

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université 8 Mai 1945 – Guelma
Faculté des Sciences et de la Technologie
Département de Génie Electrotechnique et Automatique

Réf:/2022



MEMOIRE

Présenté pour l'obtention du diplôme de MASTER Académique

Domaine: Sciences et Technologie

Filière: Automatique

Spécialité: Automatique et informatique industrielle

Par: MENAI Aymen

Thème

**Système intelligent de monitoring et de commande d'un système
MIMO a base la technologie VLC**

Soutenu publiquement, le

18 /06 /2022, devant le jury composé de:

Mr. BABOURI Abdesselam

Professeur

Univ.Guelma

Encadreur

Mr. CHAABNA Ameer

Docteur

USTHB.Algiers

Co-Encadreur

Mr. CHOUABIA Halim

Doctorant

Univ.Guelm

Co-Encadreur

Année Universitaire: 2021/2022

ACKNOWLEDGMENTS

I want to give my thanks to my supervisor Pr. Barbouri Abdesselam for teaching me well this year, and introduce me to the visible light communication and for being always up to date with me in this project, as well as I want to thank to Dr. Chaabna Ameer and Mr. Chouabia Halim that they help me with that project.

Also I want to give a thank to my family, all of them that they are really proud of me for being in this position that I reached now.

I want to have a special thing to my parent for their continuous support that they gave me during my education journey and that they being with me in my hard time weakness time where I gave up, I'm really thankful that I have parents like you and from my deepest heart I want to say "Thank you".

I want to thank my very one helped me my friends, my family, my teachers, the institutes that support me in that.

Abstract

In this work, we studied and treated artificial intelligence in a robots and connect with it using the visible light communication technology as a primary way of making the connection with the robot, the machine learning is a branch that it will be used in that project as well, we are going to use open cv which is a library to treat images and to make studying about the current image and we're going to use tensorflow lite to save some of our memory and to have better performance, in the side of the hardware we are going to use arduino as a micro-controller and raspberry pi (3 and 4) as micro-processor to build a semi-controller and self-controller robot

Résumé

Dans ce travail, nous avons étudié et traité l'intelligence artificielle dans un robot et nous y sommes connectés en utilisant la technologie de communication par la lumière visible comme moyen principal d'établir la connexion avec le robot, l'apprentissage automatique est une branche qui sera utilisée dans ce projet aussi, nous allons utiliser open cv qui est une bibliothèque pour traiter les images et faire des études sur l'image actuelle et nous allons utiliser tensorflow lite pour économiser une partie de notre mémoire et avoir de meilleures performances, du côté du matériel nous allons utiliser arduino comme micro-contrôleur et raspberry pi (3 et 4) comme micro-processeur pour construire un robot semi-contrôleur et autocontrôleur

الملخص

في هذا العمل ، قمنا بدراسة ومعالجة الذكاء الاصطناعي في الروبوتات والتواصل معها باستخدام تقنية الاتصال بالضوء المرئي كطريقة أساسية لإجراء الاتصال مع الروبوت ، والتعلم الآلي هو فرع سيتم استخدامه في هذا المشروع
سنستخدم اوبن سيفي وهي مكتبة لمعالجة الصور ولجعل دراسة الصورة الحالية وسنستخدم ايضا تنسر فلو لايت لحفظ جزء من ذاكرتنا والحصول على أداء أفضل ، من جهة الأجهزة التي سوف تستخدم : اردوينو كوحدة تحكم صغيرة و راسبيري باي (ثلاثة و اربعة) كمعالج صغير لبناء روبوت شبه تحكم وتحكم ذاتي

TABLE OF CONTENTS

List of Abbreviations.....	9
General Introduction.....	10
I. CHAPTER I : INTRODUCTION TO ROBOTS AND ROBOTICS	
I.1. Introduction	12
I.2. Robotics	12
I.3. Robot	12
I.4. Robot's types	13
I.4.1. Robot Manipulators	13
I.4.2. Mobile Robot	13
I.5. Field and application	14
I.6. Conclusion	19
II. CHAPTER II : INTRODUCTION TO MACHINE LEARNING	
II.1. Introduction.....	21
II.2. Machine learning.....	21
II.3. Machine learning application.....	22
II.4. Types of learning.....	26
II.4.1. Supervised learning.....	26
II.4.2. Unsupervised learning.....	27
II.4.3. Semi Supervised learning	28
II.4.4. Reinforcement learning.....	29
II.5. Conclusion.....	30
III. CHAPTER III : INTRODUCTION TO VISIBAL LIGHT COMMUNICATION	
III.1. Introduction	32
III.2. VLC	32
III.3. VLC architecture	33
III.3.1. Transmitter	33
III.3.1.1. LED	33
III.3.2. Receiver	35
III.3.2.1. Photodiode	35
III.3.3. Transmission Channel	36
III.4. VLC application.....	36
III.4.1. Indoor.....	36
III.4.2. Outdoor	39
III.5. Conclusion	40
IV. CHAPTER IV : APPLICATION	
IV.1. Introduction	42
IV.2. Simulation part	42
IV.3. Expiremental part	75
IV.4. Conclusion	76
General Conclusion.....	77
Refrences	78

