.

the joints closer to the base of the robot, which leads to an increase of effort in the joints. In Figure 5.6 we show the resultant line charts for all joints of the robot. In this, we can see that the anomalies are not shown in joint 5 and 6 because these two do not require any rectification on the effort of the movement because, in most situations, these two joints only move to aid the arm to reach a defined pose or to interact with objects. In the figures that represent the other joints, we can see that the anomalies are present and marked with crosses.

5.3 Web Application

The proposed method and the experiments shown before were conducted by having a python script querying the blockchain for data to generate the graphics and data presented before. This is not useful in cases where people may not have the required programming or technological skills to work with scripts and a terminal shell. Thus, we built a web application that shows the robotic logs and if they have an anomaly (detected with the proposed method), they are marked. The important remark about this is not the creation of the web application itself, but the integration of it with RobotChain and how it is possible to communicate directly with it. In this section, we explain how it is possible to build a web application that interacts directly with RobotChain and with the smart-contracts on it.

Oracles normally communicate with RobotChain or any other Tezos network by using RPCs, which causes a procedure to execute in a different address space, being this a node in the blockchain (it can be the computer caller itself that executes this procedure). But there is a library built with javascript called Eztz that serves the purpose of being an API library for Tezos [108]. Eztz creates a layer of abstraction from the queries to the blockchain by providing a set of methods to perform different operations on the blockchain. The operations that Eztz allow to be performed are split into 4 main parts:

- **eztz.crypto:** responsible for key generation, message signing and verification;
- **eztz.node:** responsible for the interaction with Tezos nodes;
- **eztz.rpc:** direct representation of Tezos node JSON RPC API;
- **eztz.contract:** responsible for the interaction with smart-contracts.

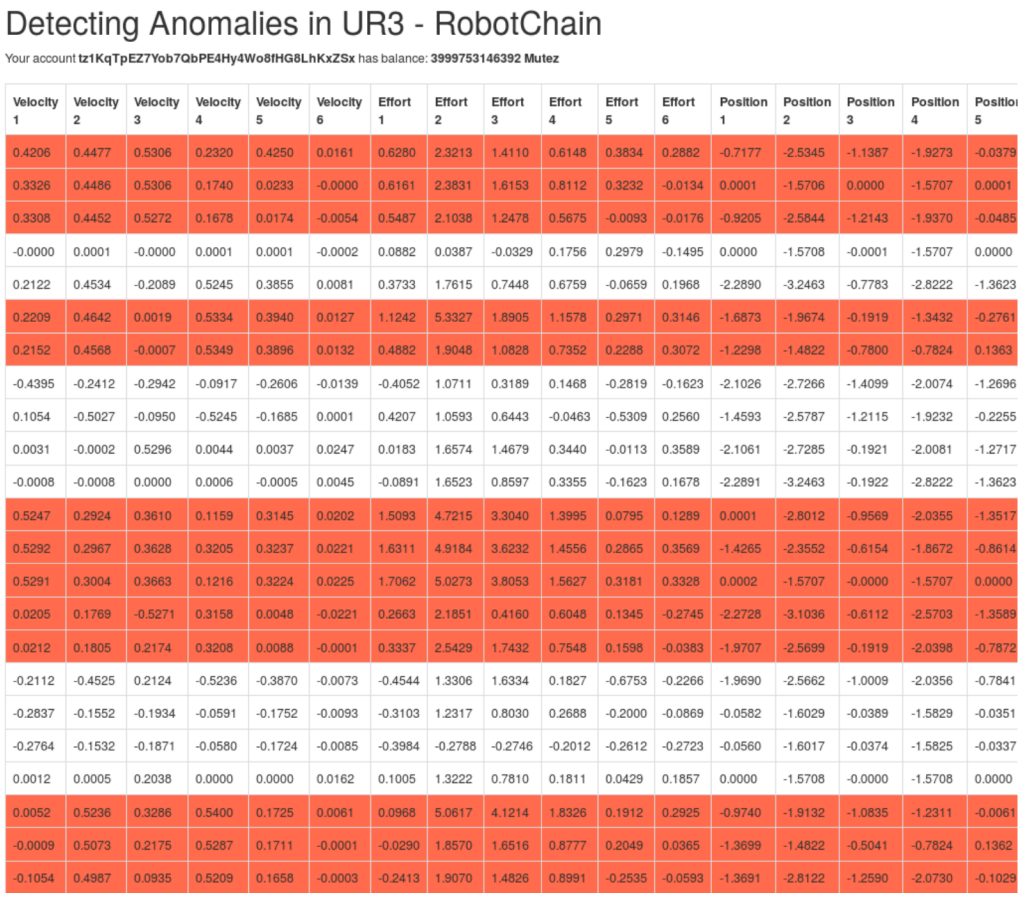
Our web application was built using Eztz for communication. To use it, it was required to perform some steps in order to make it work in our private Tezos blockchain (RobotChain). The first one was to clone the `eztz.min.js` file into the folder that contained our web application and import it in the HTML page by inserting the following code line into the `head` section: `<script src="./eztz.min.js"></script>`. The second step was to create a `.js` file to program the method we require to query the blockchain and insert the data into the HTML page. This file needs also to be imported into the HTML page. The third step was to update the configuration of Tezos and of Eztz to allow the communication to be executed locally. This is due to the fact that we are running RobotChain privately and the computer that wants to access the data needs to at least be running a Tezos node, without obligation to be participating in the consensus algorithm. This configuration was done by:

1. Set the blockchain data provider in the .js file we created by inserting the following line in the beginning of the file: `eztz.node.setProvider("http://localhost:8732");`
2. Change Tezos Node configuration to allow RPC calls to be made locally by inserting: `{"rpc": {"listen-addr": "127.0.0.1:8732", "cors-origin": ["*"], "cors-headers": ["*"]}, "p2p": {"listen-addr": "[::]:9732"}}` in the `config.json` file located in Tezos Node folder.

With the aforementioned configuration steps completed, we created one javascript function that contains four eztz method calls. These methods serve to extract the keys of the user that is running the web application, to extract the balance of the user account, to access a smart-contract storage and to watch a smart-contract every n seconds. The first method was done by calling the `eztz.crypto.extractKeys(file)` function with the file that contains the user keys as a parameter. The method returns the keys, and we extracted the public key to a variable. The second method used was the `eztz.rpc.getBalance(account)` that requires as a parameter a public key and returns the number of Tezzies owned by a given account. The third method was `eztz.contract.storage(ContractAddress)` that receives a smart-contract address, e.g., `KT1TvQ8HhuQnckCLeqGTFfcUajh65VjJSfYh`, and returns the storage that the smart-contract contains. This method is useful to be called upon first opening of the web application to populate it with the values that are already stored in the smart-contract. The fourth and final step was the creation of a method to watch a smart-contract every n seconds. This step allows us to be notified of changes on the smart-contract with a delay predefined by us, so, if we select a delay of 0 seconds, we can watch smart-contract changes in real-time. This was performed by doing a call to the eztz method: `eztz.contract.watch(ContractAddress, Seconds)`. This watcher receives as parameters a contract address and the seconds to wait before checking for changes in the smart-contract. This served as the communication logic between the web application and the blockchain. An illustration of the web application built is shown in Figure 5.7. In this it is possible to see, in the upper part of the figure, the account that is connected and the number of tokens that account holds and a table with values for each joint of the UR3 arm. The values presented in the table are the same values that were presented in the graphics shown in Figure 5.6 with the difference that in this table we coupled grouped 500 logs into 2. These two represent the higher and the lower values found in the group of 500 logs. This was done to reduce the size of the table without compromising the representation of the anomalies. The web application developed serves the purpose of showing how it is possible to create a dynamic web application that receives values from the blockchain and to showcase the anomalies detected using the proposed method in an online platform.

5.4 Conclusion

This chapter described the creation of a module of RobotChain, a novel method to store robotic events in a secure way and a method that uses those events to detect anomalies. By using a blockchain, we increase the security of the data that is registered in an environment that is susceptible to human interaction. This ensures that if a robotic failure occurs, it gets registered in the blockchain and once there, it can't be manually altered. By having such a system, we use smart-contracts to store the information about a UR3 robot in the blockchain and created four datasets, one train, one validation, and two test sets to develop, validate and test a method that can leverage the information acquired from the blockchain to detect robotic anomalies. Our method for detecting anomalies was capable of detecting anomalies induced by counteracting the movement of the arm while performing a pick and place repetitive task. The proposed methods show that it is possible to use the blockchain with robotics and with such a modular system, innovative methods for different purposes can be added by using Oracles.



Detecting Anomalies in UR3 - RobotChain

Your account **tz1KqTpEZ7Yob7QbPE4Hy4Wo8fHG8LhKxZSx** has balance: **3999753146392 Mutez**

Velocity 1	Velocity 2	Velocity 3	Velocity 4	Velocity 5	Velocity 6	Effort 1	Effort 2	Effort 3	Effort 4	Effort 5	Effort 6	Position 1	Position 2	Position 3	Position 4	Positio 5
0.4206	0.4477	0.5306	0.2320	0.4250	0.0161	0.6280	2.3213	1.4110	0.6148	0.3834	0.2882	-0.7177	-2.5345	-1.1387	-1.9273	-0.0379
0.3326	0.4486	0.5306	0.1740	0.0233	-0.0000	0.6161	2.3831	1.6153	0.8112	0.3232	-0.0134	0.0001	-1.5706	0.0000	-1.5707	0.0001
0.3308	0.4452	0.5272	0.1678	0.0174	-0.0054	0.5487	2.1038	1.2478	0.5675	-0.0093	-0.0176	-0.9205	-2.5844	-1.2143	-1.9370	-0.0485
-0.0000	0.0001	-0.0000	0.0001	0.0001	-0.0002	0.0882	0.0387	-0.0329	0.1756	0.2979	-0.1495	0.0000	-1.5708	-0.0001	-1.5707	0.0000
0.2122	0.4534	-0.2089	0.5245	0.3855	0.0081	0.3733	1.7615	0.7448	0.6759	-0.0659	0.1968	-2.2890	-3.2463	-0.7783	-2.8222	-1.3623
0.2209	0.4642	0.0019	0.5334	0.3940	0.0127	1.1242	5.3327	1.8905	1.1578	0.2971	0.3146	-1.6873	-1.9674	-0.1919	-1.3432	-0.2761
0.2152	0.4568	-0.0007	0.5349	0.3896	0.0132	0.4882	1.9048	1.0828	0.7352	0.2288	0.3072	-1.2298	-1.4822	-0.7800	-0.7824	0.1363
-0.4395	-0.2412	-0.2942	-0.0917	-0.2606	-0.0139	-0.4052	1.0711	0.3189	0.1468	-0.2819	-0.1623	-2.1026	-2.7266	-1.4099	-2.0074	-1.2696
0.1054	-0.5027	-0.0950	-0.5245	-0.1685	0.0001	0.4207	1.0593	0.6443	-0.0463	-0.5309	0.2560	-1.4593	-2.5787	-1.2115	-1.9232	-0.2255
0.0031	-0.0002	0.5296	0.0044	0.0037	0.0247	0.0183	1.6574	1.4679	0.3440	-0.0113	0.3589	-2.1061	-2.7285	-0.1921	-2.0081	-1.2717
-0.0008	-0.0008	0.0000	0.0006	-0.0005	0.0045	-0.0891	1.6523	0.8597	0.3355	-0.1623	0.1678	-2.2891	-3.2463	-0.1922	-2.8222	-1.3623
0.5247	0.2924	0.3610	0.1159	0.3145	0.0202	1.5093	4.7215	3.3040	1.3995	0.0795	0.1289	0.0001	-2.8012	-0.9569	-2.0355	-1.3517
0.5292	0.2967	0.3628	0.3205	0.3237	0.0221	1.6311	4.9184	3.6232	1.4556	0.2865	0.3569	-1.4265	-2.3552	-0.6154	-1.8672	-0.8614
0.5291	0.3004	0.3663	0.1216	0.3224	0.0225	1.7062	5.0273	3.8053	1.5627	0.3181	0.3328	0.0002	-1.5707	-0.0000	-1.5707	0.0000
0.0205	0.1769	-0.5271	0.3158	0.0048	-0.0221	0.2663	2.1851	0.4160	0.6048	0.1345	-0.2745	-2.2728	-3.1036	-0.6112	-2.5703	-1.3589
0.0212	0.1805	0.2174	0.3208	0.0088	-0.0001	0.3337	2.5429	1.7432	0.7548	0.1598	-0.0383	-1.9707	-2.5699	-0.1919	-2.0398	-0.7872
-0.2112	-0.4525	0.2124	-0.5236	-0.3870	-0.0073	-0.4544	1.3306	1.6334	0.1827	-0.6753	-0.2266	-1.9690	-2.5662	-1.0009	-2.0356	-0.7841
-0.2837	-0.1552	-0.1934	-0.0591	-0.1752	-0.0093	-0.3103	1.2317	0.8030	0.2688	-0.2000	-0.0869	-0.0582	-1.6029	-0.0389	-1.5829	-0.0351
-0.2764	-0.1532	-0.1871	-0.0580	-0.1724	-0.0085	-0.3984	-0.2788	-0.2746	-0.2012	-0.2612	-0.2723	-0.0560	-1.6017	-0.0374	-1.5825	-0.0337
0.0012	0.0005	0.2038	0.0000	0.0000	0.0162	0.1005	1.3222	0.7810	0.1811	0.0429	0.1857	0.0000	-1.5708	-0.0000	-1.5708	0.0000
0.0052	0.5236	0.3286	0.5400	0.1725	0.0061	0.0968	5.0617	4.1214	1.8326	0.1912	0.2925	-0.9740	-1.9132	-1.0835	-1.2311	-0.0061
-0.0009	0.5073	0.2175	0.5287	0.1711	-0.0001	-0.0290	1.8570	1.6516	0.8777	0.2049	0.0365	-1.3699	-1.4822	-0.5041	-0.7824	0.1362
-0.1054	0.4987	0.0935	0.5209	0.1658	-0.0003	-0.2413	1.9070	1.4826	0.8991	-0.2535	-0.0593	-1.3691	-2.8122	-1.2590	-2.0730	-0.1029

