UNIVERSIDAD DE INVESTIGACIÓN DE TECNOLOGÍA EXPERIMENTAL YACHAY

Escuela de Ciencias Matemáticas y Computacionales

Balancing an articulated pole, controlled by artificial intelligence using an Arduino board

Trabajo de integración curricular presentado como requisito para la obtención del título de Ingeniero en Tecnologías de la Información.

Autor:

Revelo Orellana José Luis

Tutor:

PhD. Chang Tortolero Oscar Guillermo

Urcuquí, agosto 2021

Urcuquí, 3 de agosto de 2021

SECRETARÍA GENERAL
(Vicerrectorado Académico/Cancillería)
ESCUELA DE CIENCIAS MATEMÁTICAS Y COMPUTACIONALES
CARRERA DE TECNOLOGÍAS DE LA INFORMACIÓN
ACTA DE DEFENSA No. UITEY-ITE-2021-00026-AD

A los 3 días del mes de agosto de 2021, a las 14:00 horas, de manera virtual mediante videoconferencia, y ante el Tribunal Calificador, integrado por los docentes:

Presidente Tribunal de Defensa	Dr. ARMAS ARCINIEGA, JULIO JOAQUIN , Ph.D.
Miembro No Tutor	Dr. AMARO MARTIN, ISIDRO RAFAEL , Ph.D.
Tutor	Dr. CHANG TORTOLERO, OSCAR GUILLERMO , Ph.D.

El(la) señor(ita) estudiante **REVELO ORELLANA, JOSE LUIS**, con cédula de identidad No. **1722217591**, de la **ESCUELA DE CIENCIAS MATEMÁTICAS Y COMPUTACIONALES**, de la Carrera de **TECNOLOGÍAS DE LA INFORMACIÓN**, aprobada por el Consejo de Educación Superior (CES), mediante Resolución **RPC-SO-43-No.496-2014**, realiza a través de videoconferencia, la sustentación de su trabajo de titulación denominado: **Balancing an articulated pole, controlled by artificial intelligence, using an Arduino board.**, previa a la obtención del título de **INGENIERO/A EN TECNOLOGÍAS DE LA INFORMACIÓN.**

El citado trabajo de titulación, fue debidamente aprobado por el(los) docente(s):

Tutor	Dr. CHANG TORTOLERO, OSCAR GUILLERMO , Ph.D.

Y recibió las observaciones de los otros miembros del Tribunal Calificador, las mismas que han sido incorporadas por el(la) estudiante.

Previamente cumplidos los requisitos legales y reglamentarios, el trabajo de titulación fue sustentado por el(la) estudiante y examinado por los miembros del Tribunal Calificador. Escuchada la sustentación del trabajo de titulación a través de videoconferencia, que integró la exposición de el(la) estudiante sobre el contenido de la misma y las preguntas formuladas por los miembros del Tribunal, se califica la sustentación del trabajo de titulación con las siguientes calificaciones:

Tipo	Docente	Calificación
Miembro Tribunal De Defensa	Dr. AMARO MARTIN, ISIDRO RAFAEL , Ph.D.	10,0
Presidente Tribunal De Defensa	Dr. ARMAS ARCINIEGA, JULIO JOAQUIN , Ph.D.	9,0
Tutor	Dr. CHANG TORTOLERO, OSCAR GUILLERMO , Ph.D.	10,0

Lo que da un promedio de: **9.7 (Nueve punto Siete)**, sobre 10 (diez), equivalente a: **APROBADO**

Para constancia de lo actuado, firman los miembros del Tribunal Calificador, el/la estudiante y el/la secretario ad-hoc.

REVELO ORELLANA, JOSE LUIS
Estudiante

Dr. ARMAS ARCINIEGA, JULIO JOAQUIN , Ph.D.
Presidente Tribunal de Defensa

Firmado electrónicamente por:
JULIO JOAQUIN ARMAS ARCINIEGA

Dr. CHANG TORTOLERO, OSCAR GUILLERMO , Ph.D.
Tutor

Firmado electrónicamente por:
OSCAR GUILLERMO CHANG TORTOLERO

Dr. AMARO MARTIN, ISIDRO RAFAEL , Ph.D.
Miembro No Tutor

ISIDRO RAFAEL AMARO MARTIN

Firmado digitalmente por ISIDRO RAFAEL AMARO MARTIN
Nombre de reconocimiento (DN): c=EC, o=BANCO CENTRAL DEL ECUADOR, ou=ENTIDAD DE CERTIFICACION DE INFORMACION-ECIBCE, l=QUITO, serialNumber=0000228903, cn=ISIDRO RAFAEL AMARO MARTIN
Fecha: 2021.08.04 06:46:35 -05'00'

MEDINA BRITO, DAYSY MARGARITA
Secretario Ad-hoc

DAYSY MARGARITA MEDINA BRITO

Firmado digitalmente por DAYSY MARGARITA MEDINA BRITO
Fecha: 2021.08.03 16:17:29 -05'00'

Autoría

Yo, **José Luis Revelo Orellana**, con cédula de identidad **1722217591**, declaro que las ideas, juicios, valoraciones, interpretaciones, consultas bibliográficas, definiciones y conceptualizaciones expuestas en el presente trabajo; así cómo, los procedimientos y herramientas utilizadas en la investigación, son de absoluta responsabilidad de el autor del trabajo de integración curricular. Así mismo, me acojo a los reglamentos internos de la Universidad de Investigación de Tecnología Experimental Yachay.

Urcuquí, Septiembre del 2021.

José Luis Revelo Orellana
CI: 1722217591

Autorización de publicación

Yo, **José Luis Revelo Orellana**, con cédula de identidad **1722217591**, cedo a la Universidad de Tecnología Experimental Yachay, los derechos de publicación de la presente obra, sin que deba haber un reconocimiento económico por este concepto. Declaro además que el texto del presente trabajo de titulación no podrá ser cedido a ninguna empresa editorial para su publicación u otros fines, sin contar previamente con la autorización escrita de la Universidad.

Asimismo, autorizo a la Universidad que realice la digitalización y publicación de este trabajo de integración curricular en el repositorio virtual, de conformidad a lo dispuesto en el Art. 144 de la Ley Orgánica de Educación Superior.

Urcuquí, Septiembre del 2021.

José Luis Revelo Orellana
CI: 1722217591

Dedication

The present project is dedicated to my parents and sister, who have been a source of motivation providing moral and emotional support.

Acknowledgments

I am grateful to my parents and sister for their unwavering support and advice.
I appreciate Yachay Tech's efforts in general for assisting students in achieving their objectives.
To my tutor Oscar Chang, who guided and advise me through the course of this work, and his effort in various subjects that he instructed me during my university studies.

Abstract

Keywords: Artificial intelligence, Arduino, Q-agent.

Automation process (AP) is an important issue in the current digitized world and in general a good automation represents an increase in the quality of the productivity as compare with manual control. Balance is a natural capacity of humans which is related to complex operations and intelligence. Balance control present an extra challenge in automation processes due to the many variables that may be involved. This thesis present a physical dynamic balancing pole where a Reinforcement Learning (RL) agent has the capacity to explore the environment, sense its position through accelerometers and eventually learns by itself how to keep the pole balanced under noise disturbance. The agent uses RL principles to explore and learn new positions and corrections that conduce toward more significant rewards in terms of pole equilibrium. By Using a Q-matrix the agent explores future conditions and acquires policy information that makes possible to maintain stability. All the process of training and testing is done and managed entirely by an Arduino microcontroller. With the help of sensors, servomotors, wireless communications and artificial intelligence all these components merge into a system that consistently recovers equilibrium under random changes of position. The obtained results prove that through RL an agent can learn by itself to use generic sensors, actuators and solve mechanical problems even under the limitations that a microcontroller presents.

Resumen

El proceso de automatización (AP) actual es de gran importancia en el mundo digitalizado, en rasgos generales, una correcta automatización representa un aumento en la calidad de producción en comparación con el trabajo hecho a mano. El equilibrio es una capacidad natural del ser humano que está relacionada en trabajos y conducta inteligente. Equilibrarse representa un desafío adicional en los procesos de automatización, debido a la presencia de múltiples variables involucradas. Esta tesis presenta el equilibrio físico y dinámico de un poste en el que un agente mediante el uso de aprendizaje por refuerzo (RL) tiene la capacidad de explorar su entorno, detectar su posición a través de sensores como el acelerómetro y un giroscopio, finalmente, aprende por sí mismo cómo mantener un poste equilibrado bajo perturbaciones en el mundo real. El agente usa los principios de RL para explorar y aprender nuevas posiciones y correcciones que conducen a recompensas más significativas en términos de equilibrio del poste. Mediante el uso de una matriz Q, el agente explora las condiciones futuras y adquiere información de política que hace posible mantener el equilibrio. Todo el proceso de entrenamiento y pruebas se realizan y gestionan íntegramente en un microcontrolador Arduino. Con la ayuda de sensores, servomotores, comunicaciones inalámbricas e inteligencia artificial, todos estos componentes se fusionan en un sistema que recupera constantemente el equilibrio bajo cambios aleatorios de posición. Los resultados obtenidos demuestran que a través de RL un agente puede aprender por sí mismo a utilizar sensores, actuadores genéricos y resolver problemas mecánicos incluso bajo las limitaciones que presenta un microcontrolador.

Palabras Clave: Inteligencia Artificial, Arduino, Q-agent.

Contents

List of Tables

List of Figures

Chapter 1

Preliminaries

1.1 Problem statement

Automation emerged in the developing world some times replacing the work force. An example is the warehouse systems as Azadeh pointed out, one of the goals for automation is to make more efficient the repetitive work that the delivery of goods requires. commits[1].

The automation technologies are employed in automobile, delivery, metallurgical, clothing, and even food industries. The machines improved human's rate production and also the quality of the work. Still, factories continue to improve and introduce new technology to reduce their costs of production. But, some problems have to be overcame before automation is fully applied, as a rule machines have to be programmed before they can perform a predetermined sequence of steps [2][3].

Machine's activity does not present adaptation or reaction to new events that occur and that are not predictable during its normal functioning. An example of unpredictable events is the developing of changing conditions, this is the reason why automation is still of limited use in many factories, where many people is used in repetitive labors. Humans possess the natural capacity to react to new events and adapt to them, Artificial intelligence appears as a valid tool to enhance and help to build machines that respond in a similar way, that is having a good performance in realizing tasks that are not always the same in different circumstances[2]. AI may be set to perform a task in variable circumstances and may give great results and be efficient in activities currently executed only by humans.

A contemporary and relevant tool in AI is Reinforcement Learning (RL), a methodology by which agents learns by themselves efficient control policies and then use this knowledge to solve logical or mechanical problems[4].

This thesis present a dynamic balancing pole environment where a RL agent, with the capacity to read sensors and control the angles of of servomotors that drive a x,y balancing pole, learns by itself how to keep the pole balanced under noise disturbance. During the training phase, the agent uses RL principles to explore and learn new position and corrections that conduce toward more significant rewards in terms of pole equilibrium. During the training phase a Q-matrix is trained to explore for future conditions and acquire policy information that makes possible to maintain stability. In the operating phase, the compo-

nents merge into a system that consistently recovers equilibrium under random changes of position. The obtained results prove that through RL an agent can learn to solve by itself mechanical balancing problems.

To be a step closer of replacing the workforce in repetitive, dangerous tasks AI machines must be capable of replicating human behavior in several fields, first they need to able to behave appropriately like a human being, capable of understand their environment and their current state. One of the most basic and important acts humans and mobile robots is to perform a good balancing, whether we want a humanoid or a computer to replace the workers they have to be capable of performing on a similar level as the humans [5]. Balancing is an incredibly necessary essential operation to be done for loaders, robotic arms, ships, anti seismic systems or a robot itself.

Ships' movement is often reliant on careful balancing; in sea navigation, equilibrium is critical. Limiting crew operation and on-board facilities[6], for example, has an effect on the performance of surface vessels and cruise ships. The balancing mechanism may be different, but one way is to use fin stabilizers, which rotate and change the fin angle to resist ocean disturbance[7].

Balance is also present in aviation, where multiple disturbances are present. Stabilizers play an important role, minimizing disruption effects and ensuring a smoother flight[8], by dealing with three different angles known as yaw, roll, and pitch. Unmanned aerial vehicles (UAVs) are commonly used for military purposes. Missions where pilots are incompatible or minimizing the risk of human death. UAVs often deal with stabilizer systems, which correct flight angle errors and enable the plane to fly level[9].