LIST OF FIGURES

- Figure I.1:** The Shadow robot hand system
- Figure I.2:** Examples of different robot manipulators
- Figure I.3:** Examples of different Mobile robot
- Figure I.4:** Industrial robots
- Figure I.5:** Wheeled robot
- Figure I.6:** Legged robot
- Figure I.7:** Medical robot
- Figure I.8:** Realistic high-resolution render image of a CENTAURO mock up model by RWTH Aachen in an urban disaster scenario
- Figure I.9:** Drone
- Figure I.10:** Armed Predator drone
- Figure I.11:** Self-driving cars sensors
- Figure II.1:** AI vs ML vs DL
- Figure II.2:** Computer Vision (image trained)
- Figure II.3:** Neural networks and speech recognition - Machine Learning
- Figure II.5:** Defined & Strategies
- Figure II.6:** Example about learning from data
- Figure II.7:** Example about unsupervised learning
- Figure II.8:** Semi-supervised learning
- Figure II.9:** Reinforcement learning
- Figure III.1:** Visible light is only a small portion of the electromagnetic spectrum
- Figure III.2:** Architecture of a full-duplex VLC communication system
- Figure III.3:** Basic representation of a standard LED
- Figure III.4:** Photodiode size and example
- Figure III.5:** Open-VLC 1.0 transceivers
- Figure III.6:** The Working Of Li-Fi Communication
- Figure III.7:** Hospital delivery robot manufactured by Panasonic
- Figure III.8:** Basic diagram on indoor positioning using VLC
- Figure III.9:** V2V and I2V communication
- Figure III.10:** Underwater Communication
- Figure IV.1:** Github repo of RoboND-Rover-Project
- Figure IV.2:** Udacity logo
- Figure IV.3:** File of the simulation after the extraction
- Figure IV.4:** Rover Simulator Configuration
- Figure IV.5:** Simulator window
- Figure IV.6:** the start of the simulation
- Figure IV.7:** set the output
- Figure IV.8:** Collecting items in the environment
- Figure IV.9:** Output
- Figure IV.10:** IMG folder
- Figure IV.11:** image from IMG folder
- Figure IV.12:** robot_log.csv
- Figure IV.13:** Jupyter Logo
- Figure IV.14:** Terminal window
- Figure IV.15:** Terminal window, and type "su" command
- Figure IV.16:** Do some command to open server for jupyter notebook
- Figure IV.17:** Jupyter notebook browser
- Figure IV.18:** Create file in Jupyter notebook

Figure IV.19: image 1 ,2
Figure IV.20: image 3,4
Figure IV.21: Result of image 1 in the filtre (normal,ground,rock and mountain)
Figure IV.22: Result of image 2 in the filtre (normal,ground,rock and mountain)
Figure IV.23: Result of image 3 in the filtre (normal,ground,rock and mountain)
Figure IV.24: Result of image 4 in the filtre (normal,ground,rock and mountain)
Figure IV.25: Result of image 1 after coded
Figure IV.26: Result of image 2 after coded
Figure IV.27: Result of image 3 after coded
Figure IV.28: Result of image 3 after coded
Figure IV.29: Result of image 3 after we remove the rock property
Figure IV.30: Image with grid
Figure IV.31: The grid picture in normal and after the perspective transform from z index view
Figure IV.32: Result of image 1 normal and with perspective transform
Figure IV.33: Result of image 2 normal and with perspective transform
Figure IV.34: Result of image 3 normal and with perspective transform
Figure IV.35: Result of image 4 normal and with perspective transform
Figure IV.36: Perspective transform of image 1 with coded
Figure IV.37: Perspective transform of image 2 with coded
Figure IV.38: Perspective transform of image 3 with coded
Figure IV.39: Perspective transform of image 4 with coded
Figure IV.40: Map of the envirimnt
Figure IV.45: Coded ground filtre, coded color of prespective transform, blocked coded color of prespective transform and position in the map (image 1)
Figure IV.46: Coded ground filtre, coded color of prespective transform, blocked coded color of prespective transform and position in the map (image 2)
Figure IV.47: Coded ground filtre, coded color of prespective transform, blocked coded color of prespective transform and position in the map (image 3)
Figure IV.48: Coded ground filtre, coded color of prespective transform, blocked coded color of prespective transform and position in the map (image 4)
Figure IV.49: VLC-Car