Figure 5.7: Web application that shows the values for the signals of the UR3 arm. Rows that are marked as red mean that an anomaly is present.

Chapter 6

Controlling Robots using Artificial Intelligence and a Consortium Blockchain

Robots are getting more common, especially in tasks that traditionally were performed by humans. There are multiple tasks where robots are working collaboratively with Humans. An example of this is the use of robots to aid health professionals in autism therapies [109], where robots are being used to evaluate the engagement of children that are undergoing autism therapy [110] and react accordingly. Robots have also been used to monitor physical spaces [111, 73] and workspaces [112]. Human-Robot interaction is a crucial role in sensitive environments, like a factory, where robots and humans work close together [113, 114]. There have been some proposals to mitigate problems that can arise from this close interaction, some try to use external sensors to assure a minimal distance between the robot and the human operators [115, 116] and vision to do reactive planning in order to avoid collisions [117]. In Human-Robot interactions, most robots need to be supervised or teleoperated to have certainty that robots behave normally [118]. This control can take many forms, but it is normally teleoperation or supervision of the robot's actions. Even though there has been a huge development in robotics, it is stated that individual robots and robotic networks need to have better security, coordination and control properties, and mechanisms to assure that the robots actuate within certain ethical and moral limits [119]. We find that blockchain can be the solution to these problems if used properly since it can provide auditability, automatic triggers with smart-contracts, global information of robotic swarms and also increases the security of the data and the robots since everything that is stored inside the blockchain is immutable and can't be tampered with.

In the literature, many methods to control robots, that behave differently depending on the information required and the behavior to be achieved, have been proposed [120, 121, 122]. However, to change the behavior of the predefined control it is usually required to do the entire control logic again. In this chapter, we propose a method that allows the integration of different modules with ease and ensures that the data is securely stored. We demonstrate this proposal by developing a novel way of controlling robots. The architecture of the proposed method uses blockchain as a decentralized ledger, that stores information about robots and coupled sensors, like cameras, and smart-contracts to define the logic of control. By having Oracles processing the data and inserting analytics into the blockchain, specialized smart-contracts can use that information to control a robot. We demonstrate this method by controlling a UR3 arm [105] in a scenario that replicates a factory pick-and-place task, where the robot needs to pick raw material and place it in a new location. As the proposed model is modular, it is easy to integrate new modules that perform other tasks or different behavior for the same one, and the method is free of restrictions, which means that it can be used with any robot without requiring the robot to have a specific structure, complexity, price tag or computing power. With this work, we demonstrate that it is possible to control robots with smart-contracts, by proposing a generalist approach that can be easily used to control any robot.

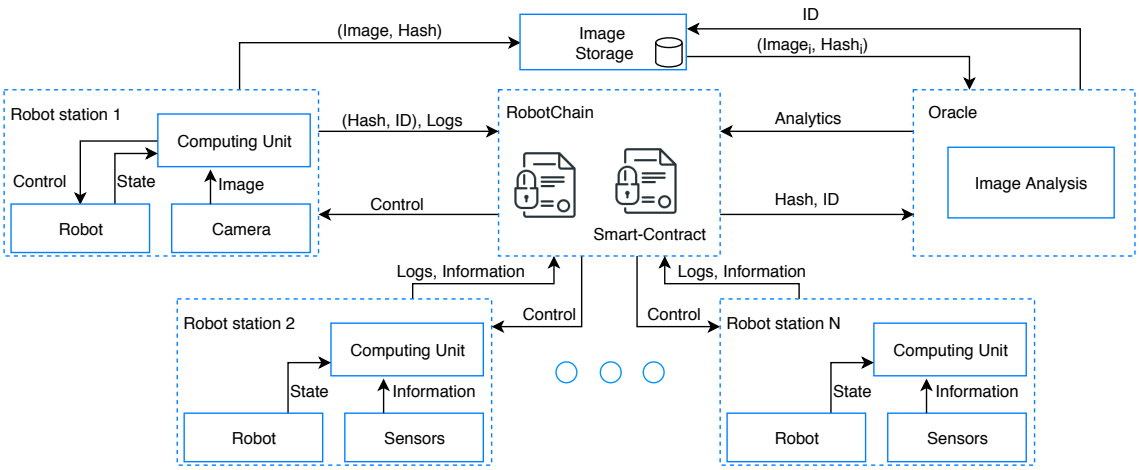


Figure 6.1: Architecture of the proposed method.

The rest of the chapter is organized as follows. In section 6.1, we explain the proposed method, that enhances the capabilities of robotics over blockchain, and his inner components. Then, in section 6.2, we present the experiences conducted and discuss them. Finally, in section 6.3, we provide a conclusion about the proposed method.

6.1 Proposed Method

6.1.1 System Description

We propose a method to control robots in a secure way, which is verified and can't be changed without permission. For this, we use RobotChain as a decentralized ledger, capable of storing robotic events in a secure and fast way. This is useful for having global information about a network and to register information about robots, which contains possible errors that the robots have or perform. We store the information in a smart-contract storage, and we control the robots with smart-contracts as well. This is a generalist approach that can be used with any robot as long as the robot allow external commands to perform changes on its behavior, e.g., changes on the speed. The smart-contract is useful because it contains all the logic for the changes to be made and it is not controlled by any party, in the sense that once published into the blockchain, it is impossible to change the code contained in it and it triggers events without the need of human actions. Another important property is the capability for having different smart-contracts with different control logic, meaning that for different environments or tasks, we can allocate one or more smart-contracts to different types of control of the robot or to other aspects that are found important, e.g., communication with other parties. We demonstrate this approach 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 ping-pong balls and the robot needs to pick it from the end of a track-line and place it at the beginning. We have 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. Denote that the motor that blocks the balls is controlled by an Arduino and it only lets one ball pass each n seconds. This scenario can be seen in Figure 6.2 in which there is one ball on the pick zone and two balls "waiting". The complete sequence of

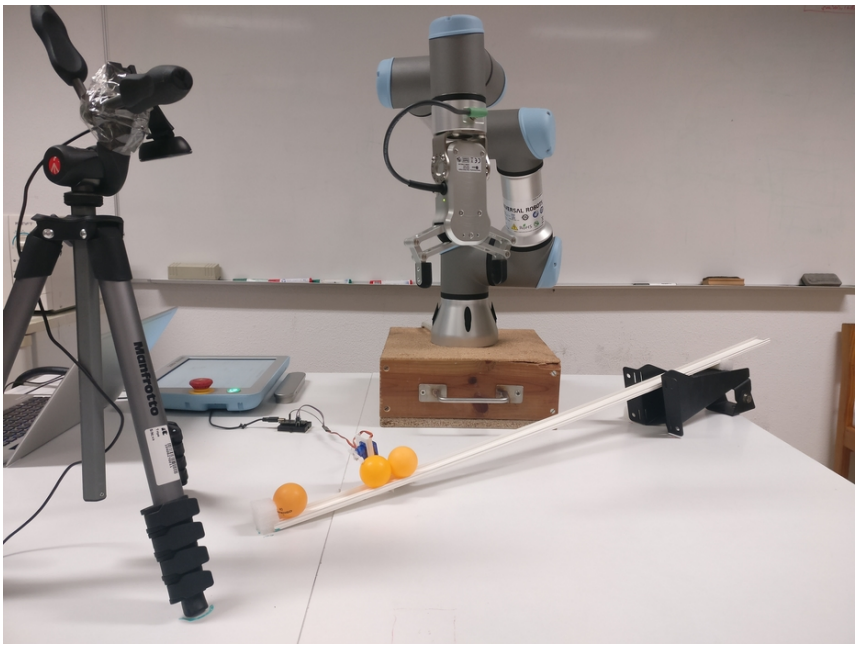


Figure 6.2: Scenario that represents a factory environment pick and place task. In this, the UR3 arm needs to pick objects from the end of the track-line and place them at the beginning.

Figure 6.3: Pick and place task. UR3 arm picks and places a ping-pong on the track and then goes to the home position because there is nothing more to pick.

movements in this task can be seen in Figure 6.3, 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.

The architecture designed for the proposed method is shown in Figure 6.1. This consists of RobotChain as a ledger that 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. In our scenario, 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 [123], we only store on the blockchain, the robotic events (logs) and a tuple comprised of a cryptographic hash and an identifier that represents the image. The image itself is stored in a database. As the information stored in the blockchain can't be changed, the hash of the images adds security to the database. The information on the blockchain is then used by a trusted external Oracle, which can be placed in the cloud or in a central server. 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 pick, or either slow down or speed up, meaning that there are few or many materials to pick, respectively.