Skyscraper construction need to balance to minimize building vibrations caused by an earthquake or air currents. While strengthening building structures increases construction costs, this strengthening can be replaced with vibration control devices implementation[10].

Balancing is a critical implementation that can affect the ability to perform a wide variety of tasks. Civil engineering, maritime, and air navigation are only a few examples of why balancing must be properly implemented in various projects in order to reduce unexpected disturbances and improve the efficiency.

How to sense or interpret a world may present another challenge. This inconvenience is easily solved thanks to sensors. Sensors allow us to identify various variables of the real world, very helpful to recognize numerous changes of the world and incorporate them into a program that uses the information. With the addition of microcontrollers and sensors is possible to build devices that perform actions in the real world, the devices may have more complexity depending on the role. Furthermore, is possible to have an AI building a robust machine that perceives the environment and follows a behavior depending on the situation. The object of this thesis has focused on the study and development of an AI in a microcontroller with the ability to analyze its state and balance it.

1.2 Justification

Existing studies of auto-balancing robots use controllers and more advanced devices to estimate the changes in their environment. The research [11] also conducts an analysis of the

issue of self-balancing robots where a center of mass is shifting evaluated in a simulation. An example of a real implementation is found in [12], where a bidep robot is built, with a very realistic structure of human legs, also with is notable the complex of building the machine of two legs.

The implementation of a simulation does not always is a successful solution and depending on the simulation method, are not take into count the different interactions with the environment as the real world, in other words, is not always very practical. Simulations may be very helpful to build new technologies as introduce balancing AI. But could be very hard to simulate all physical dynamics, as Miglino pointed out the consequence of solving in it a simulation could result in solve an unwanted problem that looks identical but is not the real one[13].

The machines do not have to be complex can have simple designs, thanks to the microcontrollers as Arduino and the multiple sensors as is showed in [14][15], is possible to build more than one specific model and balance the mass and more customizable devices that could be adapted to different situations, with the advantage of do not be so expensive easily decreasing a barrier to perform a study about different areas as AI devices, that instantly interact with the real world.

The implementation of similar devices that use Arduino has a wide area to be implemented where is possible to have an AI, opening enormous possibilities to work. The data is gathered from the world, helping to recollect data to operate. The data recall also provides a great incentive because AI requires the use of a big amount of data to perform the train part. For different projects, data recollection is one of the most difficult parts, cause the data is private or has a cost. Is planned that previous studies offer essential assistance to create new applications and variations to contribute to the creation of devices controlled by an AI inside of a microcontroller.

1.3 Contribution

The goal of this research is to develop an AI in an Arduino microcontroller that senses its surroundings through sensors and responds to natural changes. Some problems will be tackled in the development of the project and the key contributions will be solved, as defined in the following paragraph: the implementation of AI through the application and adaptation of learning techniques, with the goal of producing an optimal solution to the problem. Implementation is often achieved with the aid of common libraries and the use of existing functions, reducing the complexity of the code, helping to concentrate on the main purpose, and reducing the complexity for interested developers and researchers to understand and evaluate the steps of the software.

The work will bring other benefits in other areas, due to his enormous potential as was mentioned could be implemented to automate the process in fabrics for production. But could bring new possibilities allowing to implement the same guidelines, intelligence, and principle of balancing a mass. An example of increasing the help in the rehabilitation process, imagine a therapist does not have the strength to stand up to a patient that had lost mobility and needs support if the nurse is not capable, is possible to develop a

device with the same objective of bringing stability during the rehabilitation. Or develop a robotic prosthesis for a person with a physical disability, the prosthesis will adapt to the fickle variations of the real world, giving an effective response increasing the quality of life of the individual.

Include an AI inside a microcontroller, demonstrating the capabilities of these computers, and how extensive is their use. Microcontrollers present some limitations against other devices as laptops or desk CPUs. But, they can perform a complex task with its disadvantages, also highlighting the mains characteristics as accessible price, reduce size, and different peripherals to perform a large range of uses.

1.4 Work Organization

- **Chapter 2- Objectives** The target objectives to develop and solve after finish the investigation.
- **Chapter 3 - Related Work** presents a review of works mainly related to the main principles and objectives of Arduino and AI
- **Chapter 4 - Theoretical Background** presents background of theoretical details and key concepts.
- **Chapter 5 - Technical Framework** Presents and describes the hardware chosen elements.
- **Chapter 6 - Methodology** Describe inconveniences of the problem, but mainly how the whole prototype works, the hardware, and software processes.
- **Chapter 7 - Results and Discussion** Presents agent's after-training results, with a respective analysis.
- **Chapter 8 - Conclusions** In this last section is conclude some recommendations, considerations and future improves.

Chapter 2

Objectives

2.1 General objective

- Implement a dispositive that balance a mass in the extreme of a pole through an AI implemented in an Arduino unit.

2.2 Specific Objectives

- To identify and describe the conflicts or limitations of implement an AI in an Arduino board.
- To study the Q-table technique and its adaptability in real environments.
- Implement an algorithm to send the current state of the accelerometer through wireless communication.
- To expose the compatibility of agents, reinforcement learning when are implemented to robots in real environments.
- To demonstrate the capacity and usefulness of Arduino boards and the access to the multiple customizable peripherals, that has.

Chapter 3

Related work

The following chapter examines and evaluates works that are relevant to the planted objectives and share common themes, strategies, and method elaboration. The literature is divided into three parts based on the subject relationship: first, Arduino-controlled systems, second, AI in microcontrollers, and third, Reinforcement Learning in Real Environments.

3.1 Arduino fully managed systems

Due to their low and medium cost, Arduino-based projects can be produced or recreated. Due to easy access to a variety of hardware resources, it is simple to execute multi-purpose projects. A mobile robotic platform is presented in [16]. In the work is claimed that robots must be as inexpensive as possible in order for students and researchers to conduct real-world experiments. When working with monitoring systems or robots, it's also important to get the power supply right, particularly during the experimentation and development phases. A proper power supply is another good feature of working with Arduino boards. Wireless communication and sensor devices are also listed, if more mobile robots are to be included in an environment, these two elements are necessary. To process knowledge about the current state of its world and to carry out reliable and organized work.

The work in [17] demonstrates the versatility of Arduino when used in an IoT project with various sensors and modules. The aim is to use a user interface to manage and track the sensors, and to report the collected data through a wireless link using a Wi-Fi module. The work also mentions the benefit of Arduino prototypes in terms of how quickly they can be expanded. These microcontrollers have a variety of options for adding more sensors or other electronic resources to help with task execution. Another recommendation is to track systems and electronic connections, stating that the device requires a proper power supply to avoid failed Arduino executions or harm[17].

The Arduino microcontroller is used in [18] to build a Supervisory Control and Data Acquisition (SCADA) system, which will be used to boost Smart Grids (SGs). Interoper-

ability due to the heterogeneity of devices that adhere to SGs, as well as high costs due to the work itself, where devices must be automated and controlled continuously in order to compete and act in industrial environments, are some of the challenges that SGs present. The Arduino performs admirably in this study when it comes to handling and collecting data from various sensors. Due to the low cost of entry and the simplicity of adding more sensors to the board, Arduino systems can easily expand their use by calculating more parameters. Again, adding more sensors does not inherently mean more work when it comes to coding. The high configuration possibilities are listed again, as well as how learning to program new things is not difficult; these advantages stem from the various libraries and a fast learning rate for new users.

In [19] also supports the Arduino board's data collection, tracking, and scalability projection capabilities. The study once again demonstrates how it is possible to create devices that capture real-time data, reducing the difficulty of collecting data for studies. This work, in particular, behaves similarly to the prototype of this thesis; they use an accelerometer to calculate the vibration of certain devices, and the data is transmitted using an Arduino through a wireless module. The paper provides technical evidence that Arduino can act as a real-time wireless sensor network. Also establishes confidence in the information's ability to perform data analysis; in this paper, the information is used to create predictive models, with excellent results.

The majority of the works discuss the low-cost advantages of Arduino projects, as well as how to adapt or include more electronic devices to create more robust devices with no issues. Furthermore, the majority of the works incorporate two elements: automation and real-time operations. Arduino can conduct real-time operations without the use of simulations, concentrating on the problems that are important to the target, enabling it to detect planning mistakes or missed steps in the early stages of development. This has been a big help to our research, both technically and theoretically, showing that Arduino is a wonderful tool to use in this thesis.

3.2 Artificial intelligence in microcontrollers

In [20], a self-driving car is demonstrated using a Raspberry Pi and a Convolutional Neural Network. Car actions, on the other hand, are directly carried out by the Arduino board, which receives an order and drives the car in a specific direction. The author concludes that the car's design and testing were successful, and it can operate on both straight and curved tracks. Also shown in [15] is a mobile robot for educational purposes based on Android and Arduino. The robot has a light sensor, GPS, camera, accelerometer, Bluetooth, and Wi-Fi, and can accept commands from a smartphone via an internet connection. The author claims that software alteration can be easily applied for future projects because the mobile platform was a success when managed by an Android.

In the third study [21], image processing is used to monitor the movements of a human arm and reproduce this action in a robotic arm controlled by an Arduino. The author claims that an Arduino board can monitor the robot's behavior, but that there are many

ways to accomplish the same task.
It is clear from the previous work that the Arduino will perform actions with good results if the instructions come from a logical process of a efficient AI or a human. The argument is that with the use of a microcontroller, the majority of the works present no hardware problems. As a consequence, with the right instructions, Arduino will perform a variety of tasks. The intelligence of data analysis is executed outside the microcontroller in all previous projects, but as the following projects illustrate, it is also possible to code the intelligence or incorporate learning techniques in the Arduino itself, without having to raise costs or resources.

Artificial Intelligence has been shown to achieve better results than conventional approaches in a number of projects; these works involve intelligent agents that perform several tasks in pursuit of optimum results. [22] and [23] study intelligent agents are capable of considering timing, amount of water, and properly implementing them in what they detail as Spatio-temporal variations of the soil–plant–atmosphere system, summarizing agents present a more realistic behavior, determining behaviors in the surrounding world. Getting impressive results in terms of water efficiency and precision irrigation.

As agents work with microcontrollers like Arduino, they may create further applications. For example, [24] shows an intelligent traffic light control device that uses Arduino sensors to detect the color of traffic lights, as well as the tone, distance, and motion of vehicles in the streets where it is installed.
After the process is initiated, the agent collects all sensor data and analyzes it to create a light sequence that reduces traffic as much as possible. The entire process is coded and executed in Arduino, demonstrating how this platform can perform complex tasks using machine learning techniques.

Since the works [22], [24], [23] use Arduino to carry out the whole operation, these project reinforces the concept that Arduino hardware and software are plenty capable, and assisting this study in realizing that AI can be implemented in a microcontroller. Of course, Arduino has limitations, but the previous works only require a little extra effort to comprehend the obstacles, adapt the prototype, and include RL.

3.3 Reinforcement learning in Real environments

The aim of developing robots is to allow them to participate in a real-world environment where unexpected events can occur. The goal is to make machines more effective in performing various tasks in a manner that is equivalent to, if not better than, that of a person. To accomplish this action in computers, a variety of techniques are used.
Three classifications of methods are given in [25], as well is mentioned how computers can learn to deal with such tasks. All three methods are still commonly used in various fields, but each has its own set of characteristics.
Since it operates in more structured and stable environments, direct programming is defined as the lowest method. Is more akin to a programming method than it is to learning, in this method the positions and actions are pre-programmed. The second method is imitation learning, which is more comparable to a learning strategy and can be applied in three different ways:

- Inesthetic is the motion of manual processes is recorded.
- In teleoperation an instructor is situated in a remote area.
- In observational learning, a practitioner provides sample movements that are recorded using cameras or sensors.

The problem with this approach is that it requires a specialist to perform a perfect presentation of how to complete the job. In comparison to the other methods, Reinforcement Learning(RL) is defined as a trial-and-error method that explores the environment and the robot's body. According to the paper, RL has three benefits over the other two approaches. Learning new tasks, optimizing efficiency, and adapting to new situations.