LIST OF ABBREVIATIONS

ROS : Robot Operation System
IBM : International Business Machine
AI: Artificial intelligence
ML: Machine learning
DL: Deep learning
RL: Reinforcement learning
DRL: Deep Reinforcement learning
SVM: Support Vector Machine
KNN: K Nearest Neighbor
NN: Neural Network
CNN: Convolutional Neural Network
IoT : Internet of Things
Li-Fi : Light Fidelity
Wi-Fi: Wireless Fidelity
HOSPI:Hospital delivery robot manufactured by Panasonic
MRI:Magnetic Resonance Imaging
RGB: Red, Green, Blue LEDs
Repo : Repository
MOOC : Massive Open Online Course
NASA : National Aeronautics and Space Administration
OS : Operating System
Linux distro : Linux distribution Software

General introduction :

In our modern world, we face a lot challenges and tasks that can really improve and save our lives in better way or even the best way possible, but sometimes we have those tasks needed an extra energy that not availabe for human being to do it.

that's why human started to think about solution to make our life easier with less effort. At this point the human thinks about robots, the robot was the solution to deal with tasks that's not possible for ordinary human to do because in fit the case used.

the most interesing part in creating a smart robot is working in his brain which its the actifical intelligence that's a whole new branch in our modern world that's allowing the machine to learn from its envirements based on the inputs that it gets from its sensors like the camera, the microphone and others, the robot will use the inputs and train them to make decisions and save them to learn from those later to make future decisions.

the Visible light communication (or VLC) seems to be a very useful tool to be integrated in robots, due the modern world is more brighter than before and light is availabe now and cheaper price of components like LED, photodiode, also that technology will allow us to use it as meduim between the pilot and the robot, other advantages of using that technology is that it's safe and fast.

Chapter I

Introduction to Robots & Robotics

I.1. Introduction

Robotics is the study of devices that can perform tasks in place of humans, including physical activity and decision-making. The introduction's goal is to highlight the problems surrounding the utilization of robots in industrial applications, as well as the opportunities given by advanced robotics, a classification of the most common mechanical structures of robot manipulators and mobile robots are presented[1].

I.2. Robotics

The field of robotics has deep cultural origins. Throughout history of the human beings have continually worked to find replacements that possible.

The term robotics was taken from the word robot, which was popularized by Czech writer Karel Čapek in his play R.U.R. (Rossum's Universal Robots), released in 1920. The term "robot" is derived from the Slavic word "robota," which means "work/job" . The play starts in a factory that creates artificial people known as robots, creatures that may be mistaken for humans - quite similar to current concepts of androids. The term was not coined by Karel Čapek. He submitted a short letter in response to an etymology in the Oxford English Dictionary, naming his brother Josef Čapek as the real founder[2].

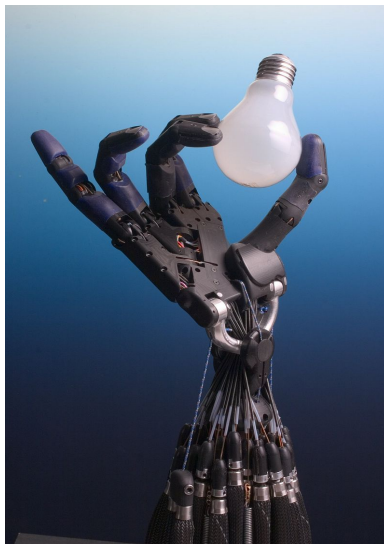


Figure I.1: The Shadow robot hand system [3]

I.3. Robot

A robot is a machine that can make decisions on its own based on data from sensors. A software agent is a program that is meant to process and output data autonomously. Perhaps the easiest way to characterize a robot is as a self-contained software agent with sensors and moving

outputs. It might also be described as an electro-mechanical platform that runs software. A robot, in any case, requires electronics, mechanical parts, and software[4].