6.1.2 Image Analysis

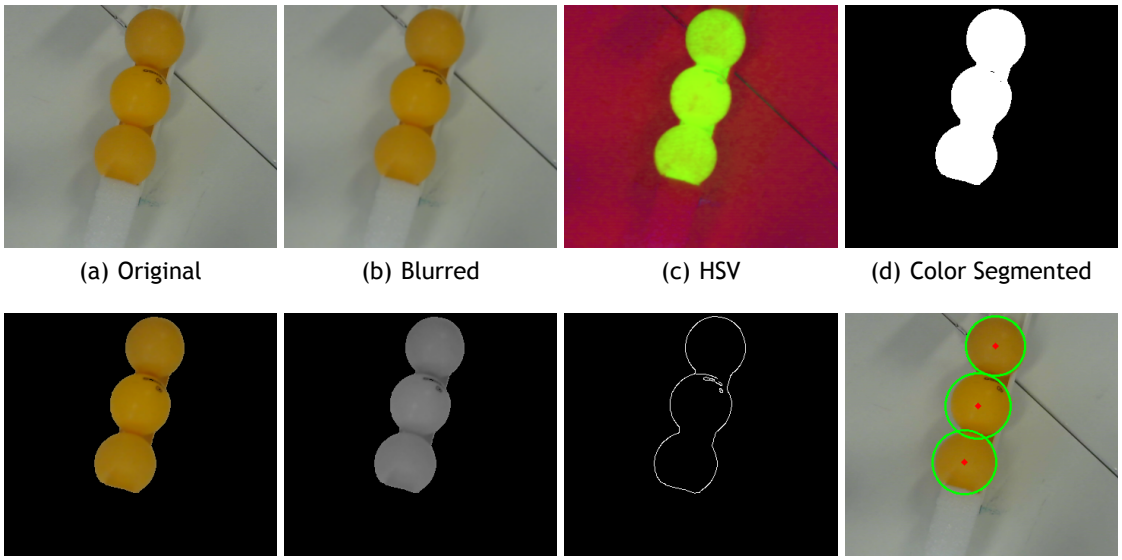
As explained before, we use external parties to the blockchain, Oracles, to process the images captured. The Oracles have read-only access to the database where the images are stored and use the hashes that are in the blockchain to validate that the images are not altered.

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. Initially, the image is pre-processed to remove noise by using a Gaussian Filter with a kernel of size $3 * 3$. What this does is a convolution between the image and the kernel, producing a smoothed image, ultimately reducing the noise, as it is diluted throughout the process. The second step is to transform the result of the first step from an RGB color space to an HSV space. This is done because we want to search for the presence of yellow to orange colors and HSV space allow us to do so in a more natural way and it is less sensitive to the illumination conditions. The search for pixels that are in the range of yellow to orange is the third step, removing all the pixels that fall outside this range. The fourth step is calculating the per-element bitwise conjunction of the result from the third step with itself and using as mask, the blurred imaged from the first step. Then, in the fifth step, the result of the last step is transformed from RGB to a gray image. This is required to perform the next steps that require gray images to detect silhouettes. So, in the sixth step, we perform a Canny Edge detection to detect the edges on the gray image. This gives us the silhouette of the material if there is any present. The final step is the extraction of features from the silhouette in the form of circles. This is done using the Circle Hough Transform, which is a specialization of Hough Transform for detection circular objects. This gives us the number of materials that are present in the image. The detector built for this problem is extremely fast and can handle the feed from cameras with good quality. This is possible because all the processing conducted in the images is optimized and computationally efficient since they only perform basic operations to achieve the desired goal. The most computationally expensive task is the use of the Hough Transform, but it is used with a low number of parameters (search for a circle) and in a very reduced space as the image only contains edges. This process can be seen in image 6.4, where the different image stages that lead to the detection of 3 ping-pong balls is shown.

6.1.3 Robot Control

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, automatically performing actions and to help nodes of a network obtain global information.

Our approach uses smart-contracts to register robotic events and 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, are 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



(a) Original (b) Blurred (c) HSV (d) Color Segmented

(e) Bitwise Conjunction (f) Gray (g) Canny Edge (h) Hough Transform

Figure 6.4: Different stages of the ball detection algorithm performed by the Oracle.

demonstrate the proposed method. The smart-contract method to define the number of balls to be transported is:

$$nballs = ImageAnalysis() + Transporting() \quad (6.1)$$

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:

$$v = \left[\frac{maxSpeed+meanSpeed}{nballs}\right] \quad (6.2)$$

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

6.2 Experiments and 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

Table 6.1: 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

of interacting with Oracles, external process of the data can be performed, which can integrate many forms, but more importantly, it can bring AI to the blockchain, which is the case of the proposed method. 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.

The constraints of our approach are: first, how easy it is for the blockchain size grow, that's why we store an image representation instead of the image itself on the blockchain, but this can be improved by removing unnecessary information in transactions. 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 behavior can be slow. However, this can be improved by changing the consensus algorithm to one that removes the need for computation in order to validate blocks, and if the environment where the blockchain is deployed is a private one like a factory, the connection speed will help, since it has less communication passing through it.

In table 6.1 we present the velocities of the robot at different times from 4 different experiments, 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 experiments 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 experiments conducted were: A: $n = 5s$, B: $n = 15s$, C: $n = 40s$, D: initial $n = 20s$ and incremented by $5s$ every iteration (each time it opens). 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 experiment 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.

6.3 Conclusion

This chapter described how blockchain can be integrated with robotics by using RobotChain as a decentralized ledger. The proposed architecture is capable of registering robotic events and uses 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. This 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 (the 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.

Chapter 7

Robot Workspace Monitoring using a Blockchain-based 3D Vision Approach

The integration of robotics, blockchain and complex algorithms, such as the ones used in computer vision, is not trivial because blockchains are usually designed for financial applications or to handle transactions of monetary assets and are not fit to work with such components. As we showed in this document, there is a lack of proposals that integrate blockchain with robotic systems. Moreover, the lack of proposals that integrate computer vision with blockchain is even higher. Robots deeply rely on vision to conduct many tasks, such as navigation, detection, manipulation control, and others. In order to have efficient integration of blockchain and robotics, it is required to have the perception of how it is possible to use computer vision algorithms with this integration. For this, the most prominent work is the one done by ABBC foundation, who proposed a method to improve the security of wallets that hold tokens by using face recognition to identify the owner of an account [55], and the method we proposed in chapter 6, that uses information from captured images to control the velocity of robots depending on the quantity of raw material needed to be transported, and blockchain to store the information processed by the Oracles and the robot logs, which is then used by smart-contracts that define the changes to be made on the robots.

In this chapter, we propose a method that by using vision, can detect the presence of multiple people inside a robot workspace and have different actions on the robots depending if the workspace has been breached by known or unknown people. The method uses a blockchain to store the logs of the robot, ensuring that all the actions that were taken by a robot are stored and can't be tampered with, the identifications of the images taken by the cameras and the analytics done by the Oracles performing image analysis. We do this by using multiple smart-contracts to serve both as storage and to define the changes to be performed on the robots. The method proposed can be adjusted to different needs since the algorithms that run on the Oracles can be virtually anywhere and new smart-contracts for new needs can easily be added. Even though people detection, tracking and identification are well studied subjects [124, 125, 126, 127], the important focus of our proposal goes beyond the algorithms used to perform the tasks described, to focus on demonstrating how it is possible to integrate robotics with blockchain and how to conduct complex image analysis to change the state of robots using smart-contracts, while keeping immutable registries.

The main contributions of this proposal can be summarized as follows:

- We show how it is possible to conciliate blockchain with computer vision (AI);
- We present a real case scenario where detection of people inside a Cobot workspace is achieved, in such a way that it is possible to register not only who enters the robot workspace, but also:
    - Unalterably keep the identity of the person (including the timestamp);

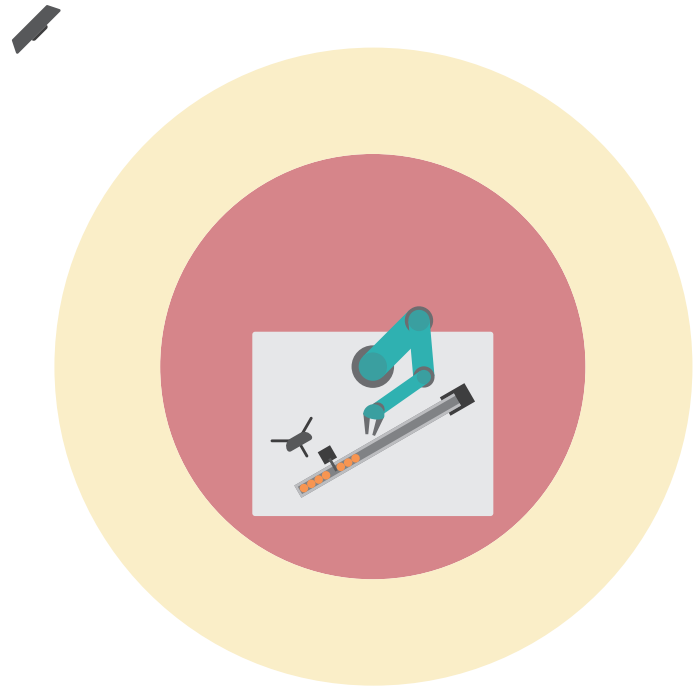


Figure 7.1: Robot workspace monitoring using a Kinect (top-left). Yellow represents the warning zone, red represents the critical zone.

  - Control the robot in order for it to adapt its behavior depending on the identity of the person (if she/he is known or unknown).

- We propose a method that can be used to increase security in environments where humans and robots work closely and that can be easily altered, maintained and adapted to more robots, environments, and tasks.

The remainder of this chapter is organized as follows: Section 7.1, explains the proposed method and, the different components that are part of it. Section 7.2, presents the experiments conducted and discuss them. Finally, section 7.3 provide a conclusion about the proposed method.

7.1 Proposed Method

7.1.1 System Description

The method proposed in this chapter consists of controlling robots to avoid collisions with humans by blockchain and 3D vision. Even though Cobots are normally equipped with sensors to stop upon collision, it is important to stop robots before a collision happens, ensuring that there is no damage neither to the robot nor to the human that violated the robot workspace. To achieve this, we use RobotChain as a decentralized ledger to store robotic events and other information in a secure and fast way. As explained before, RobotChain is a consortium blockchain designed for factory environments, capable of handling a large number of transactions per second, that implements off-chain protocols to handle the exponential growth in size that blockchains have. We use Oracles to process images and smart-contracts to handle the storing

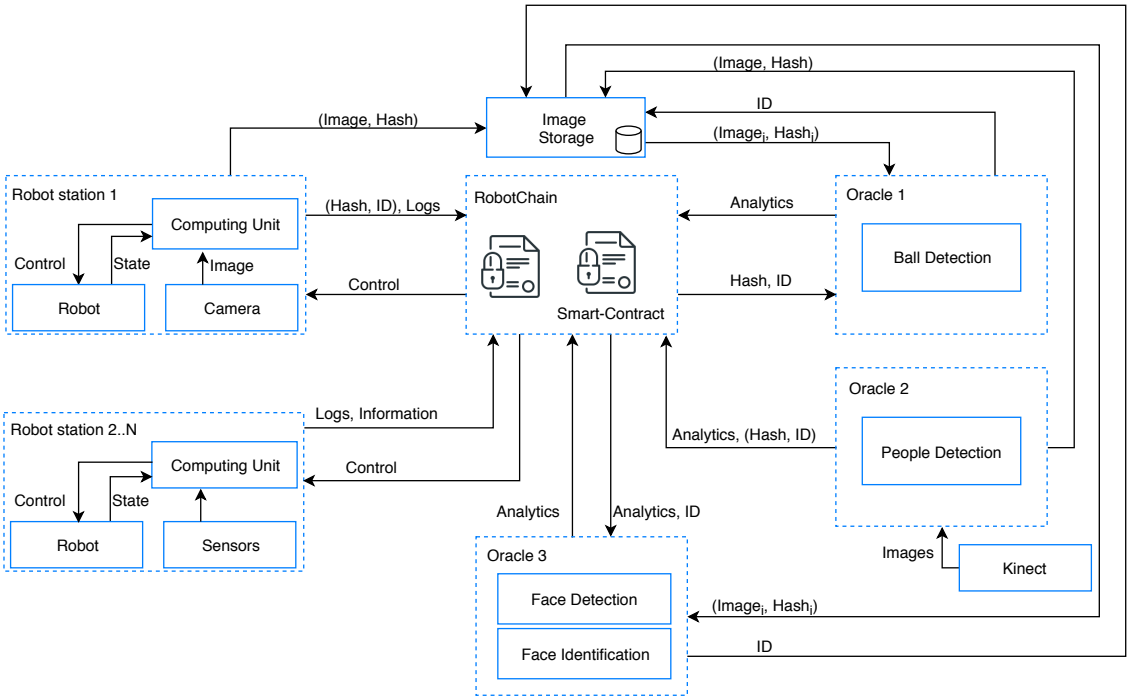


Figure 7.2: Architecture of the proposed method.

of the information and the logic to control robots. This is a generalist approach and can be used to control any robot as long as the robot allows external commands to adjust its velocity and to stop it. This method is also useful as it does not require human intervention to control the robots, since the smart-contracts automatically perform action-triggers when the premises are met, and it is easy to maintain and adapt to handle new robots and tasks since it is only needed to add a new smart-contract to handle the new requirements. To avoid collisions, we use 3D images to detect people using the method proposed in [128, 129] and, if there are people present in the warning or the critical zone, we then use [130] to detect faces and identify who they are. Figure 7.1 shows a representation of this setup, where a Kinect is placed on the left side to acquire depth information about the scene. With the captured images, we define two zones surrounding the robot, the warning, and the critical zone, which are represented in the figure by yellow and red respectively. In our approach, if someone is known (allowed to be near robots, e.g., a factory manager) and enters the warning zone, no changes are made to the normal functioning of the robot, but that event is registered in the blockchain. If a person enters the red zone, the velocity of the robot is reduced to 10 seconds per movement and this event and the event that represents the detection and identification of a person are both registered in the blockchain. When the person is unknown (not allowed to be there, e.g., a factory visitor) and enters the warning zone, the velocity of the robot is reduced to 10 seconds per movement, and if the person enters the critical zone, the robot is stopped.