RL has been used in real-world settings to solve physical problems in various experiments, but simulations can also be used in training. According to study [26] , depending on the context, the training will present undesirable driving behaviors that cause harm to the environment. As a result, depending on the case, a simulation or real-world environment may be used to train a computer. The problem with transitioning from a virtual to a real-world environment is that it takes extra time to apply the learned skills to a different scenario than the one in which they were educated. The research from [13] demonstrates how simulations can solve problems that appear to be close to the original target but are not; in other words, the simulation environment can lack realistic behavior.

However, conducting training in the real world poses a challenge; as mentioned in [27], samples can be extremely costly. This complication about the cost of collecting samples can be dependent on the tools used. As described in section 3.1, Arduino is widely used to continuously collect data, and in some experiments that perform the training part in the Arduino were mentioned as well, assisting in overcoming the barrier of training an intelligence in the real world.

It is also claimed in work 5 that using an RL in the real world could harm a sensor with delays and that certain electric tools from the hardware could not function immediately. Multiple works [22], [24] [23], [15] [20], [19], [18], [17], [16] use similar sensors, wireless communication modules, and servomotors in their works, and these works do not report transmission or execution problems with the used elements. It is also possible to monitor the work from the integrated elements using programmed delays in order to avoid jumping from one step to the next without first acting. To prevent similar problems, delays are used before making the second step of a servomotor without finishing the transfer of a previous order. As a result, the aforementioned works, as well as the use of delays, facilitate the selection of the tools needed to build the prototype and avoid unexpected behavior.

The work from [28] discusses real robotic platforms, with the aim of demonstrating how real robotic platforms can perform physical tasks in 3D real environments. It is also claimed that RL is an effective training method for these systems. The study creates and trains a robot to open a door from the scratch, which is a realistic application that requires complex contact dynamics to imitate human physical abilities.

Work [29] uses a free-reward RL algorithm to teach a robot how to travel properly within its structure and navigate. Sensory inputs, 12 degrees of freedom, and a quadruped robot After 20 hours of preparation, the machine had mastered a variety of locomotion gaits. After a short period of time, the RL methods yielded results. This research showed how RL can be used to teach an AI how to control its movements in the real world without wasting too much time in the training process.

Finally, since the approach is based on trial and error, the robot can be damaged, although this is not always the case. Work [30] includes an analysis of a biped robot that is also trained with RL in the real world and uses what they call a safe-RL. The technique yields positive results, and the application decreases robot drop. As a result, risky conditions that might damage the computer or device can be avoided. Prior to starting the train of a machine, it is important to conduct previous analyses and count it to minimize risky situations.

Chapter 4

Theoretical Background

4.1 Supervised learning Problems

It's common to use Supervised Learning to teach Machine Learning applications (SL). A classic way to apply this technique is to use statistical learners from Artificial Intelligence such as Recurrent Neural Networks(RNN), Neural Networks(NN), or Convolutional Neural Networks(CNN)[31].
Feeding a NN model with unique input and knowing what the model should produce as an output action is required when computing a NN model. To use gradients as a backpropagation model to train the model to generate the predicted results. However, using SL to solve problems or carry out activities, such as reacting to complex environments through video games, has some drawbacks[32].

There are two slight disadvantages of using SL for applications such as playing video games. Labeled cases, as Xiaojin points out, have three major drawbacks that make them more difficult to use. The first issue is our struggle to locate a specific data set for our application. The second issue with labeled data is the high cost of accessing the requisite information; genuinely excellent labeled data can be kept private on a number of sites dedicated to selling it. The third issue is that producing well-labeled data, which will necessitate expert assistance, will take a long time if we want to build our dataset[33].
A more complicated aspect with using SL is that with this procedure, a model can only learn to imitate the expert's actions rather than learning how to work. As a result, the NN's acts will not outperform the success of the player; in other words, the NN will not be a better player than the player who knew.

4.2 Reinforcement Learning (RL)

The aim of reinforcement learning is to teach a computational agent how to carry out an action on its own in a discrete, finite world[34]. This method employs a unique methodology that does not rely on labeled data or supervision. It has three main characteristics: no supervisor, trial-and-error, and a reward signal as a guide [35], the RL method has no

immediate reward, and actions have an effect on the incoming data and the next situation, RL working process is usually be represented as Figure 4.1, where the environment is unknown, and the agent receives a reward at a given state, the agent will interact by actions with its environment trying to establish a path to solve a task, the best possible, learning during the process very similar to a puzzle.

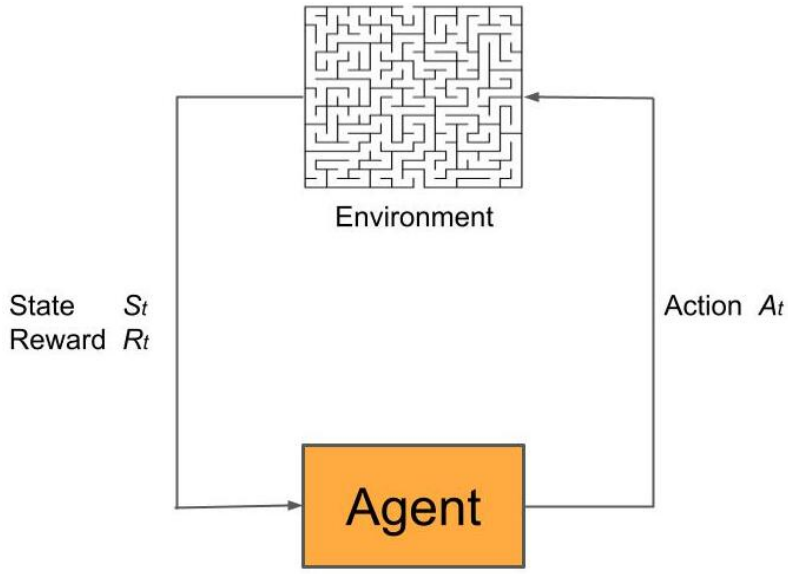


Figure 4.1: Reinforcement learning process.

4.2.1 Reward

Agent mission is to maximize a real-valued reward signal[36]. A reward R_t is a scalar feedback signal, indicates the agent's actions performed at the step t.

Sequential Decision Making

- The aim is to choose the best course of action in order to maximize total future reward.
- The acts taken can have short or long-term implications.
- The immediate reward is not always the best option; long-term actions can yield a greater reward, greedy strategy is not the right course of action.

4.2.2 Agent

The agent is entirely software-based, and it examines the current state of its environment O_t, then determines which action A_t to take, and receives feedback in the form of rewards or penalties R_t.

The main goal is to teach the agent how to make better decisions by teaching it how to learn a task after several repetitions with its environment and getting as much reward as possible[37]. The majority of observations are partial; they do not include all aspects of the climate, only the essential facts.
After calculating several episodes, the Agent can increase the possibility of actions that provide a higher reward while reducing or filtering actions that provide a lower reward.

4.2.3 Environment

The environment is the world in which the agent interacts; an agent interacts with and manipulates the environment through its actions, which cause the environment to transit to different states, with different states at a given time(t), step, or moment. [38].

$$H_t = A_1, O_1, R_1, ..., A_t, O_t, R_t \tag{4.1}$$

History may be used to define the whole process; in equation 4.1, histories are made up of various acts, observations, and rewards.

4.2.4 State

History defines the whole process; in equation 3, histories are made up of various acts, observations, and rewards [38]. From equation 4.2 state is a function of history.

$$S_t = f(H_t) \tag{4.2}$$

If the environment state H_t is Markov, the whole history is the same as the actual state.

$$H_{1:t} \rightarrow S_t \rightarrow H_{t+1:\infty} \tag{4.3}$$

- Policy: the agent's actions feature, which helps in the selection of an action.
- Value Function: how good is each condition and/or how well the behavior is doing in a given situation.
- Model: agent's representation of the environment.

4.2.5 Policy

A Policy is the logic that determines if a given state can perform an output operation. The purpose of RL is to find an optimal control policy that can be used as a strategy to optimize a criterion in certain processes[36] .
A policy is a mechanism that helps in the mapping of circumstances or states to behavior[35]. There are two types of policies: deterministic4.4 and stochastic4.5.

$$a = \pi(s) \tag{4.4}$$

The policy requires the agent to be in a state s, which is a condition in which the agent would make a decision.

$$\pi(a|s) = \mathbb{P}[A = a|S = s] \tag{4.5}$$

According to Sehnke, this approach searches in the policy space and does not derive its behavior or actions from a value function [39]. In high-dimensional problems with continuous states and behavior, MF methods can also achieve an optimization with a good performance.

In real-world or virtual scenarios, the Policy used give an output that represents the probability of possible actions. This ensures that the policy choices avoid predictable and linear outcomes. The main purpose of using probabilities to make a decision, on the other hand, is to encourage the agent to begin a random exploration of the world in order to enhance behavior and rewards.

4.2.6 Value Function

Used to determine how good or bad the current state is, allowing the agent to optimize long-term results[38]. The value function for a policy tell us the expected reward to get into the future. The expectation of the reward of time t plus the reward of the next step and so on4.6.

$$v_\pi(s) = \mathbb{E}_t[R_t + \gamma^1 R_{t+1} + \gamma^2 R_{t+2} + ...|S_t = s] \tag{4.6}$$

4.2.7 Model

Although a model is not the same as the environment, it can be used to estimate or forecast what the environment will do next. The model aids in the development of a strategy for achieving a mission. P predicts the next state in the model 4.7, while (R) predicts the next (immediate) reward4.8.

$$P^a_{ss'} = \mathbb{P}[S_{t+1} = s'|S_t = s, A_t = a] \tag{4.7}$$

$$R^a_s = \mathbb{E}[R_{t+1}|S_t = s, A_t = a] \tag{4.8}$$

4.3 Model-based

A Model-Based(MB) is a convenient option for maximizing functionality in various problems, such as complex control systems, to solve a complex task through abstraction, as Kramer said [40].

4.4 Model-free

In contrast to MB, a Model-Free(MF) approach avoids the construction of a model and instead learns directly from experiences with the environment[41], without any prior guidance or knowledge about the problem.

4.5 Exploration exploitation

RL process has issues in not so simple environments. The process is trial-and-error learning, exploring the environment could be performed greedily, but will not learn or explore new possibilities, interfering to discover an optimal policy. The solves this issue with a random exploration, to select an action the agent chooses one option randomly not the optimal option. This random exploration method allows us to eventually discover more possible moves that could have better rewards than a greedy strategy would have found. But still, the objective is agent becomes more confident and experienced. Exploration consists of finding more information about the environment, exploitation exploits known information maximizing the reward.

4.6 Markov Process

Most of the problems can be formalized as Markov Decision Process (MDP). Markov mentions that a current state State (S) contains all useful information from history.

$$\mathbb{P}[S_{t+1}|S_t] = \mathbb{P}[S_{t+1}|S_1, ..., S_t] \tag{4.9}$$

Equation4.9 implicates that the future is independent of the past given the present actions[42]. The probability of the next state is conditioned by the actual state which contains all relevant information from history.

4.6.1 State Transition

A Markov state s and successor state s', the state transition probability is defined by equation 4.10

$$P_{ss'} = P[S_{t+1} = s'|S_t = s] \tag{4.10}$$

the state transition matrix P, defines P which is the transition probabilities from all states s to all successor states s'.

$$P = \textbf{\textit{from}} \begin{bmatrix} P_{11} & \dots & P_{1n} \\ \vdots & & \\ \vdots P_{n1} & \dots & P_{nn} \end{bmatrix} \tag{4.11}$$

A Markov process(or Markov Chain) is a tuple $(S, A, P, R, \gamma)$where S is a (finite) set of states , A a finite set of actions, P is a state transition probability matrix, reward function R from equation 4.8, and $\gamma \epsilon [0, 1]$ as a discount factor[36].

4.6.2 Bellman equation

Allows to choosing the action with the highest value function. In RL the Q-function $Q(s,a)$ predicts the best action a in state s in order to maximize a cumulative reward. This Q-function is estimated using Q-learning, iterative updates $Q(s,a)$using the Bellman equation.

$$Q(s,a) = r + \gamma max_{a'} Q(s^{'}, a^{'}) \tag{4.12}$$

Bellman equation 4.12, states that the maximum reward for a given state is equal to the immediate reward (r) plus maximum future reward for the next state$(max_{a'} Q(s^{'}, a^{'}))$[37]. Due to the Bellman equation is non-linear, multiple iterative solutions methods are used as Value Iteration, Policy iteration. or Q-learning.