I.4. Robot's types

I.4.1 Robot Manipulators

A robot manipulator's mechanical structure consists of a series of rigid bodies (links) joined by articulations (joints); a manipulator is distinguished by an arm that provides mobility, a wrist that provides dexterity, and an end-effector that executes the robot's task. The serial or open kinematic chain is the basic framework of a manipulator. When there is just one series of links connecting the two endpoints of a kinematic chain, it is called open from a topological standpoint. A manipulator, on the other hand, has a closed kinematic chain when a series of links creates a loop[1].

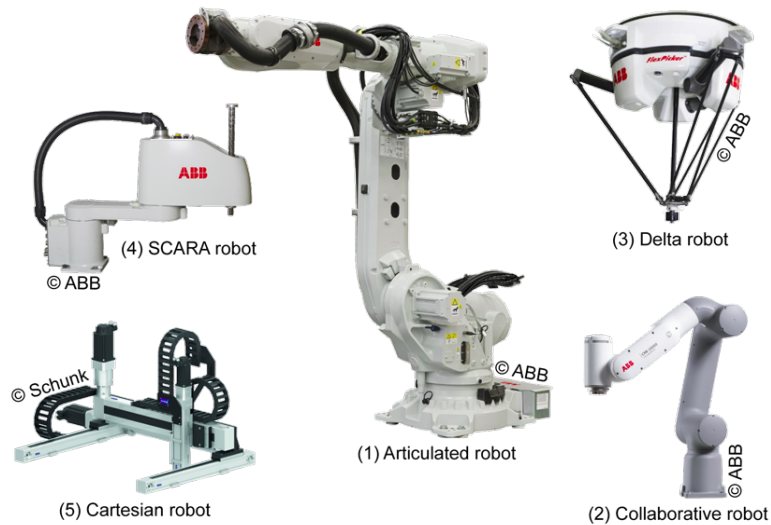


Figure I.2: Examples of different robot manipulators[5].

I.4.2. Mobile Robots

A robot manipulator's mechanical structure consists of a series of rigid bodies (links) joined by articulations (joints); a manipulator is distinguished by an arm that provides mobility, a wrist that provides dexterity, and an end-effector that executes the robot's task. The serial or open kinematic chain is the basic framework of a manipulator. When there is just one series of links connecting the two endpoints of a kinematic chain, it is called open from a topological standpoint. A manipulator, on the other hand, has a closed kinematic chain when a series of links creates a loop[6].



Figure I.3: Examples of different Mobile robot[7].

I.5. Fields and application

Using Robots seems so common in nowadays so this is some of the fields used the robots on it [6].

I.5.1. Industrial

An industrial robot is a production robot. Industrial robots are programmable, automated, and have three or more axes of motion. The Unimate, the grandfather of all manufacturing robots, is one such example. Systems like Amazon's warehouse robots and collaborative production robots that can work with humans are included in this category[6].



Figure I.4: Industrial robots[8].

I.5.2. Wheeled robots

Wheeled robots are those that use their wheels to move around on the ground. This is a popular design since it is the most basic in terms of design, manufacture, and programming. Wheeled robots also have the benefit of being easier to manage than other robot designs[6].



Figure I.5: Wheeled robot[9].

I.5.3. Legged Robots

Legged robots are mobile robots, but their mobility is more complex and intelligent than that of their wheeled counterparts. They manage their mobility using their legs, as the name implies, and they perform far better on rough terrain than wheeled robots. Despite their high cost and great production complexity, these robots are important for most applications because of their benefits over uneven terrain[6].



Figure I.6: Legged robot[10].

I.5.4. Medical Robots

Artificial machines and stationary robots like the da Vinci surgical robot are examples of health-care robots. Watson, IBM's question-answering supercomputer, has been employed in healthcare applications and falls into this category without being a robot[6].



Figure I.7: Medical robot[11].

I.5.5. Disaster Response

These robots conduct perilous tasks such as hunting for survivors after a disaster. Pack bots were employed to evaluate damage at the Fukushima Daiichi nuclear power plant after an earthquake and tsunami ravaged Japan in 2011[6].



Figure I.8: Realistic high-resolution render image of a CENTAURO mock up model by RWTH Aachen in an urban disaster scenario.[12].

I.5.6. Drones

Drones are available in a variety of sizes and levels of autonomy. DJI's Phantom series and Parrot's Anafi are two examples, as are military systems like the Global Hawk, which are used for long-term surveillance[6].



Figure I.9: Drone[13].