Figure 7.2 shows the complete architecture of the proposed method. It is possible to see from this that the most important module of this architecture is the blockchain, as it serves as a ledger to store all the information but most importantly, as a way for different modules communicate and to share information about the system. The way the blockchain interacts with the other system components is with smart-contracts. They serve both as storage and to handle different logics, such as determining a robot speed depending on the information from

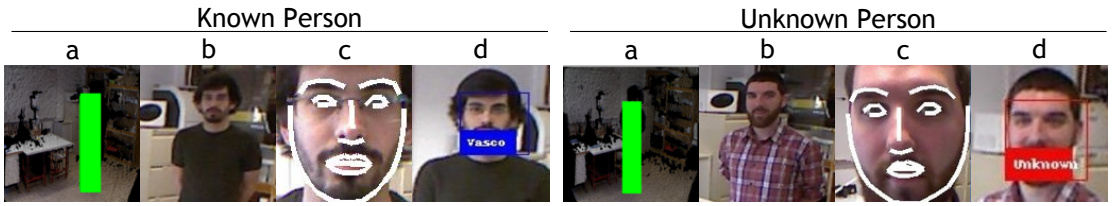


Figure 7.3: Detection of known and unknown people. Step A represents the people detection algorithm, B the face detection, C the feature extraction and D the identification of the person.

the Oracles. The robots are coupled with a computing unit that handles the interaction with the blockchain and, if needed, with an external database that stores the images. This ensures that the blockchain does not grow in size exponentially and, by saving the hash of the images on the blockchain, it ensures that if the images are altered, this will be known. The computing units receive information from the smart-contracts about the changes to be conducted in the robots. In the proposed method we have three Oracles, one to handle the task that the robot used in the experiments it is doing, one to detect people based on the 3D image acquired by a Kinect and one to detect faces and identify to whom they belong. The first Oracle is based on the method proposed in the last chapter, in which a robotic arm is performing a pick and place task and uses a camera to detect if there are any raw materials to pick. The second and the third one are used to compute vision algorithms, and they share information by using a smart-contract that stores information about people being detected on the 3D images and the identification for those images (which are stored in a database). The third Oracle receives the information that the second Oracle inserted in the blockchain and retrieves the correspondent images to perform its task, which is detecting faces and identifying them. After concluding its work, it inserts information about how many people were detected and their identities (if known) in the blockchain. With this, a smart-contract automatically determines if there are any changes to be conducted on the robot. In Figure 7.3 we show a simplified flow of the method, for a known and an unknown person. Step *a* represents the detection of a person in the robot workspace, *b* represents the face detection, *c* is the feature extraction and *d* is the identification of the person. If the person is not known, the method returns “Unknown”. After these steps, the smart-contract defines the actions to be taken.

The proposed method is a novel approach on how to safely validate a robotic workspace and control the robot to adjust to different scenarios. The created method integrates blockchain, robotics, and AI in the form of image processing algorithms, which is an innovative way of dealing with computer vision algorithms and robotics. By using multiple Oracles to process data we can distribute and decentralize the computation and securely store the information inside the blockchain.

In the following subsections, we explain the components of the system and how they work.

7.1.2 RobotChain and Oracles

We use RobotChain as the blockchain [2]. RobotChain allows the registration of robotic and other events in a trusted and secure way. It has properties that are essential in robotic environments, such as: a low energy consumption and faster consensus algorithm (Delegated Proof-of-Stake), when compared to traditional ones, e.g., Proof-of-Work, it also has self-amending properties

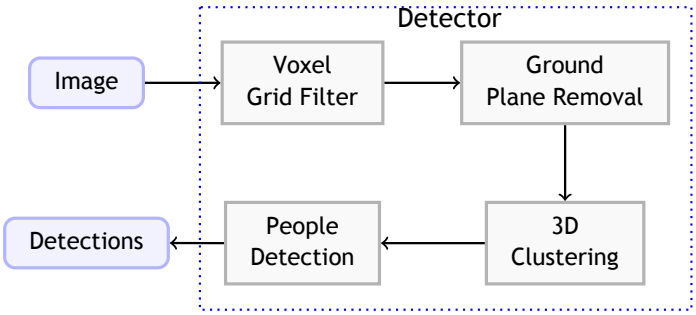
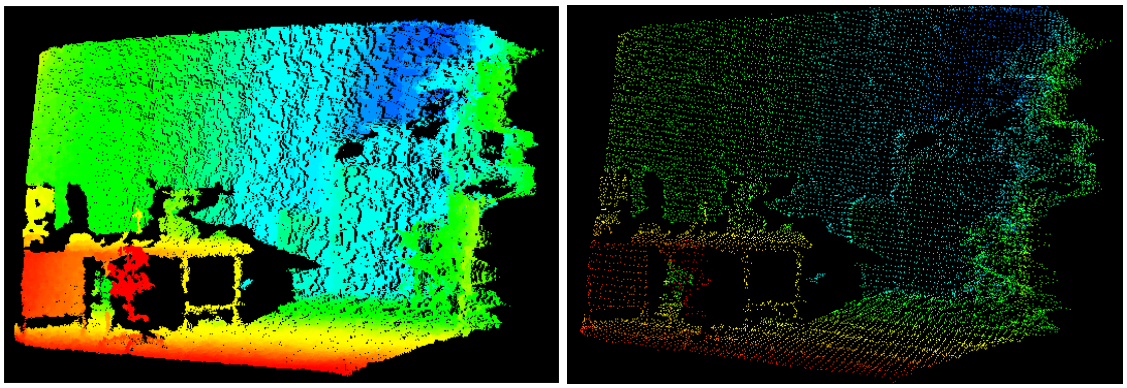


Figure 7.4: Diagram describing the method to detect people in 3D images.

that allow changes in the blockchain core to be conducted without the need for hard-forks and the support to formally verify the smart-contracts. This is essential in critical software to ensure that the amount of developer induced bugs is reduced. We use smart-contracts to store information, not only from the robots but also from Oracles, which insert analytics into them. Oracles can be virtually anywhere, e.g., in the cloud, requiring only a network to communicate with the blockchain network. It is essential that the address of the Oracle(s) is defined on the smart-contract logic. This enforces that only allowed parties can update the information on the blockchain. In the proposed method, there are three separated Oracles performing computation on data (from the blockchain and external sensors, such as a Kinect) and inserting analytics on smart-contracts that are dedicated to: control the robot, store information about people that enter the robot workspace and about their identities. The smart-contracts allow the system either to be private and closed (not allowing external parties to communicate) or to allow trusted parties to interact with it, opening the opportunity for complex algorithms to be performed and to allow other applications to work over them.

7.1.3 People Detection

To detect people we use 3D images captured by a Kinect and process them with the Point Cloud Library [131]. We based our approach in the one proposed in [128, 129]. The detector runs on CPU and is capable of detecting people in real-time, even if they are grouped or near walls. In Figure 7.4 we present an overview of the detection method in the form of a diagram. Initially, the detector receives a 3D image and down samples it using a voxel grid filter. This filter works by subdividing the space into a set of voxels (volumetric pixels) and all points inside each voxel are approximated to the coordinates of their centroid. The result of this pre-processing is shown in Figure 7.5, where the image on the left represents the original 3D image and the image on the right is the result of the filtering process. The next step performed by the detector is to remove ground planes. This is done using RANSAC to calculate the largest plane and removing all the inliers. The third step is to perform a 3D clustering based on Euclidean distances. This gives possible locations where people might be in the 3D image. Based on these results, a Histogram of Oriented Gradients people detector [132] is applied to the part of the RGB image corresponding to the cluster bounding box. This outputs as results either the presence of people in each cluster or not. As we know the position of the cluster bounding boxes, we know where are the detections located. We further changed this method to include the warning and critical areas aforementioned and to extract the image correspondent to the bounding boxes of the detected people in order to send that information to the blockchain, reducing the space where it is required to search for faces.



(a) Original Point Cloud (b) After Voxel Grid Filtering

Figure 7.5: Comparison of two Point Clouds. One original, and one after being filtered using Voxel Grid Filter.

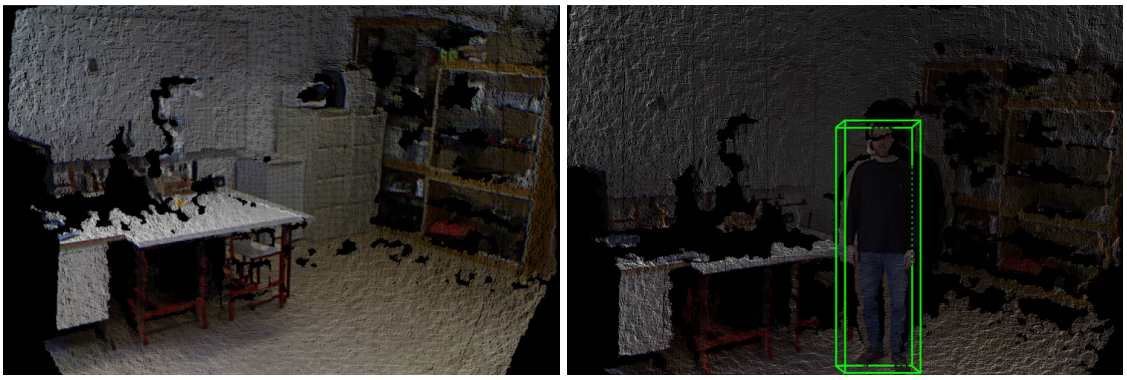


Figure 7.6: People detection algorithm. The image on the left was captured when there were no people on the scene. In the second image, one person was detected and a bounding box over the person.

In Figure 7.6 we show an example of this detection in action. We present two 3D images, where the left one does not contain people in it. In the right image, there was one person detected and a green bounding box was drawn over the person.

7.1.4 Face Detection and Identification

The pipeline of an algorithm that aims to do face recognition contains several steps: first, it needs to find all the faces on a picture; then, for each face, it needs to be able to understand the direction that the face is facing, since it is unusual that a person is looking directly at the camera; third, it needs to be able to extract the unique features of the face, so that those features can be used to distinguish one person from another. The final step is normally the comparison of the unique features of the face with all the people that are already known. This serves to determine if the person is known or unknown to us (identifying to whom the face belongs).