4.6.3 On and off policy

- On policy learning, an agent learns about the policy π from experience sampled from π, in other words, learn when works on the job.
- Off-policy learning the agent learns about the policy π from experience.

$$\pi(s) = \underset{a \epsilon A}{argmax} Q(s,a) \tag{4.13}$$

4.7 Q-learning

Is a off-policy learning of action values $Q(s,a)$, it works selecting a next action which is chosen using behavior policy $A_{t+1} \sim \mu(\cdot|S_t)$, but is considered an alternative successor $A'\pi(\cdot|S_t)$. Then is updated $Q(S_t, A_t)$ towards the value of alternative action Figure4.2.

$$Q(S,A) \leftarrow Q(S,A) + \alpha\big(R + \gamma \underset{a'}{max} Q(S', a') - Q(S_t, A)\big) \tag{4.14}$$

but address also has present some issues: improving the player or finding labeled data. The main difference RL between and SL, is that we don't have a label target, and since we don't have a data set, we don't have a direct instruction for what an agent should answer to for a given state.

4.7.1 Q-table

The Q-table or Q-matrix is a look-up table, a matrix that helps to store and calculate the maximum expected future reward an agent will get given the pair *Action* and *State*. Columns are the possible actions, rows represent the possible states, each value in the Q-table is a Q-value [43]. The table will be updated, its values will increase or decrease iterative during the Q-learning process. Figure 4.3 summarizes the information's flow in the Q-table during the Q-learning the inputs are*States* and *Actions*, to produce a Q-value to continue updating the table.

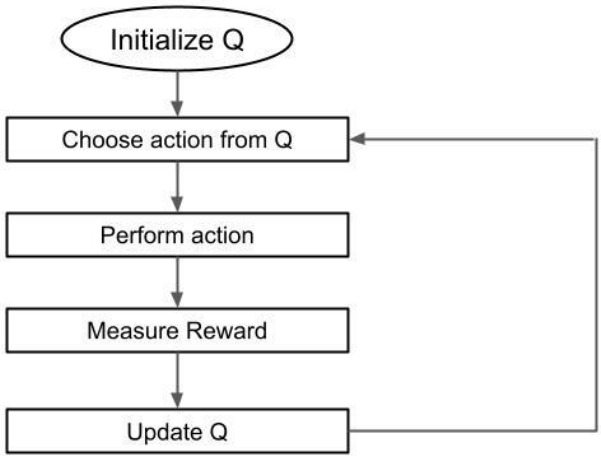


Figure 4.2: Q-learning process.

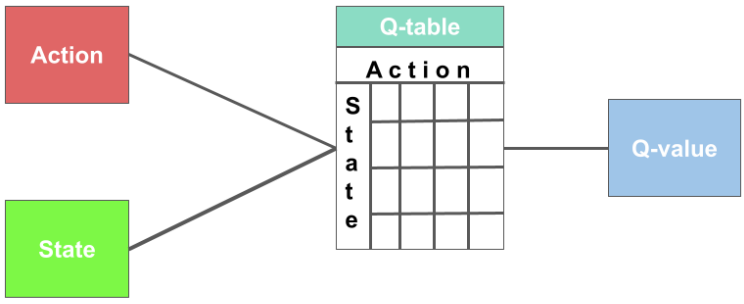


Figure 4.3: Q-table implementation.

Chapter 5

Technical Framework

5.1 Microcontroller

The microcontroller is an integrated circuit that contains all of the components used to carry out different operations, tasks, or routines [44]. The microcontroller commonly used on the Arduino boards has three types of memory:

- The Static Random Access Memory (SRAM) is usually used for local variables, and where the program creates and manages the variables local during the execution.
- The Electrically Erasable Programmable Read-Only Memory (EEPROM) is the space of memory to save values after the board is turned off, a non-volatile memory.
- The program or sketch is stored in the Flash memory, also non-volatile memory.

Figure 5.1: Microcontroller used in the Arduino boards.

It contains a central processing unit (CPU), a memory of 32 KB (0.5 KB used by bootloader) for storing a program and the data it generates, it has 2KB of SRAM, 1 KB of EEPROM, as well as input/output peripherals[45]. Microcontrollers have all three of a computer's main functional components. Microcontrollers, on the other hand, are mostly designed to execute a predefined set of instructions, whereas computers are multipurpose. This restriction is due to its advantages, such as how easily it can be implemented in the real world, and, of course, the small amount of space it requires. Microcontrollers are described as a much simpler, scaled-down computer; another advantage is that they are

less expensive than a desktop machine; but, as previously stated, they are very useful for performing a single task [46].

5.2 Arduino

The Arduino boards are electronic boards based on the microcontroller from Figure 5.1 the ATmega328P, and being considered as "small" computers that can run programs, most often used to create electronic projects. The board allowed the development of a wide range of projects due to the ease of use of input devices, sensors, and other electronic peripherals that could be included[47].

Due to its low cost and energy efficiency, it is used for the research and development of projects in various fields of study. Arduino is widely used in universities in a variety of courses[44]. Because of its simplicity and success, it has received a lot of support from community users. All of the funding is also for its open-source hardware and software, which comes with several useful features as libraries, and code examples[48]. Even some of these features are directly accessible in the Arduino Development Environment, allowing the programmer to concentrate solely on the task at hand.

5.2.1 Arduino IDE

The Arduino IDE is the most popular environment for programming in this microcontroller, and since most microcontrollers use a variant of the C programming language, it provides flexibility in program writing and hardware control for the programmer[48]. Arduino offers a variety of board designs, as well as a variety of items depending on the situation. Some boards are best for learning, while others are better for advanced features or specific areas as the Internet of Things. Existing more than one option to acquire.

5.2.2 Arduino Uno

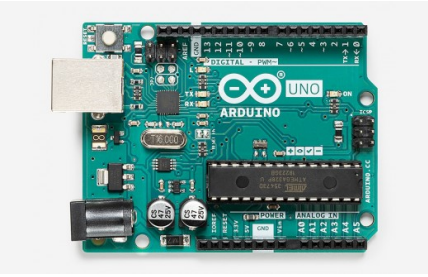


Figure 5.2: Arduino uno version.

The Arduino Uno from Figure 5.2, is the standard and basic version of the boards, with 25g of weight. It has a 5V operating voltage, 6 analog input pins, 14 digital I/O pins, 6 PWM Digital I/O pins, a power jack, and a reset button[49].

5.2.3 Arduino Nano

The Arduino Nano from Figure 5.3is a smaller version, with 7g of weight, of the Arduino Uno that is also based on the ATmega328. The key distinction is 8 analog IN pins, 22 digital I/O pins, and the Nano edition lacks a DC power jack and the USB port is a Mini-B USB port[49].

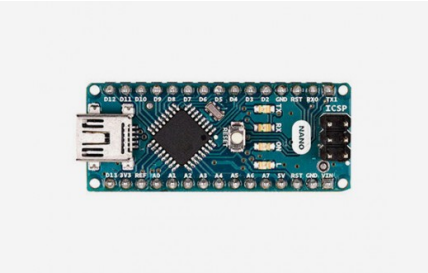


Figure 5.3: Arrduino nano version.

5.3 MPU6050 sensor

An inertial measurement unit (IMU) is incorporated in a wide range of electronic devices such as drones, robots, and even cellphones, and has a valuable peripheral to use in balance projects. An IMU is a movement sensor that is widely used in navigation, goniometry, stabilization, and robotics projects. In simple terms, it aids in the measurement of kinematic parameters as the system's speed, orientation, and acceleration. The MPU6050 is a popular Arduino IMU that combines an accelerometer and a gyroscope[50].

5.3.1 Accelerometer

A Micro Electro Mechanical Systems (MEMS) works based on Newton's second law, which states that the acceleration of an object with constant mass is proportional to the net force acting on the object. MEMS works as a mass-spring device to measure the acceleration of a structure in the three axes of X, Y, and Z. The sensor is continuously measuring gravity's acceleration, and this particular unit has a scale rank of 2g, 4g, 8g, and 16g to capture the smallest and fastest variations accurately[51] [52].

5.4 Gyroscope

In the simplest form, angular velocity is the change of angular displacement over time, or how quickly an object spins around its axis. Via the Coriolis effect, gyroscopes use a MEMS to calculate angular velocity. The used accelerometer and gyroscope, can measure the thre axis X, Y, and Z. The gyroscope is constantly measuring the angular velocity. This particular IMU's gyroscope has a scale rank of 250 Grad/Seg, 500Grad/Seg, 1000Grad/Seg.

MPU6050's IMU has six degrees of freedom (DOF). A degree of freedom is defined as the number of logically independent values that can be adjusted [53]. The accelerometer

has three degrees of freedom (DOF), and the gyroscope has three degrees of freedom (DOF). The IMU values are used to calculate the inclination and rotation angle; essential information to create an auto-balancing Arduino system, to estimate the current state of its environment[51] [52].

5.5 Servomotor

An electronic motor having three cables Voltage Common Collector(VCC) for power input, ground (GND), and signal. It has Metal gears for better performance, having more strength than other classic servomotors as Sg90. It produces a rotation shift from 0 degrees to 180 degrees. The MG90S servo motor has a torque of 2.20 Kg and a speed of 0.1 seconds at 60 degrees at 4.8V of power supply, 0.08 seconds at 60 degrees at 6V[54].

A servo motor is a very useful tool for performing acts derived from the logic computed in the Arduino, acting as a stabilizer. the current state of its environment.

5.6 nRF24L01 Transceiver

This module allows wireless communication between Arduino nodes, in a simultaneous communication the module works with a limit capacity of six nodes. The range of reach is between 2 and 3 meters more elaborated version of the module allows increasing the range at most 100m. It operates at the frequency of 2400 to 2525 MHz, which constitutes a frequency range for free use, having a total of 125 channels with a space of 1MHz. This module has a transmission speed of 250 Kbps, 1 Mbps, and 2 Mbps. An error correction and data forwarding are also included, minimizing code complexity. The module works with a power supply of 3.3V, from the Arduino output pin[55].

Wireless communication brings several options to be added in projects, upgrading systems to perform wireless control, and this communication is possible between two controllers.

5.7 Electrical Power Supply

Most Arduino boards have an operating voltage of 5V, it has various powering inputs the USB port, a Jack socket(not in the Arduino Nano), the Vin socket, and a 5V socket. The recommended input voltage is between 7-12V and a limit voltage of 6-20V[49].

In the prototype, the Arduino Uno board is powered by a USB connector to constantly save the information in the computer. For the Arduino Nano, the power supply is a 9V battery to avoid power disconnections while the pole moves.

Chapter 6

Methodology

This chapter describes the proposed system, which learns to detect a properly stable position. The construction of the balancing prototype has three main points to be described, divided into the following sections: General description of the system. A detailed description of the transmitter device. A detailed description of the receiver device. Finally an algorithm description of the training process.

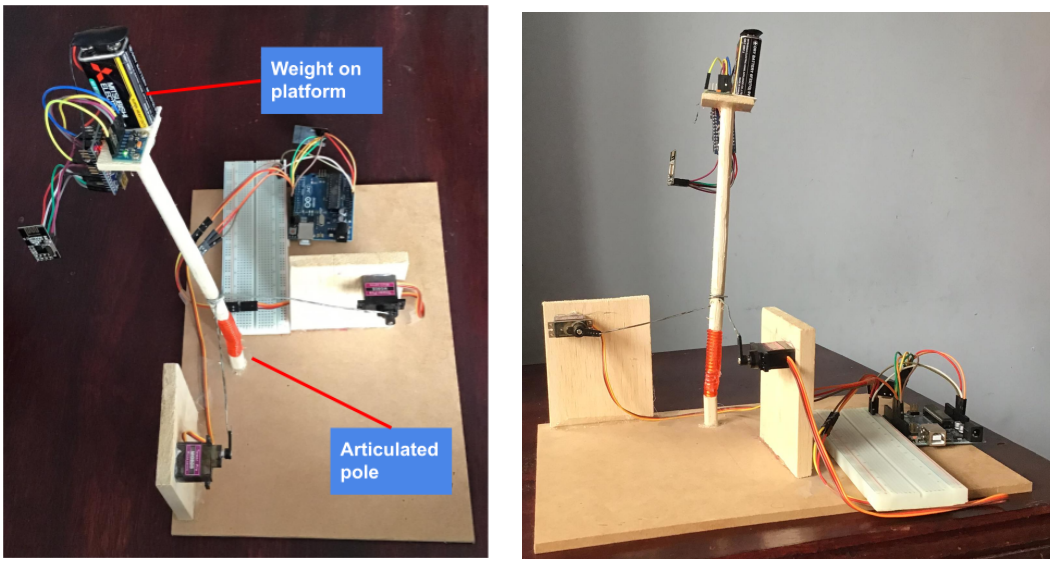


Figure 6.1: A view of the structure to balance.