I.5.7. Military & Security

Ground systems like Endeavor Robotics' PackBot, which is used to search for improvised explosive devices, and BigDog, which assists troops in carrying heavy gear, are examples of military robots. Cobalt, an autonomous mobility system, is an example of a security robot[6].



Figure I.10: Armed Predator drone [14].

I.5.8. Self-Driving Cars

Many robots navigate by themselves. Early autonomous cars include those produced for DARPA's autonomous-vehicle contests, as well as Google's self-driving Toyota Prius, which was eventually split off to become Waymo[6].

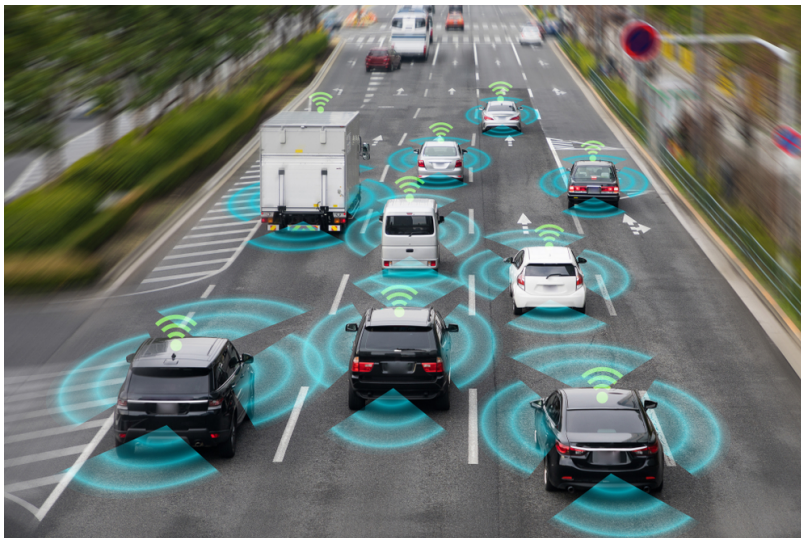


Figure I.11: Self-driving cars sensors [15].

I.6. Conclusion

In conclusion, we can say that Robotics is the science that allows us to build robots, a robot is a machine that uses data from sensors to make choices on its own. The mechanical structure of a robot manipulator is made up of a sequence of rigid bodies (links) connected by articulations (joints): an arm for movement, a wrist for dexterity, and an end-effector for completing the robot's mission. Industrial robots have three or more axes of motion, are programmable, and automated. Legged robots are mobile robots with more complicated and intelligent movement than their wheeled cousins. After an earthquake and tsunami struck Japan in 2011, pack bots were used to assess damage at the Fukushima Daiichi nuclear power facility. Drones come in a wide range of sizes and autonomous levels. As a simple sentence we can say that robotics simplify human life and produced

Chapter II

Introduction to machine learning

II.1. Introduction

When most people hear "machine learning," they immediately think of a robot: either a trusty butler or a scary Terminator, depending on who you ask. Machine Learning, on the other hand, isn't simply a distant dream; it's already here. In certain specialized applications, such as Optical Character Recognition, it has been around for decades. But it was the spam filter that was the first ML application to truly go popular, impacting the lives of hundreds of millions of people in the 1990's.

It isn't quite a self-aware Skynet, but it qualifies as Machine Learning (it has really learnt so effectively that you seldom need to mark an email as spam any longer). Hundreds more machine learning apps followed, silently powering dozens of goods and features you use every day, from smarter suggestions to voice search.

When it comes to machine learning, where does it begin and where does it end? What does it imply for a machine to be able to learn something? Has my computer "learned" anything if I downloaded a copy of Wikipedia? Is it becoming smarter? We'll start by defining Machine Learning and why you would want to apply[16].

II.2. Machine learning

Machine learning or the ML is a branch of artificial intelligence that allows computers to learn and develop without being explicitly programmed. The goal of machine learning is to create computer algorithms that can access data and learn on their own[17].