To detect faces and identify them, we based our approach on a library called `Face_recognition` [130]. This library is a facial recognition API for Python that interfaces with *dlib* [133]. *Dlib* is a toolkit to aid in the development of machine learning algorithms and data analysis tasks and it is widely used due to its efficient run time. `Face_recognition` uses different algorithms to

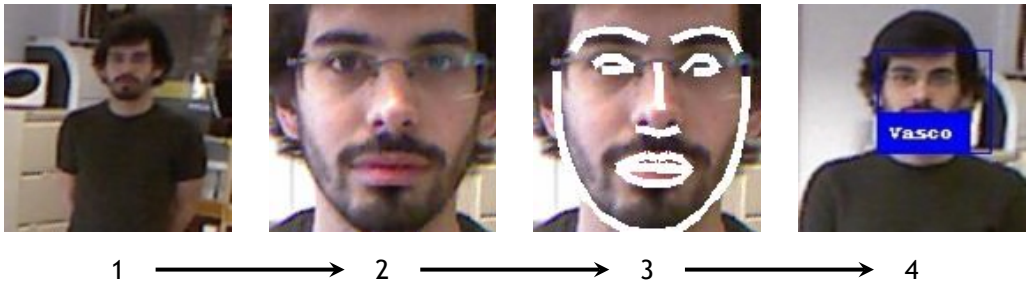


Figure 7.7: Pipeline of the Face Detection and Identification method.

tackle the steps aforementioned in a face recognition algorithm. In the first stage, a Histogram of Oriented Gradients [132] is used to find faces in an image. On the second stage, it needs to figure out the pose of the face by finding the main landmarks in the face. These landmarks are found using the face landmark estimation method [134], which is based on an ensemble of regression trees. Once landmarks are found, they are used to warping the image so that the eyes and mouth are centered. After this, the information is then used in FaceNet [135], which is trained to measure 128 features of a face. This NN is used to get simpler data from the image to have a faster search time on the features instead of doing traditional methods such as comparing pixels or sliding windows. In the end, a linear SVM classifier is used to identify a person as unknown or known and get the person identifier in the latter case.

We based our approach on this method due to the fact that it is easy to use out-of-the-box, it achieved 99.38% on the Labeled Faces in the Wild benchmark [136] and it is easy to maintain, add and remove people from the known database. In Figure 7.7 we show an example of the pipeline of the face recognition and identification that we use. The first step is the information about the image received by the people detection algorithm. The second is the face detection and image crop. The third is the feature extraction and finally, the last step is the identification of the person.

7.1.5 Robot Control

The robot is controlled by smart-contracts. This approach is based on the one proposed in chapter 6. It uses a blockchain to store robot events and depending on the amount of material needed to be picked, a smart-contract changes the speed of the robot to accommodate the needs of the factory. However, this approach uses only one Oracle. In our approach, we use a similar method to control the robots but we use multiple Oracles to perform different types of algorithms to achieve the desired behavior. In the proposed method, the control of the robot is inside a smart-contract, which means that it can't be changed (without having a specific method that allows so) and performs automatically upon meeting the criteria. The criteria are the detection of people trespassing the robot workspace and their identification. If the trespassing happens in the warning zone, it either slows down the robot if the person is unknown or does nothing, if the person is known and allowed to be there. In the case that the trespassing happens in the critical zone if the person is unknown, the robot is stopped, if the person is known, the robot is slowed down. By controlling the robots in such a way, we ensure that they are controlled without human intervention, that the outcomes of the algorithm are as desired and that every event is registered in a secure and immutable way, meaning that it is possible

to see eventual errors and have real-time information about what is and should be happening with the robots.

7.2 Experiments and Discussion

To illustrate the working conditions of the proposed method, we conducted two experiments in a scenario where a robotic arm is picking and placing material from one point to another, in which the material returns to the starting point. This simulates a factory job where a robotic arm needs to transport something from one conveyor to another one. Figure 7.1 represents the scenario created, where the robotic arm picks orange balls from the beginning of the trail and places them at the end, which is slightly higher so that the balls go back to the beginning. The first experiment conducted consisted of a known person entering into the robot workspace, and the second one consisted of an unknown person doing the same. The logic defined in the smart-contract to handle these situations is: if an unknown person enters the warning zone, the robot must change its speed to 10 seconds per movement, if the person enters the critical zone, the robot must stop. For a known (allowed) person, if he/she enters the warning zone, no changes are performed, if the person enters the critical zone, the robot must change its speed to 10 seconds per movement until the person leaves this zone. The normal speed for the robot is 4 seconds per movement.

In Figure 7.8, two charts that show the velocity of the robot in seconds per movement are shown. The first one represents the experiment with an unknown person while the second one represents the experiment with a known person. In the first image, the points A and B represent the moment when the person enters the warning and the critical zone, respectively. In this, it is possible to see that initially the robot is working at a constant speed of 4 seconds per movement, once the person is detected inside the warning zone, the smart-contract "fires" the information that the speed should be changed to 10 seconds per movement, which is visible in the passage from second 29 to second 30. Then, the person keeps moving inside the warning zone until the second 56, where the person enters the red zone and the robot is stopped. The experiment finished with the robot velocity being 0, which means that it is stopped until new orders. In the second chart, the same experiment is shown but with a person that is allowed to be near the robot. In this case, there are three points, A that represents the moment a person enters the warning zone, B which represents the entry on the critical zone and C, that represents the moment the person leaves the critical zone. It is possible to see that, because the person is known and allowed to be there, in point A, no changes are made to the robot speed. In point B, as the person enters the critical zone, the velocity of the robot is reduced to 10 seconds per movement (as opposed to stopping if the person were unknown). Then, as the person leaves the area, the robot returns to its normal behavior, which is a velocity of 4 seconds per movement.

7.3 Conclusion

This chapter describes how it is possible to integrate robotics with vision algorithms while using a blockchain as a decentralized ledger. The proposed architecture is capable of registering robotic events and Oracle information in a secure way and use the blockchain as the gate between all communications. This proposal also shows that the integration of such technologies goes

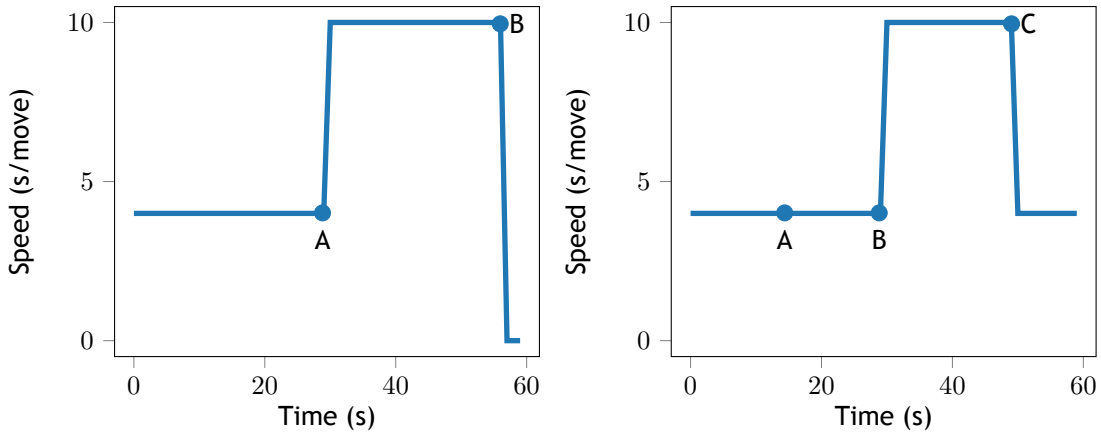


Figure 7.8: Experiments conducted, the first chart represents an experiment with a known person while the second chart represents an experiment with an unknown person. Point A represents the entry in the warning zone, B represents the entry into the critical zone, C represents the moment when the person leaves the critical zone.

beyond a simple way of registering information, to more complex behaviors and possibilities. We use Oracles to do heavy computational processing that robots can't do, which leads to more opportunities in the type of tasks the robots can perform. By using a blockchain we also have global information about the whole system in real-time. The proposed architecture is modular, as it is possible to insert new modules that can serve as Oracles and do any type of data processing, or new smart-contracts to handle new tasks and behaviors. This method also ensures that once a smart-contract is deployed to the blockchain, no one can alter its logic and that the whole system can work without human intervention. The method can be expanded to handle multiple robots just by adding smart-contracts that perform the required actions to communicate with those robots, meaning that a whole swarm of robots can be integrated, and they can share information over the blockchain.

Chapter 8

Conclusion and Future Work

8.1 Conclusion

In this work, our first goal was to study the approaches that use blockchain to leverage other systems. With a keen interest in methods that use blockchain with AI and robotics. But the main goal was to perceive how it was possible to: 1) give AI algorithms access to the information stored on the blockchain; 2) improve AI capabilities leveraging the large amount of data available; and 3) allow AI algorithms to register their own events on the blockchain. These goals were generalist in terms of explicitly saying how this work should be done to achieve them due to the fact that there was no previous work to support our ideas and goals. But they were also explicit enough to define that our work was intended to, somehow, have AI algorithms, blockchain, and robotics working seamlessly in the same system.

To achieve such goals, we conducted an extensive and detailed overview of the methods that use blockchain with AI, robotics or that try to improve blockchain by using AI [10, 11]. Doing so, we concluded that almost all academical and industrial proposals that try to do such integration are either missing or are just using blockchain as a ledger. Besides that, we have also identified the ineffectiveness of the systems that store robotic logs. Those systems normally have low security and can be tampered with. We proposed an architecture that uses blockchain as a decentralized ledger and smart-contracts to process robot logs in order to automatically and autonomously find anomalies [12]. With this, we were capable of using smart-contracts for both the action of implementing algorithms, outside the financial scope, to detect anomalies and to serve as storage for the logs generated by robots. With this, we assured that the data could not be tampered with, that only allowed robots could insert values into the smart-contracts and that only allowed Oracles could read the values and insert analytics in smart-contracts dedicated to storing such information. This work served to prove that blockchain and robots can be integrated and that it is possible to conduct analysis to the information presented in the blockchain, resulting in an anomaly detection algorithm that was capable of detecting when a real UR3 arm conducted abnormal events. With this base, we proposed a second method. This method tackled the problem of controlling robots [13]. Most Cobot environments require that the robots are controlled by humans or that the robots are constantly supervised. Our proposal uses smart-contracts to control the robot, ensuring that the movements the robot takes are well-defined and are always sent to the robot the same way. We created a scenario to test this method, in which a UR3 arm performs a pick-and-place task and a webcam is looking at the conveyor to check for the existence of material to be picked. The images are sent to the blockchain and processed by Oracles, that insert into a smart-contract the quantity of material present. With this information, the smart-contract defines the speed of the robot - slowing it down if there is few materials to be picked, speeding it if there is many material to be picked or stop it until there is something to be picked. This proposal also has the advantage that because smart-contracts define the control logic, no human intervention is required, because

smart-contracts run autonomously, that the logic can be propagated to other robots with ease and that if the robot is required to do an old-task, the smart-contract is still present in the blockchain and can be activated again. The third and final method proposed [14] uses the others and proposes an innovative way to solve the problem of factory robots requiring to be isolated, to avoid damaging the robot or hurting a human. Our proposal uses blockchain to store the robot logs and information regarding the robot workspace. This information is in the form of 3D images extracted from a Kinect. These images are processed by Oracles to detect the presence of people in the workspace, more specifically, inside a warning or a critical zone. If they are any, that information is stored on the blockchain and a different Oracle tries to identify the person. If the person is known and is inside the warning zone, the information is only registered. But if the person walks into the critical zone, the information is not only registered, but the robot movement speed is slowed down to 10 seconds per movement. On the other hand, if the person is unknown and walks inside the warning zone, the robot is slowed down to 10 seconds per movement. If the person walks into the critical zone, the robot is completely stopped. This method proves that it is possible to use heavily computation algorithms with blockchain and provides an architecture that is auditable and that can store information in a secure way. Experiments were conducted in all the proposed methods, and they were successful, showing that the intersection between robotics, AI and blockchain is not only possible, but useful.