6.1 Phases of Problem Solving

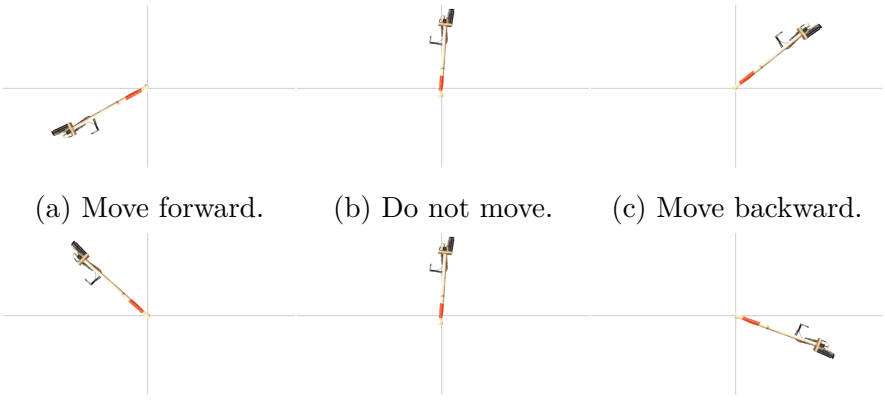
6.1.1 Description of the problem

The structure to balance consists in a pole at the top it has a platform where objects can be placed. The pole can not stand up straight, the problem is a flexible component in the base, the pole is connected to this component which allows the pole to constantly fall in

any direction. To control the pole direction two servo motors are connected using a cable one servo motor move the pole up and down, the other one moves the pole from left to right, the structure to balance and the prototype are in Figure 6.1.

6.1.2 Analysis of the problem

The first issue is to translate the environment in terms that is possible to work in programming, select the correct data type attempting to not excess the memory capacity of the Arduino.
The speed execution of the code happens out stands the speed movement of the servo motors, to train an agent and measure a reward after is necessary to measure after a change or move is done, the Arduino board can execute multiple moves virtually before a servo motor actually has finished the first move. In other words the measurement have mistakes, due to virtually is supposes to have performed a move but is not the real.



(a) Move forward. (b) Do not move. (c) Move backward.

(d) Move left. (e) Do not move. (f) Move right.

Figure 6.2: Graphical representation of the possible actions that two servo motors can perform.

6.1.3 Implementation

The condition and possible behavior of the agent are represented by a Q-matrix, also known as a Q-table; the dimensions of this Q-matrix for the project are 9x2, represented in Figure 6.3 the number of columns represents the number of possible actions from which the pole will pass, while the number of files represents the inclination states in which the platform will be.
The servo motors in use operate in degrees from $0° - 180°$; one pushes the pole forward Figure 6.2a or backward Figure 6.2c, while the other moves the pole left Figure 6.2d, or right Figure 6.2f, but if the system detects that is in balance both servo motor will not move pole Figure 6.2b and 6.2e. The first column depicts the action of decreasing the current angle at which the servo motor is located, while the second column depicts the action of raising the current angle at which the servo motor is located. The servo motor

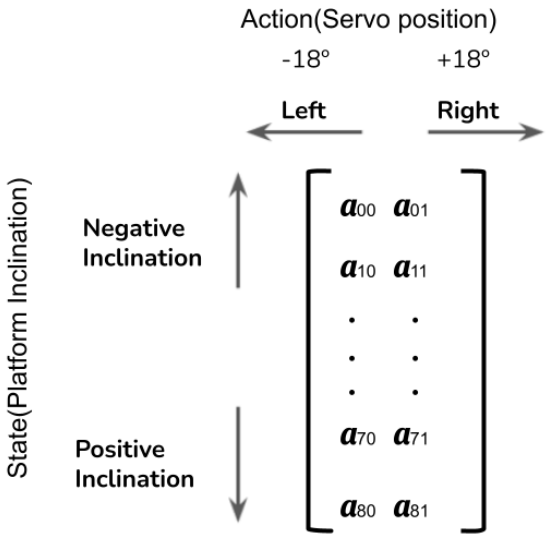


Figure 6.3: Q-Matrix, states (rows) and actions (columns).

has 144° of inclination since the eighth row is 8, which is the highest value that can be found in this matrix beginning at state 0 from 0° in the servo motor.

6.2 System description

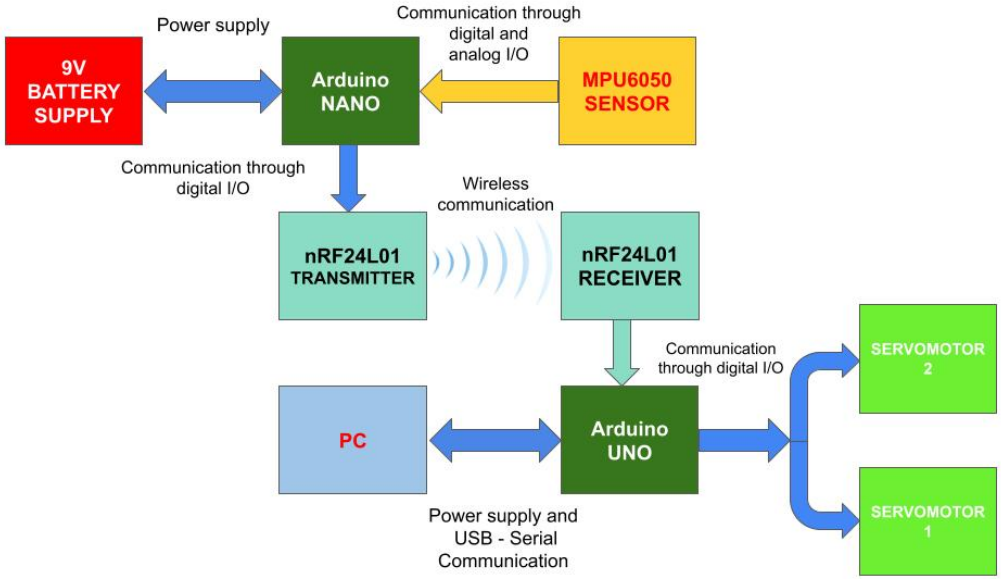
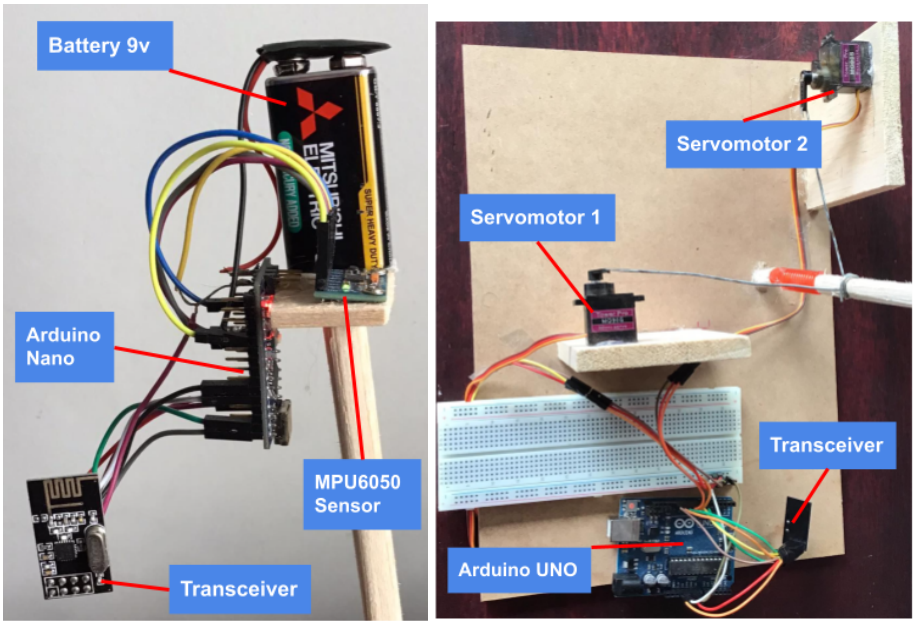


Figure 6.4: Block diagram Prototype of the whole process.

The prototype design is structured by two devices that work together to balance the pole Figure 6.4, details with which component is connected and how communicates, information flows from upper components to which are below, the device A and B have wireless

communication, the implementation and a deeper description of the internal process of each device will be described in the following sections. Device A due to its structure could be reused in more areas to estimate more precise position changes. The device B does not have the same reuse capacity, it has to be modified depending of the balance structure.

Device A



(a) Device A. (b) Device B.

Figure 6.5: Two pictures from a different angle, of each device and its elements.

Device A task is recollect and notify position changes of the platform in which is placed. In Figure 6.5a shows the electronic elements used as the Arduino Nano, the accelerometer and gyroscope that are in the MPU6050 sensor, and the nRF24L01 RF transceiver. The device A perform the flow process in the Figure 6.6.

Device B

This device is managed by Arduino Uno, and will execute the process from Figure 6.7 and after the Train process finishes will execute the steps from the Figure 6.8. Device A in Figure 6.5b shows the two servo motors, that manages the positions of the pole, also has the nRF24L01 RF transceiver for a wireless communication.

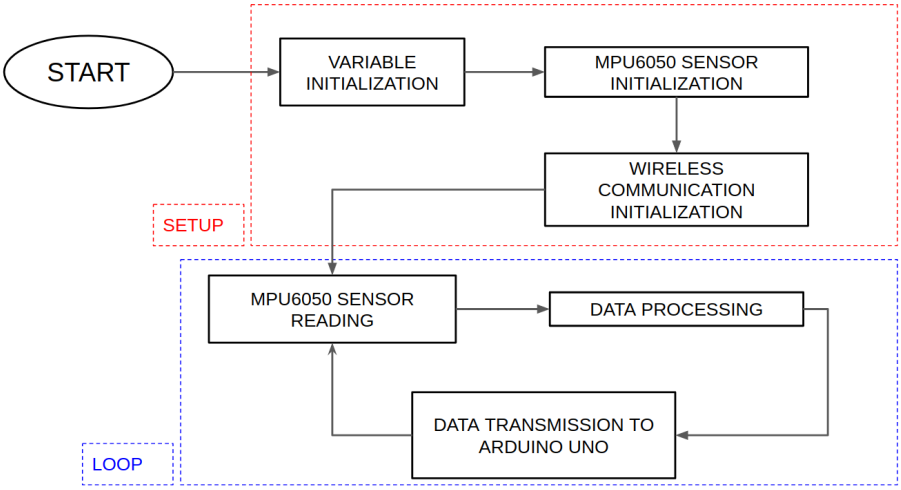


Figure 6.6: Flow process of the Arduino NANO.

6.3 Device A process

Setup and Loop functions

The setup and loop functions are the two main functions of most Arduino programs. After uploading the program to an Arduino microcontroller, the setup function is the first to run. The loop function is a loop that repeats the actions inside the microcontroller until the microcontroller is reset or the power supply is disconnected. Both of these functions are required in Arduino programs.

6.3.1 Sense and Transmission Flow Process

The Arduino Nano is used to carry out the entire process depicted in Figure 6.6. First is the **setup** function,where all variables of the accelerometer and gyroscope in the three-axis x, y, and z are initialized in the **variable initialization** step, also the data rate in symbols per second (baud), each baud has several bits per second(bps) in which the Arduino will communicate during the transmission data is setup. **Sensor initialization** begins the MPU6050 functionality after ensuring that it is operational. The Power Amplifier level, the channel between 2.400 and 2.524 GHz, the data rate, and the address of the receiver to which we will send data are all set up in the **wireless communication initialization** step.