ARTIFICIAL INTELLIGENCE VS MACHINE LEARNING VS DEEP LEARNING

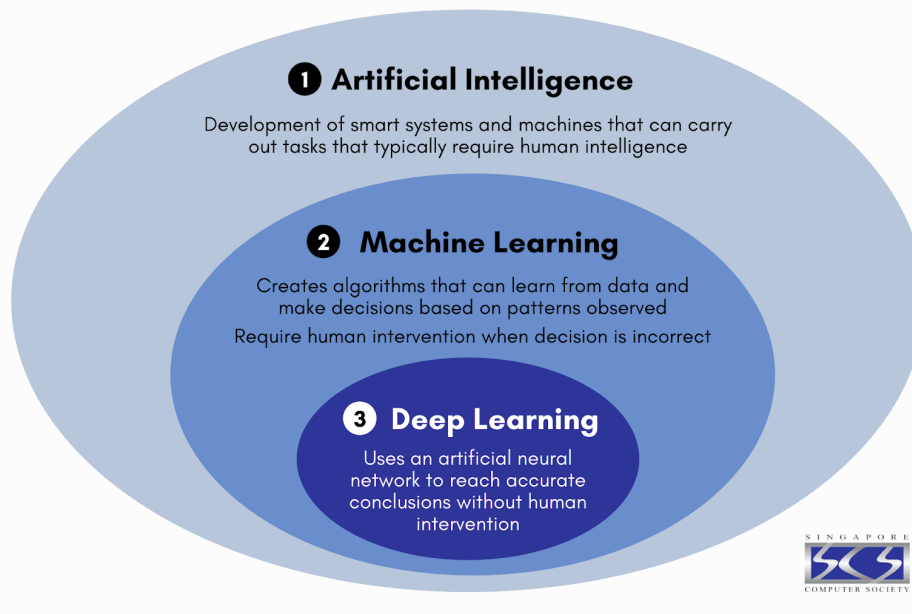


Figure II.1: AI vs ML vs DL [18].

II.3. Machine learning applications

II.3.1. Image recognition

In the real world, image recognition is a well-known and often used example of machine learning. Based on the intensity of the pixels in black and white or color photos, it may recognize an item as a digital image[19].

Image recognition in the real world:

- Determine if an x-ray is malignant or not.
- Give an image a name (also known as "tagging" on social media).
- Segment a single letter into smaller pictures to recognize handwriting.

Face recognition inside a picture is another use of machine learning. The technology can discover similarities and match them to faces using a database of individuals. In law enforcement, this is frequently employed[19].

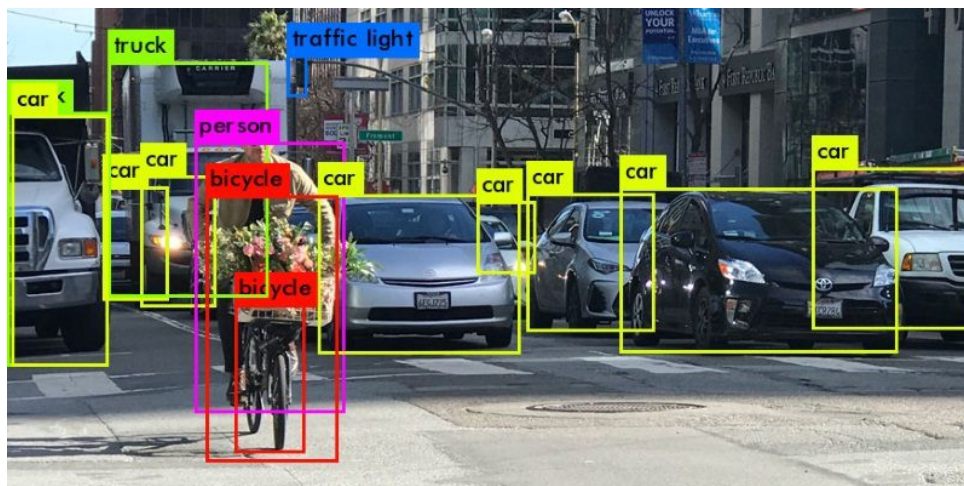


Figure II.2: Computer Vision (image trainede)[20].

II.3.2. Speech recognition

Speech to text translation is possible using machine learning. Live voice and recorded speech may both be converted to text files using certain software tools. Intensities on time-frequency bands can also be used to partition speech[19].

Speech recognition in the real world:

- Search by voice
- Dialing by voice
- Controlling the appliances

Devices like Google Home and Amazon Alexa are two of the most common uses of speech recognition technology[19].