In conclusion, the development of robotics and AI approaches over the blockchain is still in its infancy and will start to flourish once the blockchain gets more common and easy to use and deploy. In the future, there will be blockchain approaches that are totally focused on integrating multiple parties over the same platform to distribute a way to share information without consensus problems and without financial approaches underneath. These methods will be used in the industry 4.0 paradigm and will integrate the cloud in their inner components, either to work as Oracles or partially store the information of the blockchain. Doing work in a field that is narrow in terms of development and past research has its disadvantages. The biggest one is the lack of past work, since we can't rely on literature to answer questions that are related to complementary work, which is not truly focused on the work we conduct but are also needed to do it. However, this also presented us with good and exciting challenges, since we are marking the first steps in the field, and we are demystifying uncharted territory so that people that come after us have a little more to base on than we did. Furthermore, we successfully proposed three different methods that integrate blockchain with robotics and AI. We also explained how can smart-contracts be used to serve as a communication gate between the blockchain and Oracles, we have shown how it is possible to create web applications that are fed with data from the blockchain and the proposed methods were capable of: monitoring a robot internal state, control a robot by conducting image analysis, both from 2D and 3D images, and were capable of communicating with external parties that had AI algorithms running that leveraged from the blockchain data. Therefore, we consider all our goals fulfilled.

8.2 Future Work

Even though this is outside the scope of this work, the proposed architectures could be further improved by changing blockchain core features, such as the consensus algorithm to one that does not require any token, since in robotic environments it is usually not required (when thinking of factories and industrial services) or by replacing it for a reputation system, which could

be useful for task management and consensus. As future work, it would be important to see from a different perspective how the integration of robotics and blockchain can outperform the traditional systems. This could be done by showing how attacks or byzantine robots are handled by this approach and how it would affect the network. In addition, we are also interested in extending our work so that it provides more modules to the blockchain. A new module could be the creation of a method to conduct federated learning with all the information that the robots insert into the blockchain. With this, the robots could learn from the others to perform new tasks and to improve in the ones they already perform. This could be used in households, where robots inside a house communicate through a blockchain and insert into it the objects they detect so others can improve their models. This could also be used to ask for information about an object - if a robot does not know how to classify one object, he could ask the network to classify it for him.

Bibliography

[1] *The Voting Process*, Tezos Documentation, 2018, accessed: 2018-12-31. [Online]. Available: http://tezos.gitlab.io/master/whitedoc/voting.html xv, 25

[2] M. Fernandes and L. A. Alexandre, "Robotchain: Using tezos technology for robot event management," in *Proceedings of the Symposium on Blockchain for Robotic Systems, MIT Media Lab, December 2018*. Pittsburgh: Ledger, December 2018, p. 32–41. xv, 15, 18, 27, 28, 62

[3] T. T. Andersen, "Optimizing the universal robots ros driver." Technical University of Denmark, Department of Electrical Engineering, Tech. Rep., 2015. [Online]. Available: http://orbit.dtu.dk/en/publications/optimizing-the-universal-robots-ros-driver(20dde139-7e87-4552-8658-dbf2cdaab24b).html xv, 37, 38

[4] R. de Nóbrega Nogueira, "Self-adaptive Cobots in Cyber-Physical Production Systems," Master's thesis, Universidade do Porto, Portugal, 2019. xvii, 39

[5] Y. Yuan and F.-Y. Wang, "Towards blockchain-based intelligent transportation systems," in *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2016, pp. 2663–2668. 1, 8

[6] B. Leiding and W. V. Vorobev, "Enabling the vehicle economy using a blockchain-based value transaction layer protocol for vehicular ad-hoc networks," 2018. 1

[7] X. Yue, H. Wang, D. Jin, M. Li, and W. Jiang, "Healthcare data gateways: found healthcare intelligence on blockchain with novel privacy risk control," *Journal of medical systems*, vol. 40, no. 10, p. 218, 2016. 1, 8

[8] P. Mamoshina, L. Ojomoko, Y. Yanovich, A. Ostrovski, A. Botezatu, P. Prikhodko, E. Izumchenko, A. Aliper, K. Romantsov, A. Zhebrak *et al.*, "Converging blockchain and next-generation artificial intelligence technologies to decentralize and accelerate biomedical research and healthcare," *Oncotarget*, vol. 9, no. 5, p. 5665, 2018. 1, 8

[9] M. Mettler, "Blockchain technology in healthcare: The revolution starts here," in *2016 IEEE 18th International Conference on e-Health Networking, Applications and Services (Healthcom)*, Sep. 2016, pp. 1–3. 1

[10] V. Lopes and L. A. Alexandre, "An overview of blockchain integration with robotics and artificial intelligence," *CoRR*, vol. abs/1810.00329, 2018. [Online]. Available: http://arxiv.org/abs/1810.00329 2, 69

[11] V. Lopes and L. Alexandre, "An overview of blockchain integration with robotics and artificial intelligence," *Ledger*, vol. 4, no. 0, 2019. [Online]. Available: https://ledgerjournal.org/ojs/index.php/ledger/article/view/171 2, 69

[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. 2, 69

[13] V. Lopes, L. A. Alexandre, and N. Pereira, "Controlling robots using artificial intelligence and a consortium blockchain," *CoRR*, vol. abs/1903.00660, 2019. [Online]. Available: http://arxiv.org/abs/1903.00660 3, 69

[14] V. Lopes, N. Pereira, and L. A. Alexandre, "Robot workspace monitoring using a blockchain-based 3d vision approach," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, June 2019. 3, 70

[15] L. Lamport, "The part-time parliament," *ACM Trans. Comput. Syst.*, vol. 16, no. 2, pp. 133–169, May 1998. [Online]. Available: http://doi.acm.org/10.1145/279227.279229 5

[16] S. Nakamoto, "Bitcoin: A Peer-to-Peer Electronic Cash System," *Www.Bitcoin.Org*, p. 9, 2008. [Online]. Available: https://bitcoin.org/bitcoin.pdf 5

[17] Z. Zheng, S. Xie, H. Dai, X. Chen, and H. Wang, "An Overview of Blockchain Technology: Architecture, Consensus, and Future Trends," *Proceedings - 2017 IEEE 6th International Congress on Big Data, BigData Congress 2017*, pp. 557–564, 2017. 5

[18] A. J. Menezes, S. A. Vanstone, and P. C. V. Oorschot, *Handbook of Applied Cryptography*, 1st ed. Boca Raton, FL, USA: CRC Press, Inc., 1996. 6

[19] M. Conti, S. K. E, C. Lal, and S. Ruj, "A survey on security and privacy issues of bitcoin," *IEEE Communications Surveys Tutorials*, pp. 1–1, 2018. 6

[20] D. Yaga, P. Mell, N. Roby, and K. Scarfone, "Blockchain Technology Overview (NISTIR-8202)," NIST: National Institute of Standards and Technology, Tech. Rep., October 2018. [Online]. Available: https://csrc.nist.gov/publications/detail/nistir/8202/final 7

[21] V. Buterin, "A next-generation smart contract and decentralized application platform," *Etherum*, no. January, pp. 1–36, 2014. [Online]. Available: http://buyxpr.com/build/pdfs/EthereumWhitePaper.pdf 7, 29

[22] N. Szabo, "Formalizing and securing relationships on public networks," vol. 2, 01 1997. 7, 29

[23] S. Popov, "The tangle," IOTA, Tech. Rep., April 2018, accessed: 2018-10-15. [Online]. Available: https://assets.ctfassets.net/r1dr6vzfxhev/2t4uxvsIqk0EUau6g2sw0g/45eae33637ca92f85dd9f4a3a218e1ec/iota1_4_3.pdf 8

[24] EOS.IO, "Eos.io technical white paper v2," EOS, Tech. Rep., March 2018, accessed: 2018-10-15. [Online]. Available: https://github.com/EOSIO/Documentation/blob/master/TechnicalWhitePaper.md 8

[25] Skycoin, "Skycoin business whitepaper," Skycoin, Tech. Rep., accessed: 2018-10-15. [Online]. Available: https://downloads.skycoin.net/whitepapers/Skycoin-Whitepaper-v1.0.pdf 8

[26] L. Goodman, "Tezos — a self-amending crypto-ledger," Tech. Rep., August 2014, accessed: 2018-09-24. [Online]. Available: https://github.com/tezos/tezos-papers 8, 23, 27

[27] L. M. Goodman, "Tezos - White paper," Tech. Rep. July 2016, September 2014, accessed: 2018-09-25. [Online]. Available: https://github.com/tezos/tezos-papers 8, 23, 27

[28] Digiconomist, “Bitcoin energy consumption index,” Digiconomist, Tech. Rep., accessed: 2018-10-18. [Online]. Available: https://digiconomist.net/bitcoin-energy-consumption 8

[29] Z. Zheng, S. Xie, X. Chen, and H. Wang, “Blockchain challenges and opportunities: a survey,” *IJWGS*, vol. 14, pp. 352–375, 2018. 8

[30] K. Christidis and M. Devetsikiotis, “Blockchains and smart contracts for the internet of things,” *IEEE Access*, vol. 4, pp. 2292–2303, 2016. 8

[31] W. Li, S. Tug, W. Meng, and Y. Wang, “Designing collaborative blockchained signature-based intrusion detection in iot environments,” *Future Generation Computer Systems*, vol. 96, pp. 481 – 489, 2019. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0167739X18327237 8

[32] T. Marwala and B. Xing, “Blockchain and Artificial Intelligence,” *arXiv preprint arXiv:1802.04451*, p. 13, 2018. [Online]. Available: http://arxiv.org/abs/1802.04451 8, 14, 16

[33] J. Chen, K. Duan, R. Zhang, L. Zeng, and W. Wang, “An AI Based Super Nodes Selection Algorithm in BlockChain Networks.” [Online]. Available: https://arxiv.org/pdf/1808.00216.pdf 8, 14, 16

[34] D. Chain, “White paper: Deepbrain chain artificial intelligence computing platform driven by blockchain,” Tech. Rep., 2017. [Online]. Available: https://www.deepbrainchain.org/ 8, 10, 16

[35] Neuromation, “White paper: Neuromation - ”where androids dream of electric sheep”,” Tech. Rep., October 2017. [Online]. Available: https://neuromation.io/neuromation_white_paper_eng.pdf 8, 10, 16

[36] K. Salah, M. H. U. Rehman, N. Nizamuddin, and A. Al-Fuqaha, “Blockchain for ai: Review and open research challenges,” *IEEE Access*, vol. 7, pp. 10 127–10 149, 2019. 8

[37] Namahe, “White paper: Namahe sustainable value chains,” Tech. Rep., 2018. 9, 16

[38] SingularityNET, “White paper: Singularitynet: A decentralized, open market and inter-network for ais,” Tech. Rep., December 2017. [Online]. Available: https://public.singularitynet.io/whitepaper.pdf 9, 16

[39] F. Vogelsteller and V. Buterin, “ERC-20 Token Standard,” Internet Requests for Comments, Ethereum, RFC 20, November 2015. [Online]. Available: https://github.com/ethereum/EIPs/blob/master/EIPS/eip-20.md 9

[40] G. Wood, “Ethereum: A secure decentralised generalised transaction ledger,” *Ethereum project yellow paper*, vol. 151, pp. 1–32, 2014. 9

[41] D. Hart and B. Goertzel, “Opencog: A software framework for integrative artificial general intelligence,” in *Proceedings of the 2008 Conference on Artificial General Intelligence 2008: Proceedings of the First AGI Conference*. Amsterdam, The Netherlands, The Netherlands: IOS Press, 2008, pp. 468–472. [Online]. Available: http://dl.acm.org/citation.cfm?id=1566174.1566223 10

[42] H. R. Limited, “Sohpia,” http://www.hansonrobotics.com/robot/sophia/, accessed: 2018-09-06. 10