The loop, as seen in Figure 6.6, is the second part that constantly reads position changes. MPU6050 is constantly read during **sensor reading**, and the data obtained is then sent to **data processing**, where all information is processed in degrees and g (terms of standard gravity). After that, **data transmission** begins, which reports the changes in the pole's extreme to the Arduino UNO.

6.4 Device B process

6.4.1 Train Flow Process

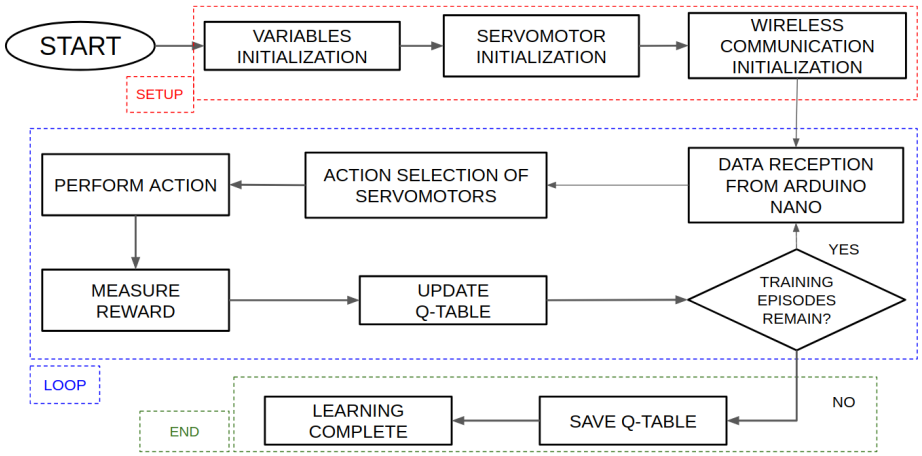


Figure 6.7: Flow process of the Arduino UNO in the train Phase.

Setup in Figure 6.7 and Figure 6.6 are very similar, except for the **variables initialization** step with the R and Q matrices for the training process. Besides, the same parameters as in Figure 6.6 must be used for **wireless communication initialization**.

In the **loop** section, the **data reception** is emitted from the accelerometer in the Arduino Nano. **Action Selection** consist of a random selection of the available options from the servo motors. Now **Perform action** executes the selected action from the previous step. Next step a reward is calculated from the previous actions, then updating the Q-table. All the previous steps are repeated until a fixed number of iterations. The **end** section has only one step, saving the experience that is in the Q-table earned in the **loop** section for forthcoming testings.

6.4.2 Testing Flow process

Setup in Figure 6.7 and Figure 6.8 the same steps. Then in the loop function the **data reception** is received from the Arduino Nano then **check the Q-table** select the best option and perform the action selected in the servo motors establishing the balance and repeating the process from the **data reception**.

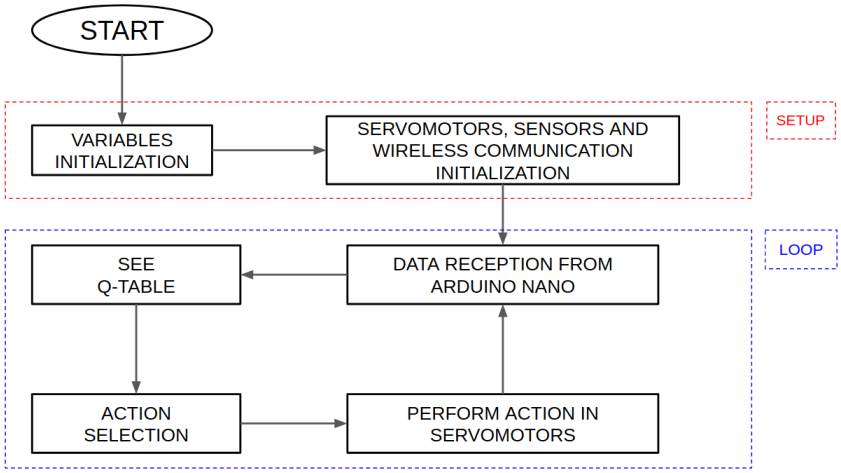


Figure 6.8: Flow process of the Arduino UNO already trained.

6.5 Code structure

The Pseudocode 1 **train Agent** is executed to train the agent, the input of the algorithm are the variables that describe the current state of the environment.
First are initialized the different variables the Qm is Q-matrix and all the values are initialized from zero, that will present changes in the values while learns the policy, K is the number of limited episodes, $Moves$ is the variable that limits the number of moves an agent can execute before finish an episode.

Then the train starts first is selected an x and y position, a random number selected between the 0 and the number of columns for x and the number of files for y of the Qm matrix, the variable $Agent$ is assigned an random starting action from Qm$[x][y]$ position.

In the following inner loop works as a limited sequence of steps where the agent can move around the environment until reach the goal or execute a limited number of moves defined previously in the variable $Moves$.The function StepSelection looks at matrix Qm neighborhood values, selects an action of the possible options. The possible options of the act allow the agent to move in four possible ways; up, down, left, right.

The variable $Agent$, x, y are updated depending on the previous move. The variable State' saves the new environment variables. The matrix Qm updates the index x,y value calling the function reward, which uses the variables $State$, $action$ to calculate the new value function Reward.

If the agent reaches the goal state the episode finish, otherwise this inner loop is repeated until finish the allowed number of steps.
The output is a Q-table matrix the Qm which has the experience obtained after the agent exploration.

Algorithm 1: Train agent

Input: "State" = Environment variables

Initialize of the Q matrix Qm, Initialize Qm; Initialize number episodes K;
Initialize number of moves $Moves$;
for $i \leftarrow 0$ ***to*** K **do**
    x=Random() and y=Random();
    $Agent = Qm[x][y]$;
    for $j \leftarrow 0$ ***to*** $Moves$ **do**
        take an action a,;
        ($step_x$,$step_y$) = StepSelection(action);
        $Agent = Qm[x + step_x][y + step_y]$;
        $State'$= Current environment variables;
        $Qm[x][y]$= Reward ($action$,$state$);
        if *CurrentStateBalanced() ==True* **then**
            break;
        end
    end
end

Output: Q-table matrix

Chapter 7

Results and Discussion

Table 7.1 shows the Q-table values for the servomotor that pushes the pole left and right, while table 7.2 shows the Q-table values for the second servomotor's forward and backward. The table depicts the Q-table during various training episodes, is possible to observe how the absolute value rises. The 0 value does not change during training because the episode ends when it enters this state; this state is considered the most stable because it receives a reward of 0, while the other states receive a punishment depending on the platform's inclination. It is easy to see how values that are farther from the stable state are punished more severely, resulting in extreme values gaining more knowledge, or learning faster than states closer to the optimal state. As compared to Q-tables with more iterations, it is possible to see that certain values in one column that should be lower than the other column are not. In the first 50 episodes in the table Q-table 7.1, it was determined that the agent does not have the experience to choose the proper actions at this point in the training.

In the cumulative reward Figure 7.1, the x-axis represents the number of episodes trained, while the y-values represent the absolute value total of each variable corresponding to the Q-table. The graphic demonstrates that the agent is learning, gaining experience after each iteration from the first to the nth episode. The Figure 7.1 plots a summary of all the learning work, both curves present a general growth doing decently, and over the last two hundred iterations agent 1 does not have a noticeable increase in comparison to earlier episodes, agent 2 shows that could improve its experience, this behavior comes from the random aspect of the training process, at some episodes would have a noticeable increase in comparison to earlier episodes, this behavior comes from the random aspect of the training process, at some episodes will have a noticeable.

In Figure 7.2 and 7.3 the column 0 and 1 are the 100 episodes Q-matrix, columns 2 and 3 are 500 episodes Q-table, finally, the column 4 and 5 are 1000 episodes Q-table. The colors with a more negative value represent a higher point and have a darker color, values closer to zero are more clear and lower points. The agent's path will be the clearest option where would be located, in the last two columns, the disparity in colors between the different states is more apparent compared to the prior columns, where the path to follow is somewhat diffuse.

Episode	50		100		500		1000	
State	Actions							
0	−3.254	−3.54	−3.254	−3.54	−9.02	−9.02	−14.30	−14.30
1	−2.15	−1.73	−2.15	−1.73	−10.21	−8.22	−15.15	−13.49
2	−3.35	−2.24	−3.35	−2.24	−9.13	−7.74	−13.52	−10.74
3	−3.75	−2.77	−3.75	−2.77	−9.04	−3.99	−11.75	−6.32
4	0	0	0	0	0	0	0	0
5	−.83	−.47	−.83	−.47	−1.00	−1.46	−1.	−1.67
6	−1.18	−1.73	−1.18	−1.73	−2.68	−3.28	−1.9	−3.4
7	−2.68	−4.69	−2.68	−4.69	−3.71	−7.44	−3.71	−6.50
8	−5.78	−5.78	−5.78	−5.78	−7.34	−7.34	−7.34	−7.34

Table 7.1: Agent 1 Q-table evolution of values at different episodes of training

Episode	100		500		1000	
State	Actions					
0	−7.63	−7.63	−16.42	−16.42	−16.13	−16.13
1	−7.59	−5.66	−17.45	−13.32	−17.44	−13.32
2	−3.91	−3.06	−12.95	−9.45	−12.96	−8.38
3	−1.44	−1.76	−6.56	−4.94	−8.48	−3.93
4	0	0	0	0	0	0
5	−0.86	−0.54	−1.00	−2.33	−1.00	−2.75
6	−3.10	−4.99	−3.90	−7.04	−3.90	−6.77
7	−7.04	−9.41	−8.51	−13.99	−8.51	−12.37
8	−9.49	−9.49	−12.66	−12.66	−12.66	−12.66

Table 7.2: Agent 2 Q-table evolution of values at different episodes of training

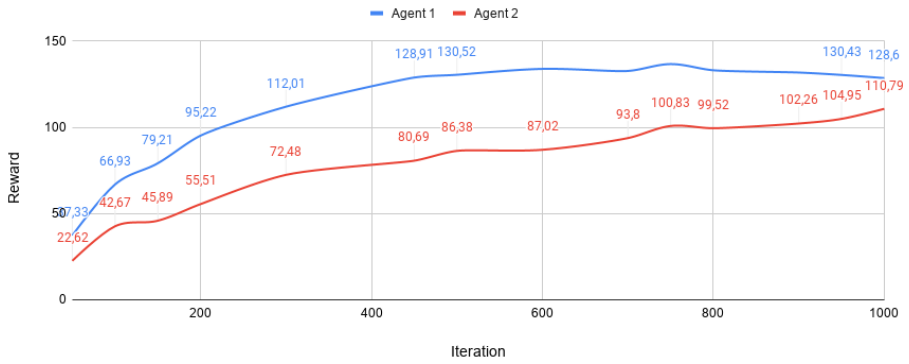


Figure 7.1: Cumulative reward as a function of the number of episodes, from both agents.

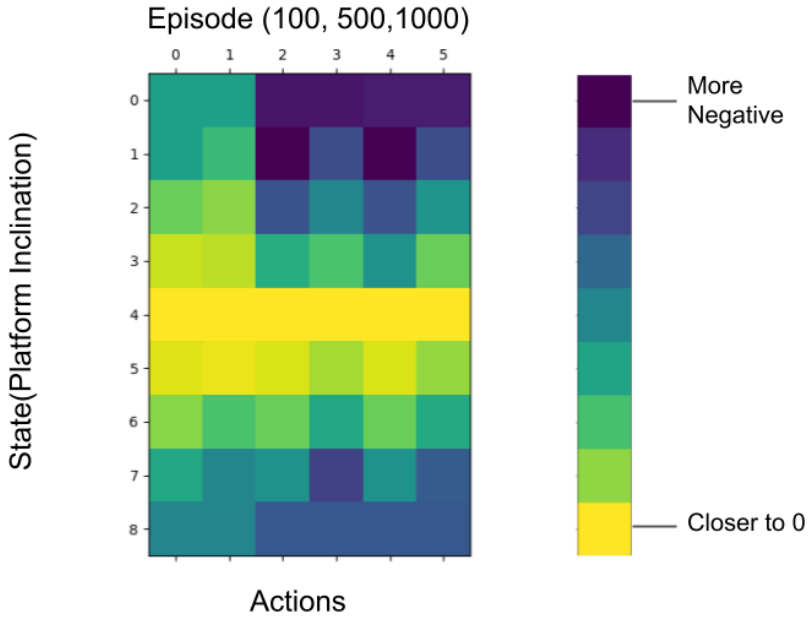


Figure 7.2: Agent heatmap left and right actions.