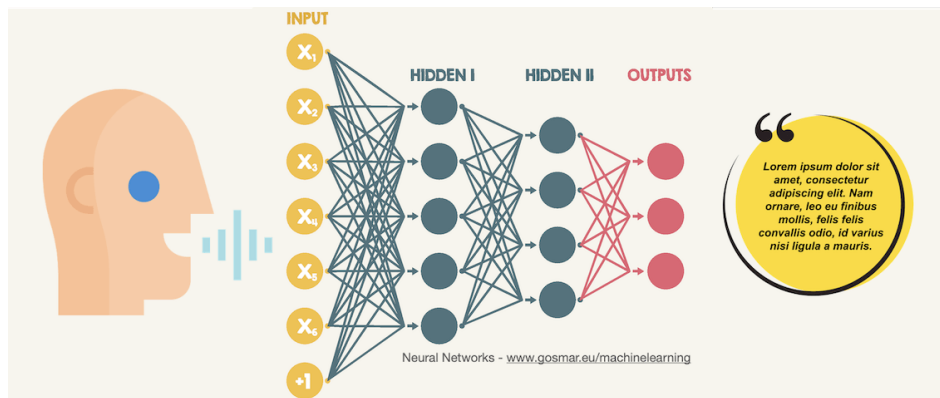


Figure II.3: Neural networks and speech recognition - Machine Learning[21].

II.3.3. Medical diagnosis

Machine learning can assist in illness diagnosis. Many doctors utilize speech recognition chatbots to find trends in their patients' complaints[c].

Examples of real-life medical diagnosis:

- Assisting in the formulation of a diagnostic or making therapy recommendations
- Machine learning is used in oncology and pathology to identify malignant tissue.
- Examine the bodily fluids

Face recognition software and machine learning are combined to scan patient photographs and find traits that correspond with uncommon genetic illnesses in the case of rare diseases[19].

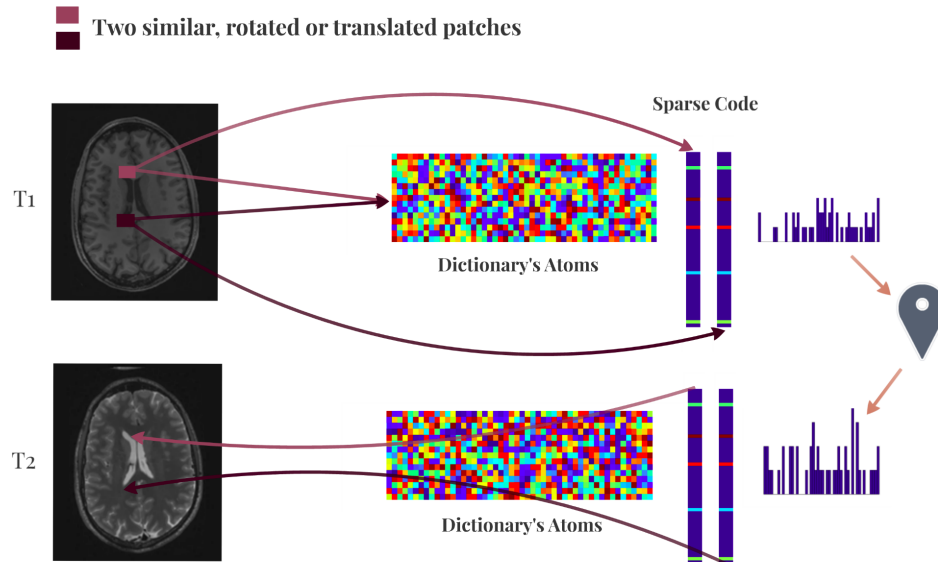


Figure II.4: Medical Diagnosis Machine Learning Discount[22].

II.3.4. Statistical arbitrage

Arbitrage is a finance term for an automated trading method that is used to handle a large number of securities. A trading algorithm is used to analyze a group of securities using economic data and correlations[19].

Statistical arbitrage in the real world:

- Algorithmic trading examines the microstructure of a market.
- Analyze massive amounts of data
- Find arbitrage chances in real time.

Machine learning improves the arbitrage strategy's performance[19].



Figure II.5: Defined & Strategies[23].

II.3.5. Predictive analytics

Machine learning may divide accessible data into categories, which are subsequently defined by analyst-specified rules. The analysts can determine the likelihood of a defect after the categorization is complete[19].

Examples of predictive analytics in practice:

- Identifying whether or not a transaction is fraudulent
- Improve prediction systems that determine the likelihood of a problem.

One of the most promising applications of machine learning is predictive analytics. Everything from product creation to real estate pricing may benefit from it[19].