[43] R. Craib, G. Bradway, X. Dunn, and J. Krug, “White paper: Numeraire: A cryptographic token for coordinating machine intelligence and preventing overfitting,” Tech. Rep., February 2017. [Online]. Available: https://numer.ai/ 10, 16

[44] Aitheon, “White paper: Aitheon,” Tech. Rep., April 2018. [Online]. Available: https://www.aitheon.com/ 11, 16

[45] Eligma, “White paper: Eligma - ”ai-driven and blockchain-based cognitive commerce platform”,” Tech. Rep., March 2018. [Online]. Available: https://www.eligma.com/pdf/eligma-white-paper_v1.1.pdf 11, 16

[46] TraDove, “White paper: Tradove - find connect trade,” Tech. Rep., 2018. [Online]. Available: https://bbcoin.tradove.com 11, 16

[47] AdHive, “White paper: Adhive ”add your passion - the first ai-controlled platform for influencer marketing”,” Tech. Rep., 2017. [Online]. Available: https://adhive.tv/docs/AdHive_Whitepaper.pdf 12, 16

[48] Matrix, “Matrix technical whitepaper,” Tech. Rep., 2018. [Online]. Available: https://www.matrix.io/html/MATRIXTechnicalWhitePaper.pdf 12, 16

[49] ATN, “Atn white paper: Leveraging blockchain to technology to provide a secure, trustful, and interoperable a.i. marketplace.” Tech. Rep., 2017. [Online]. Available: https://atn.io/system/whitepaper-en.pdf 12, 16

[50] Z. Chen, W. Wang, X. Yan, and J. Tian, “Cortex - ai on blockchain: The decentralized ai autonomous system,” Tech. Rep., August 2018. [Online]. Available: https://whitepaperdatabase.com/cortex-ctxc-whitepaper/ 13, 16

[51] N. Inc., “White paper: Nam coin - revolution in medical care with ai and blockchain,” Tech. Rep., July 2018. [Online]. Available: https://namchain.net/whitepaper/EN_whitepaper_20180728134338.pdf 13, 16

[52] D. Nagothu, R. Xu, S. Y. Nikouei, and Y. Chen, “A Microservice-enabled Architecture for Smart Surveillance using Blockchain Technology,” 2018. [Online]. Available: http://arxiv.org/abs/1807.07487 14, 16

[53] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” in *Advances in Neural Information Processing Systems 25*, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2012, pp. 1097–1105. [Online]. Available: http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf 14

[54] N. Alchemy, “A short history of smart contract hacks on ethereum,” https://medium.com/new-alchemy/a-short-history-of-smart-contract-hacks-on-ethereum-1a30020b5fd, accessed: 2018-09-20. 14

[55] ABBC, “White paper: Abbc,” Tech. Rep., March 2018. [Online]. Available: https://www.abbcfoundation.com/assets/whitepaper_pdf/whitepaper_v_2.01_eng.pdf/ 15, 16, 59

[56] B. Degardin, *Blockchain for Robotic Event Recognition*. Universidade da Beira Interior, 2018. 15, 18

[57] E. C. Ferrer, "The blockchain: a new framework for robotic swarm systems," 2016. [Online]. Available: http://arxiv.org/abs/1608.00695 15, 18

[58] Amazon, "Amazonrobotics : Vision," https://www.amazonrobotics.com/#/vision, accessed: 2018-09-21. 15

[59] A. Brown, "Amazon's army of more than 100,000 warehouse robots," https://www.dailymail.co.uk/sciencetech/article-5808319/Amazon-100-000-warehouse-robots-company-insists-replace-humans.html, accessed: 2018-09-21. 15

[60] E. C. Ferrer, O. Rudovic, T. Hardjono, and A. Pentland, "RoboChain: A Secure Data-Sharing Framework for Human-Robot Interaction," *eTELEMED conference*, 2018. [Online]. Available: http://arxiv.org/abs/1802.04480 16, 18

[61] T. Hardjono, D. Shrier, and A. Pentland, *Trust::Data: A New Framework for Identity and Data Sharing*. Visionary Future, 2016. 17

[62] N. Teslya and A. Smirnov, "Blockchain-based framework for ontology-oriented robots' coalition formation in cyberphysical systems," *MATEC Web Conf.*, vol. 161 EDP Sciences, no. 03018, pp. 1–6, 2018, https://doi.org/10.1051/matecconf/201816103018. 17, 18

[63] 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*. Springer, 2017, pp. 75–91. 17, 18

[64] V. Strobel and M. Dorigo, "Blockchain technology for robot swarms: A shared knowledge and reputation management system for collective estimation," IRIDIA, Université Libre de Bruxelles, Brussels, Belgium, Tech. Rep. TR/IRIDIA/2018-009, May 2018. 17, 18

[65] V. Strobel, E. Castelló Ferrer, and M. Dorigo, "Managing byzantine robots via blockchain technology in a swarm robotics collective decision making scenario," in *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*, ser. AAMAS '18. Richland, SC: International Foundation for Autonomous Agents and Multiagent Systems, 2018, pp. 541–549. [Online]. Available: http://dl.acm.org/citation.cfm?id=3237383.3237464 17, 18

[66] E. C. Ferrer, T. Hardjono, and A. Pentland, "Secure and secret cooperation of robotic swarms by using merkle trees," *CoRR*, vol. abs/1904.09266, 2019. [Online]. Available: http://arxiv.org/abs/1904.09266 17, 18

[67] Kambria, "Fueling the robotics economy," Kambria, Tech. Rep., accessed: 2018-10-17. [Online]. Available: https://kambria.io/Kambria_White_Paper_v2_20180615.pdf 17, 18

[68] S. Lonshakov, A. Krupenkin, A. Kapitonov, E. Radchenko, A. Khassanov, and A. Starostin, "Robonomics: platform for integration of cyber physical systems into human economy," Robonomics, Tech. Rep., accessed: 2018-10-17. [Online]. Available: https://robonomics.network/robonomics_white_paper_en.pdf 17

[69] B. Jakimovski and E. Maehle, "Artificial immune system based robot anomaly detection engine for fault tolerant robots," in *Autonomic and Trusted Computing*, C. Rong, M. G. Jaatun, F. E. Sandnes, L. T. Yang, and J. Ma, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 177–190. 18

[70] B. Jakimovski, M. Litza, F. Mösch, and A. E. S. Auf, "Development of an organic computing architecture for robot control," in *GI Jahrestagung*, 2006. 18

[71] H. Lu, Y. Li, S. Mu, D. Wang, H. Kim, and S. Serikawa, "Motor anomaly detection for unmanned aerial vehicles using reinforcement learning," *IEEE Internet of Things Journal*, vol. 5, no. 4, pp. 2315–2322, Aug 2018. 19

[72] "Markov chain-based feature extraction for anomaly detection in time series and its industrial application," *Proceedings of the 30th Chinese Control and Decision Conference, CCDC 2018*, pp. 1059–1063, 2018. 19

[73] W. Lawson, E. Bekele, and K. Sullivan, "Finding Anomalies with Generative Adversarial Networks for a Patrolbot," *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, vol. 2017-July, pp. 484–485, 2017. 19, 51

[74] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," *CoRR*, vol. abs/1511.06434, 2015. 19

[75] H. Lau, I. Bate, and J. Timmis, "An immuno-engineering approach for anomaly detection in swarm robotics," in *ARTIFICIAL IMMUNE SYSTEMS, PROCEEDINGS*, P. Andrews, J. Timmis, N. Owens, U. Aickelin, E. Hart, A. Hone, and A. Tyrrell, Eds., vol. 5666 LNCS. SPRINGER-VERLAG BERLIN, 2009, pp. 136–150. 19

[76] F. Gonzalez and D. Dasgupta, "Neuro-immune and self-organizing map approaches to anomaly detection: a comparison," *Proceedings of the First International Conference on Artificial Immune Systems*, pp. 203–211, 2002. 20

[77] X. Guo, C. Shen, and L. Chen, "Deep fault recognizer: An integrated model to denoise and extract features for fault diagnosis in rotating machinery," *Applied Sciences*, vol. 7, no. 1, 2017. [Online]. Available: http://www.mdpi.com/2076-3417/7/1/41 20

[78] K. Loparo, "Case western reserve university bearing data center," Tech. Rep., accessed on 24 November 2018. [Online]. Available: http://csegroups.case.edu/bearingdatacenter/home 20

[79] Z. Huo, Y. Zhang, and L. Shu, "A short survey on fault diagnosis of rotating machinery using entropy techniques," *Lecture Notes of the Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering, LNICST*, vol. 221, pp. 279–284, 2018. 20

[80] A. Verma, S. Sarangi, and M. Kolekar, "Misalignment faults detection in an induction motor based on multi-scale entropy and artificial neural network," *Electric Power Components and Systems*, vol. 44, pp. 1–12, 04 2016. 20

[81] S. Aouabdi, M. Taibi, S. Bouras, and N. Boutasseta, "Using multi-scale entropy and principal component analysis to monitor gears degradation via the motor current signature analysis," *Mechanical Systems and Signal Processing*, vol. 90, pp. 298 – 316, 2017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0888327016305490 20

[82] D. Gu, J. Kim, T. Kelimu, S.-C. Huh, and B.-K. Choi, "Evaluation of the use of envelope analysis and dwt on ae signals generated from degrading shafts," *Materials Science and Engineering: B*, vol. 177, no. 19, pp. 1683 – 1690, 2012, structural Materials: Properties, Microstructure and Processing. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0921510712001250 20

[83] Y. He and X. Zhang, "Approximate entropy analysis of the acoustic emission from defects in rolling element bearings," *Journal of Vibration and Acoustics*, vol. 134, p. 061012, 12 2012. 20

[84] H. Zhao, M. Sun, W. Deng, and X. Yang, "A new feature extraction method based on eemd and multi-scale fuzzy entropy for motor bearing," *Entropy*, vol. 19, no. 1, 2017. [Online]. Available: http://www.mdpi.com/1099-4300/19/1/14 20

[85] F. Jia, Y. Lei, J. Lin, X. Zhou, and N. Lu, "Deep neural networks: A promising tool for fault characteristic mining and intelligent diagnosis of rotating machinery with massive data," *Mechanical Systems and Signal Processing*, vol. 72-73, pp. 303–315, 2016. 20

[86] X. Lou and K. A. Loparo, "Bearing fault diagnosis based on wavelet transform and fuzzy inference," *Mechanical Systems and Signal Processing*, vol. 18, no. 5, pp. 1077 – 1095, 2004. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0888327003000773 20

[87] O. Janssens, V. Slavkovikj, B. Vervisch, K. Stockman, M. Loccufier, S. Verstockt, R. Van de Walle, and S. Van Hoecke, "Convolutional Neural Network Based Fault Detection for Rotating Machinery," *Journal of Sound and Vibration*, vol. 377, pp. 331–345, 2016. 20

[88] O. Janssens, R. Schulz, V. Slavkovikj, K. Stockman, M. Loccufier, R. Van De Walle, and S. Van Hoecke, "Thermal image based fault diagnosis for rotating machinery," *Infrared Physics and Technology*, vol. 73, pp. 78–87, 2015. [Online]. Available: http://dx.doi.org/10.1016/j.infrared.2015.09.004 21

[89] "Deep Learning for Infrared Thermal Image Based Machine Health Monitoring," *IEEE/ASME Transactions on Mechatronics*, vol. 23, no. 1, pp. 151–159, 2018. 21

[90] A. Bagozi, D. Bianchini, V. De Antonellis, A. Marini, and D. Ragazzi, "Big data summarisation and relevance evaluation for anomaly detection in cyber physical systems," in *OTM Confederated International Conferences" On the Move to Meaningful Internet Systems"*. Springer, 2017, pp. 429–447. 21

[91] C. C. Aggarwal, J. Han, J. Wang, and P. S. Yu, "A framework for clustering evolving data streams," in *Proceedings of the 29th international conference on Very large data bases-Volume 29*. VLDB Endowment, 2003, pp. 81–92. 21