Figure 7.4 is an internal representation of the actual behavior after finished the training following a path of choosing the clearest option, depicts the direction that agent 1 will take if it is in the rightmost state, a right inclination. Agent 1 will gradually shift the pole to the left until it reaches the state in row 4. Agent is at state 0, which means the platform is facing backwards, and it will shift to the front by increasing the signal sent to the servomotor by 18°, moving from state 0 to state 4. Lastly in 7.5a is presented the real-world behaviour, where the platform has an inclination and the 7.5b the servomotors moved the pole the stable position.

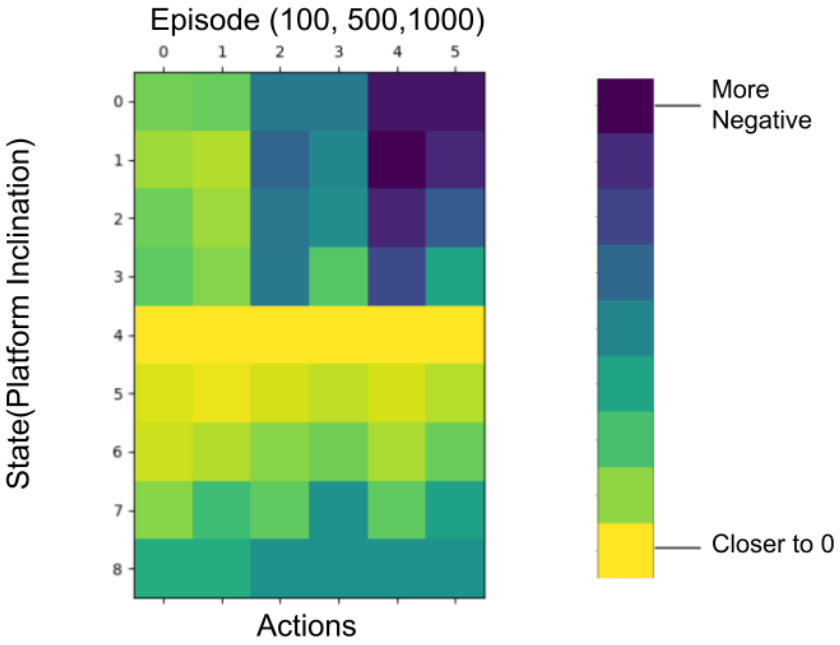


Figure 7.3: Agent heatmap back and forward actions.

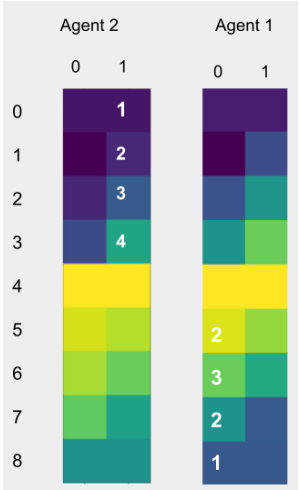
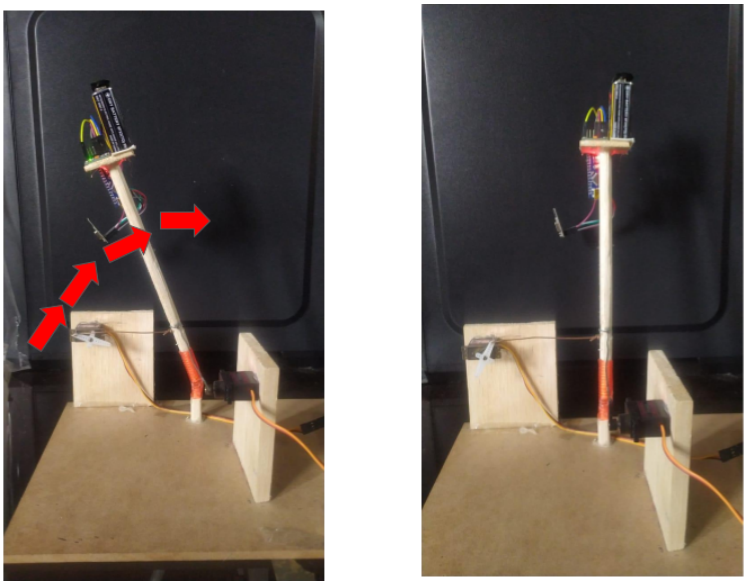


Figure 7.4: Internal behaviour of agents.



(a) Inclined platform. (b) Stable position.

Figure 7.5: Real-world behavior of the whole system working after detect an inclination.

Chapter 8

Conclusions

Within the reach of this thesis the automation process implementation with Artificial Intelligence and RL agents running in Arduino boards presents very promising results; however, constructing intelligent machines in the real world poses more development challenges like increasing the number of variables, the resolution of the sensors, the power of the actuators, etc. This degree project includes a study of Arduino microcontrollers and some of their peripherals and the implementation of reinforcement learning in a completely Arduino-based prototype consisting of two devices that operate by wireless communication. Also, during the project's progress, the following can be concluded:

- The restricted memory space of the Arduino for storing variables is not a barrier to running learning RL algorithms. In particular Q-learning algorithms can be fully implemented in current Arduino microcontrollers, as demonstrated in this project.
- In the real world most environments are analog, however, as demonstrated in this study, using Q-learning and a finite Q-matrix it is possible to convert analog changes in discrete signals and use them to control a pole balancing situation as a MDP.
- Wireless communication adds to the value of Arduino projects by allowing them to collect data and/or send instructions in complex environments with limited space and/or constant movement. It also opens possible valuable applications in the world of the internet of things (IoT)
- Agents and RL algorithms are a powerful tools because they allow robots to train themselves in tasks such as maintaining dynamic balance using only their implemented components such as sensors and actuators.
- Arduino platforms have access to a variety of peripherals, with methodical inclusion, and is a fertile ground to work and develop self-learning macro sensors, robots, and control systems.

Bibliography

[1] K. Azadeh, R. De Koster, and D. Roy, "Robotized and automated warehouse systems: Review and recent developments," *Transportation Science*, vol. 53, no. 4, pp. 917–945, 2019.

[2] D. A. Norman, "The 'problem'with automation: inappropriate feedback and interaction, not 'over-automation'," *Philosophical Transactions of the Royal Society of London. B, Biological Sciences*, vol. 327, no. 1241, pp. 585–593, 1990.

[3] T. Bretl, "Motion planning of multi-limbed robots subject to equilibrium constraints: The free-climbing robot problem," *The International Journal of Robotics Research*, vol. 25, no. 4, pp. 317–342, 2006.

[4] J. Foerster, N. Nardelli, G. Farquhar, T. Afouras, P. H. Torr, P. Kohli, and S. Whiteson, "Stabilising experience replay for deep multi-agent reinforcement learning," in *International conference on machine learning*. PMLR, 2017, pp. 1146–1155.

[5] S.-H. Hyon, J. G. Hale, and G. Cheng, "Full-body compliant human–humanoid interaction: balancing in the presence of unknown external forces," *IEEE Transactions on Robotics*, vol. 23, no. 5, pp. 884–898, 2007.

[6] T. Perez, M. Blanke *et al.*, "Ship roll motion control," in *8th IFAC Conference on Control Applications in Marine Systems, Rostock, Germany*, vol. 183, 2010.

[7] M. Sun, T. Luan, and L. Liang, "Rbf neural network compensation-based adaptive control for lift-feedback system of ship fin stabilizers to improve anti-rolling effect," *Ocean Engineering*, vol. 163, pp. 307–321, 2018.

[8] H. Korkmaz, O. B. Ertin, C. Kasnakoğlu *et al.*, "Design of a flight stabilizer system for a small fixed wing unmanned aerial vehicle using system identification," *IFAC Proceedings Volumes*, vol. 46, no. 25, pp. 145–149, 2013.

[9] S. Akyurek, G. S. Ozden, B. Kurkcu, U. Kaynak, and C. Kasnakoglu, "Design of a flight stabilizer for fixed-wing aircrafts using h∞*loop* shaping method," in *2015 9th International Conference on Electrical and Electronics Engineering (ELECO)*. IEEE, 2015, pp. 790–795.

[10] S. Elias and V. Matsagar, "Research developments in vibration control of structures using passive tuned mass dampers," *Annual Reviews in Control*, vol. 44, pp. 129–156, 2017.

[11] I. Jmel, H. Dimassi, S. Hadj Said, and F. M'Sahli, "Adaptive observer-based output feedback control for two-wheeled self-balancing robot," *Mathematical Problems in Engineering*, vol. 2020, 2020.

[12] C. Ott, M. A. Roa, and G. Hirzinger, "Posture and balance control for biped robots based on contact force optimization," in *2011 11th IEEE-RAS International Conference on Humanoid Robots*. IEEE, 2011, pp. 26–33.

[13] O. Miglino, H. H. Lund, and S. Nolfi, "Evolving mobile robots in simulated and real environments," *Artificial life*, vol. 2, no. 4, pp. 417–434, 1995.

[14] I. A. Taha and H. M. Marhoon, "Implementation of controlled robot for fire detection and extinguish to closed areas based on arduino," *Telkomnika*, vol. 16, no. 2, pp. 654–664, 2018.

[15] F. M. López-Rodríguez and F. Cuesta, "Andruino-a1: Low-cost educational mobile robot based on android and arduino," *Journal of Intelligent & Robotic Systems*, vol. 81, no. 1, pp. 63–76, 2016.

[16] A. Araújo, D. Portugal, M. S. Couceiro, and R. P. Rocha, "Integrating arduino-based educational mobile robots in ros," *Journal of Intelligent & Robotic Systems*, vol. 77, no. 2, pp. 281–298, 2015.

[17] S. A. Ram, N. Siddarth, N. Manjula, K. Rogan, and K. Srinivasan, "Real-time automation system using arduino," in *2017 International Conference on Innovations in Information, Embedded and Communication Systems (ICIIECS)*. IEEE, 2017, pp. 1–5.

[18] I. González and A. J. Calderón, "Integration of open source hardware arduino platform in automation systems applied to smart grids/micro-grids," *Sustainable Energy Technologies and Assessments*, vol. 36, p. 100557, 2019.

[19] D. Wu, S. Liu, L. Zhang, J. Terpenny, R. X. Gao, T. Kurfess, and J. A. Guzzo, "A fog computing-based framework for process monitoring and prognosis in cybermanufacturing," *Journal of Manufacturing Systems*, vol. 43, pp. 25–34, 2017.

[20] A. K. Jain, "Working model of self-driving car using convolutional neural network, raspberry pi and arduino," in *2018 Second International Conference on Electronics, Communication and Aerospace Technology (ICECA)*. IEEE, 2018, pp. 1630–1635.

[21] P. S. Lengare and M. E. Rane, "Human hand tracking using matlab to control arduino based robotic arm," in *2015 International Conference on Pervasive Computing (ICPC)*. IEEE, 2015, pp. 1–4.

[22] A.-F. Jimenez, P.-F. Cardenas, A. Canales, F. Jimenez, and A. Portacio, "A survey on intelligent agents and multi-agents for irrigation scheduling," *Computers and Electronics in Agriculture*, p. 105474, 2020.

[23] R. Salazar, J. C. Rangel, C. Pinzón, and A. Rodríguez, "Irrigation system through intelligent agents implemented with arduino technology," 2013.

[24] M. F. Mata-Rivera, R. Zagal-Flores, and C. Barría-Huidobro, *Telematics and Computing: 8th International Congress, WITCOM 2019, Merida, Mexico, November 4–8, 2019, Proceedings*. Springer Nature, 2019, vol. 1053.

[25] P. Kormushev, S. Calinon, and D. G. Caldwell, "Reinforcement learning in robotics: Applications and real-world challenges," *Robotics*, vol. 2, no. 3, pp. 122–148, 2013.

[26] X. Pan, Y. You, Z. Wang, and C. Lu, "Virtual to real reinforcement learning for autonomous driving," *arXiv preprint arXiv:1704.03952*, 2017.