[92] D. Pelleg, A. W. Moore *et al.*, "X-means: Extending k-means with efficient estimation of the number of clusters." in *Icml*, vol. 1, 2000, pp. 727–734. 21

[93] Tezos, "The michelson language," https://www.michelson-lang.com/, accessed: 2018-04-14. 23, 29

[94] V. Buterin, "Slasher: A punitive proof-of-stake algorithm," https://blog.ethereum.org/2014/01/15/slasher-a-punitive-proof-of-stake-algorithm/, accessed: 2018-09-26. 23

[95] K. J. Wilkinson and A. Reinhardt, “Cryptocurrencies without Proof of Work,” no. 240258, pp. 1–18, 2005. 23

[96] B. Wiki, “Proof-of-burn,” https://en.bitcoin.it/wiki/Proof_of_burn, accessed: 2018-09-26. 23

[97] J. Arluck, “Amending tezos: Traversing the amendment process,” November 2018, accessed: 2018-12-31. [Online]. Available: https://medium.com/tezos/amending-tezos-b77949d97e1e 25

[98] Tzscan, “Tezos proposals: Chart,” accessed: 2019-1-1. [Online]. Available: https://tzscan.io/charts?Chart_id=proposals 25

[99] Tzscan, “Tezos protocols,” accessed: 2019-1-1. [Online]. Available: https://tzscan.io/protocols 25

[100] Z. Li, J. Kang, R. Yu, D. Ye, Q. Deng, and Y. Zhang, “Consortium blockchain for secure energy trading in industrial internet of things,” *IEEE Transactions on Industrial Informatics*, vol. 14, no. 8, pp. 3690–3700, Aug 2018. 27

[101] C. Natoli and V. Gramoli, “The balance attack or why forkable blockchains are ill-suited for consortium,” in *2017 47th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, June 2017, pp. 579–590. 27

[102] M. Fernandes and L. A. Alexandre, “A Time-Segmented Consortium Blockchain for Robotic Event Registration,” *arXiv e-prints*, p. arXiv:1904.04306, Apr 2019. 28

[103] Ethereum, “Solidity.” [Online]. Available: https://solidity.readthedocs.io/ 29

[104] OCamlPRO, “The liquidity language,” http://www.liquidity-lang.org/, accessed: 2019-04-14. 29, 31

[105] “Ur3 robot,” https://www.universal-robots.com/products/ur3-robot/, accessed: 2019-05-21. 35, 42, 51

[106] “Rg2 gripper,” https://onrobot.com/en/products/rg2-gripper, accessed: 2019-05-29. 36

[107] D. Coleman, I. A. Sucan, S. Chitta, and N. Correll, “Reducing the barrier to entry of complex robotic software: a moveit! case study,” *CoRR*, vol. abs/1404.3785, 2014. [Online]. Available: http://arxiv.org/abs/1404.3785 37

[108] TezTech, “eztz - Javascript API library for Tezos,” https://github.com/TezTech/eztz, 2017. 48

[109] O. Rudovic, J. Lee, M. Dai, B. Schuller, and R. W. Picard, “Personalized machine learning for robot perception of affect and engagement in autism therapy,” *Science Robotics*, vol. 3, no. 19, 2018. 51

[110] O. Rudovic, Y. Utsumi, J. Lee, J. Hernandez, E. C. Ferrer, B. Schuller, and R. W. Picard, “Culturenet: A deep learning approach for engagement intensity estimation from face images of children with autism,” in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 339–346. 51

[111] M. Dunbabin and L. Marques, "Robots for environmental monitoring: Significant advancements and applications," *IEEE Robotics Automation Magazine*, vol. 19, no. 1, pp. 24–39, March 2012. 51

[112] S. Bexten, J. Scholle, J. Saenz, C. Walter, and N. Elkmann, "Validation of workspace monitoring and human detection for soft safety with collaborative mobile manipulator using machine learning techniques in the," in *ISR 2018; 50th International Symposium on Robotics*. VDE, 2016, pp. 1–8. 51

[113] I. Maurtua, A. Ibarguren, J. Kildal, L. Susperregi, and B. Sierra, "Human–robot collaboration in industrial applications: Safety, interaction and trust," *International Journal of Advanced Robotic Systems*, vol. 14, no. 4, p. 1729881417716010, 2017. 51

[114] A. Bauer, D. Wollherr, and M. Buss, "Human–robot collaboration: a survey," *International Journal of Humanoid Robotics*, vol. 5, no. 01, pp. 47–66, 2008. 51

[115] F. Vicentini, N. Pedrocchi, M. Giussani, and L. Molinari Tosatti, "Dynamic safety in collaborative robot workspaces through a network of devices fulfilling functional safety requirements," in *ISR/Robotik 2014; 41st International Symposium on Robotics*, June 2014, pp. 1–7. 51

[116] S. Sarkar, G. Ghosh, A. Mohanta, A. Ghosh, and S. Mitra, "Arduino based foot pressure sensitive smart safety system for industrial robots," in *2017 Second International Conference on Electrical, Computer and Communication Technologies (ICECCT)*, Feb 2017, pp. 1–6. 51

[117] G. Dumonteil, G. Manfredi, M. Devy, A. Confetti, and D. Sidobre, "Reactive planning on a collaborative robot for industrial applications," *2015 12th International Conference on Informatics in Control, Automation and Robotics (ICINCO)*, vol. 02, pp. 450–457, 2015. 51

[118] T. B. Sheridan, "Human–robot interaction: Status and challenges," *Human Factors*, vol. 58, no. 4, pp. 525–532, 2016, pMID: 27098262. 51

[119] G.-Z. Yang, J. Bellingham, P. E. Dupont, P. Fischer, L. Floridi, R. Full, N. Jacobstein, V. Kumar, M. McNutt, R. Merrifield, B. J. Nelson, B. Scassellati, M. Taddeo, R. Taylor, M. Veloso, Z. L. Wang, and R. Wood, "The grand challenges of science robotics," *Science Robotics*, vol. 3, no. 14, 2018. 51

[120] T. Balch and R. C. Arkin, "Behavior-based formation control for multirobot teams," *IEEE Transactions on Robotics and Automation*, vol. 14, no. 6, pp. 926–939, Dec 1998. 51

[121] A. M. Almeshal, M. O. Tokhi, and K. M. Goher, "Robust hybrid fuzzy logic control of a novel two-wheeled robotic vehicle with a movable payload under various operating conditions," in *Proceedings of 2012 UKACC International Conference on Control*, Sep. 2012, pp. 747–752. 51

[122] A. Pereira and M. Althoff, "Safety control of robots under computed torque control using reachable sets," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, May 2015, pp. 331–338. 51

[123] Y. Nishida, K. Kaneko, S. Sharma, and K. Sakurai, "Suppressing chain size of blockchain-based information sharing for swarm robotic systems," in *2018 Sixth International Symposium on Computing and Networking Workshops (CANDARW)*, Nov 2018, pp. 524–528. 53

[124] L. Xia, C. Chen, and J. K. Aggarwal, "Human detection using depth information by kinect," in *CVPR 2011 WORKSHOPS*, June 2011, pp. 15–22. 59

[125] J. Neves and H. Proenca, ""A Leopard Cannot Change Its Spots": Improving face recognition using 3d-based caricatures," *IEEE Transactions on Information Forensics and Security*, vol. 14, no. 1, pp. 151–161, Jan 2019. 59

[126] T. Linder and K. O. Arras, "People detection, tracking and visualization using ros on a mobile service robot," in *Robot Operating System (ROS)*. Springer, 2016, pp. 187–213. 59

[127] X. Peng, N. Ratha, and S. Pankanti, "Learning face recognition from limited training data using deep neural networks," in *2016 23rd International Conference on Pattern Recognition (ICPR)*, Dec 2016, pp. 1442–1447. 59

[128] M. Munaro, F. Basso, and E. Menegatti, "Tracking people within groups with rgb-d data," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Oct 2012, pp. 2101–2107. 61, 63

[129] M. Munaro and E. Menegatti, "Fast rgb-d people tracking for service robots," *Autonomous Robots*, vol. 37, no. 3, pp. 227–242, Oct 2014. [Online]. Available: https://doi.org/10.1007/s10514-014-9385-0 61, 63

[130] A. Geitgey, "face_recognition the world's simplest facial recognition api for python and the command line," 2018. 61, 64

[131] R. B. Rusu and S. Cousins, "3D is here: Point Cloud Library (PCL)," in *IEEE International Conference on Robotics and Automation (ICRA)*, Shanghai, China, May 9-13 2011. 63

[132] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *international Conference on computer vision & Pattern Recognition (CVPR'05)*, vol. 1. IEEE Computer Society, 2005, pp. 886–893. 63, 65

[133] D. E. King, "Dlib-ml: A machine learning toolkit," *Journal of Machine Learning Research*, vol. 10, pp. 1755–1758, 2009. 64

[134] V. Kazemi and J. Sullivan, "One millisecond face alignment with an ensemble of regression trees," in *2014 IEEE Conference on Computer Vision and Pattern Recognition*, June 2014, pp. 1867–1874. 65

[135] F. Schroff, D. Kalenichenko, and J. Philbin, "Facenet: A unified embedding for face recognition and clustering," *CoRR*, vol. abs/1503.03832, 2015. [Online]. Available: http://arxiv.org/abs/1503.03832 65

[136] G. B. Huang, M. Ramesh, T. Berg, and E. Learned-Miller, "Labeled faces in the wild: A database for studying face recognition in unconstrained environments," University of Massachusetts, Amherst, Tech. Rep. 07-49, October 2007. 65

Glossary

Anomaly Detection	The process of identifying deviations from normal conditions. Normally anomaly detectors are trained using only samples from normal data.
Blockchain	Distributed ledger, comprised of immutable, digitally recorded data in blocks. Each block is then chained to the next block, using a cryptographic signature. This allows blockchain to be shared and accessed by anyone in the network.
Cobot	Robot intended to physically interact with humans in a shared workspace. Normally, Cobots contain internal sensors to detect collisions and defined limits in terms of velocity, effort, and acceleration so they can be used near humans safely.
Consensus Algorithm	An algorithm that is designed to achieve consent in a network involving multiple nodes, where some may be unreliable. This is conducted by forcing the nodes to solve a problem - normally a mathematical one.
Decentralization	In the context of blockchains, it means that the authority and responsibility of centralized organizations, governments or parties are transferred to a distributed network.
Decentralized Application	An application that has its software running on a decentralized peer-to-peer network rather than a centralized server. Normally these are created using smart-contracts.
Deep Learning	A subset of ML methods that are based on NN. The deep represents the higher amount of layers that these NN contain. Normally, these architectures reduce the need for feature extraction.
Fault Detection	The process of identifying which anomaly occurred. Fault detection algorithms are based on anomaly detection algorithms but are trained with both normal and abnormal samples to be able to identify the type of anomaly.
Hard Fork	A rule change in a blockchain core process that makes the blocks validated according to the new rules incompatible and invalid with the present blockchain. This requires a new blockchain that is compatible with the new rules to be initialized.
Mining	The process by which transactions are verified and added to a blockchain. This process involves the attempt to solve the consensus problem.

Neural Networks	A set of algorithms that performs a process of training to recognize the underlying relationships in a set of data through a process that tries to mimic the way the human brain operates.
Oracle	An external party to the blockchain that communicates with it via smart-contracts. Oracles can be used to process information or insert external information into the blockchain.
RobotChain	Consortium blockchain designed for robotic systems. It is based in the public blockchain Tezos but has off-chain properties to reduce the space required to store the data and the time to validate the blocks.
Smart-contract	Is a computer program that defines the rules and penalties related to an agreement in the same way that a traditional contract does, but it can also automatically enforce those obligations once the premises are met.
White Paper	Official document issued by a blockchain organization to inform the readers about the technology, methodology, product or service provided.