[27] A. Nagabandi, I. Clavera, S. Liu, R. S. Fearing, P. Abbeel, S. Levine, and C. Finn, "Learning to adapt in dynamic, real-world environments through meta-reinforcement learning," *arXiv preprint arXiv:1803.11347*, 2018.

[28] S. Gu, E. Holly, T. Lillicrap, and S. Levine, "Deep reinforcement learning for robotic manipulation with asynchronous off-policy updates," in *2017 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2017, pp. 3389–3396.

[29] A. Sharma, M. Ahn, S. Levine, V. Kumar, K. Hausman, and S. Gu, "Emergent real-world robotic skills via unsupervised off-policy reinforcement learning," *arXiv preprint arXiv:2004.12974*, 2020.

[30] J. Garcia and D. Shafie, "Teaching a humanoid robot to walk faster through safe reinforcement learning," *Engineering Applications of Artificial Intelligence*, vol. 88, p. 103360, 2020.

[31] D. A. Cohn, Z. Ghahramani, and M. I. Jordan, "Active learning with statistical models." MASSACHUSETTS INST OF TECH CAMBRIDGE ARTIFICIAL INTELLIGENCE LAB, Tech. Rep., 1995.

[32] N. A. Barriga, M. Stanescu, F. Besoain, and M. Buro, "Improving rts game ai by supervised policy learning, tactical search, and deep reinforcement learning," *IEEE Computational Intelligence Magazine*, vol. 14, no. 3, pp. 8–18, 2019.

[33] X. Zhu, J. Lafferty, and R. Rosenfeld, "Semi-supervised learning with graphs," Ph.D. dissertation, Carnegie Mellon University, language technologies institute, school of . . . , 2005.

[34] C. J. Watkins and P. Dayan, "Q-learning," *Machine learning*, vol. 8, no. 3-4, pp. 279–292, 1992.

[35] R. S. Sutton, "Introduction: The challenge of reinforcement learning," in *Reinforcement Learning*. Springer, 1992, pp. 1–3.

[36] J. Garcıa and F. Fernández, "A comprehensive survey on safe reinforcement learning," *Journal of Machine Learning Research*, vol. 16, no. 1, pp. 1437–1480, 2015.

[37] O. Chang, L. Zhinin-Vera, and F. Quinga-Socasi, "Self-taught neural agents in clever game playing," in *Proceedings of the Future Technologies Conference*. Springer, 2020, pp. 512–524.

[38] L. Buşoniu, R. Babuška, and B. De Schutter, "Multi-agent reinforcement learning: An overview," *Innovations in multi-agent systems and applications-1*, pp. 183–221, 2010.

[39] F. Sehnke, C. Osendorfer, T. Rückstieß, A. Graves, J. Peters, and J. Schmidhuber, "Parameter-exploring policy gradients," *Neural Networks*, vol. 23, no. 4, pp. 551–559, 2010.

[40] A. Kramer and B. Legeard, *Model-Based Testing Essentials*. Wiley Online Library, 2016.

[41] K. Arulkumaran, M. P. Deisenroth, M. Brundage, and A. A. Bharath, "A brief survey of deep reinforcement learning," *arXiv preprint arXiv:1708.05866*, 2017.

[42] A. McCallum, D. Freitag, and F. C. Pereira, "Maximum entropy markov models for information extraction and segmentation." in *Icml*, vol. 17, no. 2000, 2000, pp. 591–598.

[43] Y. Hirashima, Y. Iiguni, A. Inoue, and S. Masuda, "Q-learning algorithm using an adaptive-sized q-table," in *Proceedings of the 38th IEEE Conference on Decision and Control (Cat. No. 99CH36304)*, vol. 2. IEEE, 1999, pp. 1599–1604.

[44] Y. Güven, E. Coşgun, S. Kocaoğlu, H. Gezici, and E. Yılmazlar, "Understanding the concept of microcontroller based systems to choose the best hardware for applications." 2017.

[45] D. E. Bolanakis, "A survey of research in microcontroller education," *IEEE Revista Iberoamericana de Tecnologias del Aprendizaje*, vol. 14, no. 2, pp. 50–57, 2019.

[46] A. Trevennor, "A brief history of microcontrollers," in *Practical AVR Microcontrollers*. Springer, 2012, pp. 3–11.

[47] M. Margolis, B. Jepson, and N. R. Weldin, *Arduino cookbook: recipes to begin, expand, and enhance your projects*. O'Reilly Media, 2020.

[48] S. F. Barrett, "Arduino microcontroller processing for everyone!" *Synthesis Lectures on Digital Circuits and Systems*, vol. 8, no. 4, pp. 1–513, 2013.

[49] Arduino. Open-source electronic prototyping platform enabling users to create interactive electronic objects. [Online]. Available: https://store.arduino.cc

[50] A. A. Rafiq, W. N. Rohman, and S. D. Riyanto, "Development of a simple and low-cost smartphone gimbal with mpu-6050 sensor," *Journal of Robotics and Control (JRC)*, vol. 1, no. 4, pp. 136–140, 2020.

[51] H. Jian, "Design of angle detection system based on mpu6050," in *7th International Conference on Education, Management, Information and Computer Science (ICEMC 2017)*. Atlantis Press, 2016, pp. 6–8.

[52] G. Chinnadurai and H. Ranganathan, "Balancing and control of dual wheel swarm robots by using sensors and port forwarding router," in *2017 Third International Conference on Advances in Electrical, Electronics, Information, Communication and Bio-Informatics (AEEICB)*. IEEE, 2017, pp. 532–536.

[53] M. Araki and H. Taguchi, "Two-degree-of-freedom pid controllers," *International Journal of Control, Automation, and Systems*, vol. 1, no. 4, pp. 401–411, 2003.

[54] electronicoscaldas.com. Mg90s servo, metal gear with one bearing. [Online]. Available: http://www.electronicoscaldas.com/datasheet/MG90S_Tower-Pro.pdf

[55] www.sparkfun.com. nrf24l01 + preliminary product specification. [Online]. Available: https://www.sparkfun.com/datasheets/Components/SMD/nRF24L01Pluss_Preliminary_Product_Specification_v1_0.pdf

Appendices

.1 Appendix 1.

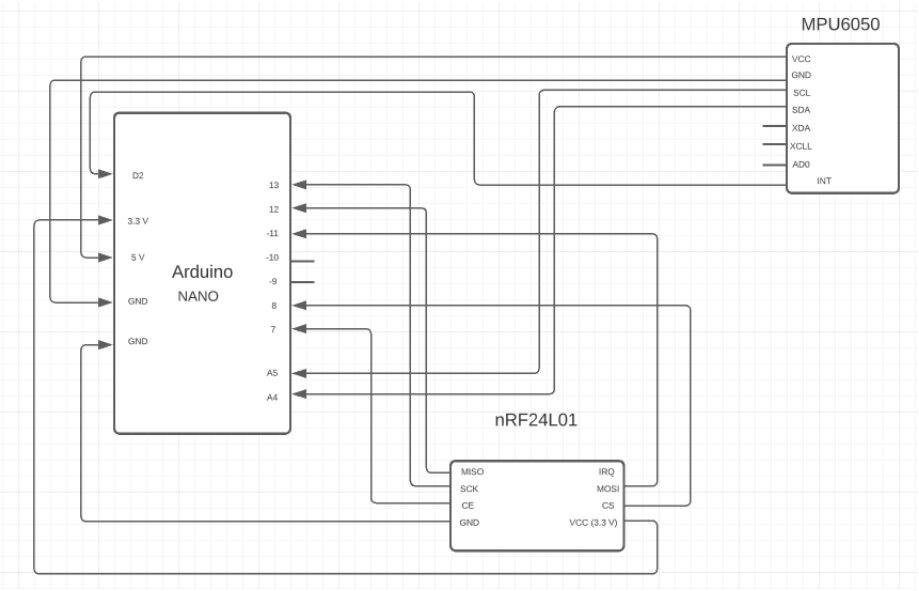


Figure 1: Device A electrical circuit wiring diagram.

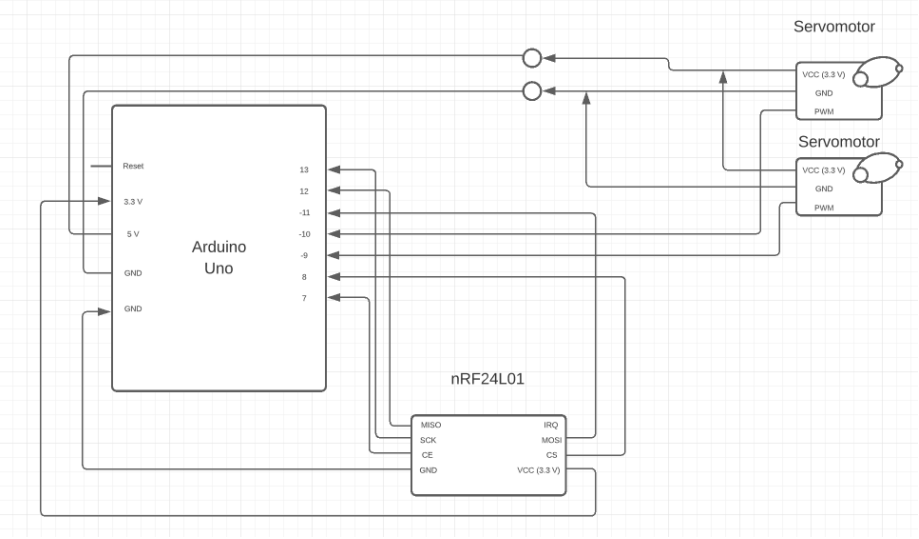


Figure 2: Device B electrical circuit wiring diagram.

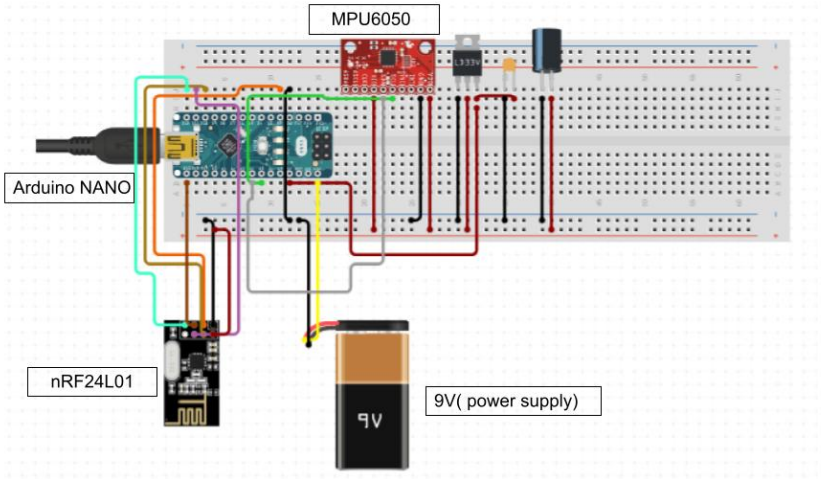


Figure 3: Device B electrical circuit wiring diagram.

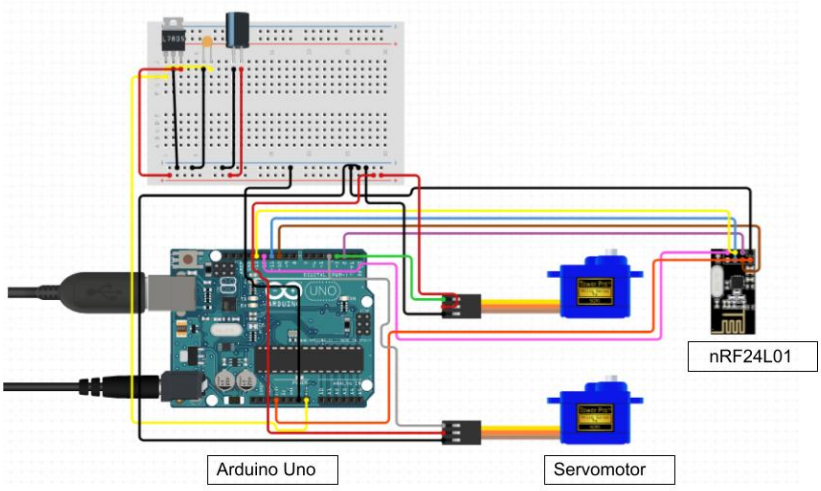


Figure 4: Device B electrical circuit wiring diagram.