II.4. Types of Machine Learning Systems

II.4.1. Supervised learning

In supervised learning, you input the algorithm training data that includes the desired answers, known as labels.[16]

A frequent supervised learning task is classification. The spam filter is a good example of this: it is taught how to recognize new emails by looking at a huge number of sample emails and categorizing them (spam or ham).[16]

Another typical purpose is to forecast a target numeric value, like as the price of a car, using a set of predictors (mileage, age, brand, etc). This sort of work is known as regression. You

must feed the system with a large number of car instances containing both predictors and labels in order to train it (i.e., their prices).[16]

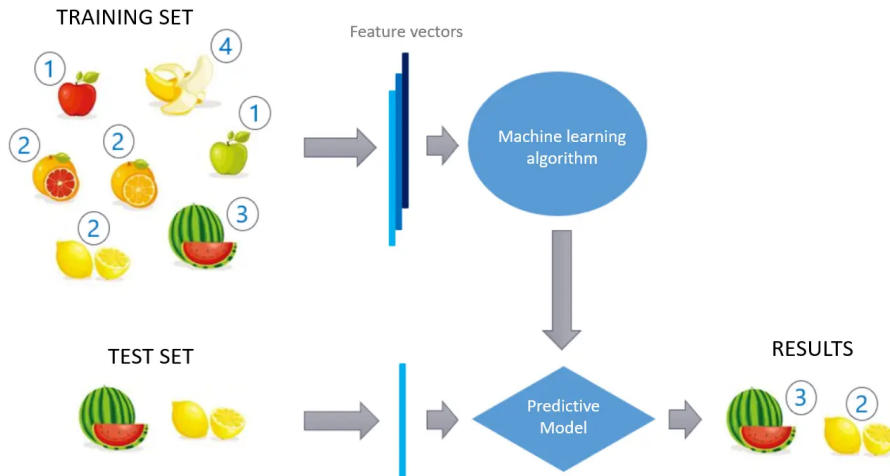


Figure II.6: Example about learning from data[24].

II.4.2. Unsupervised learning

The training data in unsupervised learning is unlabeled[16]. Assume you have a lot of information on your blog's visitors. To find groups of similar visitors, you might wish to use a clustering method. You never tell the algorithm which group a visitor belongs to; it discovers such relationships on its own. For example, it could be noted that 40% of your visitors are men who prefer comic books and read your site in the evenings, while 20% are youthful science-fiction fans who visit on weekends, and so on[13]. When using a hierarchical clustering algorithm, each group may be subdivided into smaller groups. This may assist you in tailoring your postings to each audience[13].

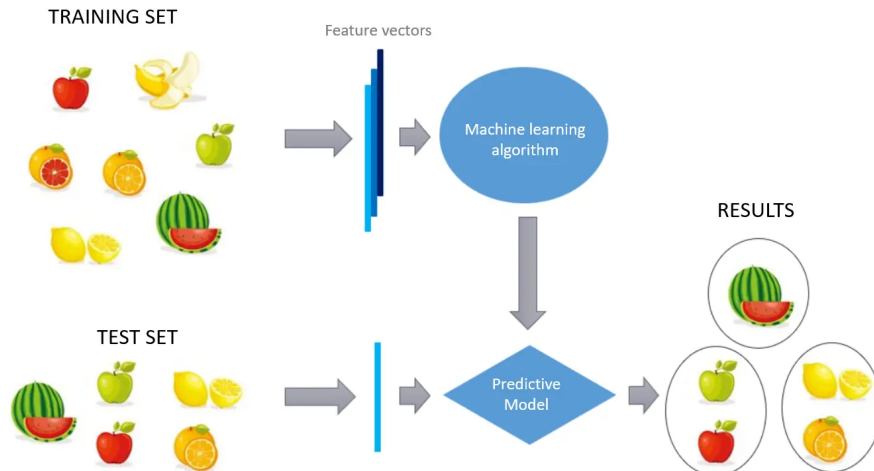


Figure II.7: Example about unsupervised learning [25].

II.4.3. Semi Supervised learning

Some algorithms can handle partially labeled training data, which often includes a large amount of unlabeled data and a small amount of labeled data. Semi-supervised learning is what it's termed[16].

This may be seen in photo-hosting services like Google Photos. When you submit all of your family photographs to the site, it will instantly detect that person A appears in images 1, 5, and 11, while person B appears in photos 2, 5, and 7. This is where the algorithm becomes unsupervised (clustering). All that's left for you to do now is inform the system who these folks are. There is just one label per person, with a total of four labels, and it can name everyone in every shot, which is great for searching[16].

The majority of semi-supervised learning methods combine unsupervised and supervised learning techniques. Deep belief networks (DBNs), for example, are made up of stacked unsupervised components termed restricted Boltzmann machines (RBMs). Unsupervised learning techniques are used to train RBMs sequentially, and then supervised learning techniques are used to fine-tune the entire system[16].

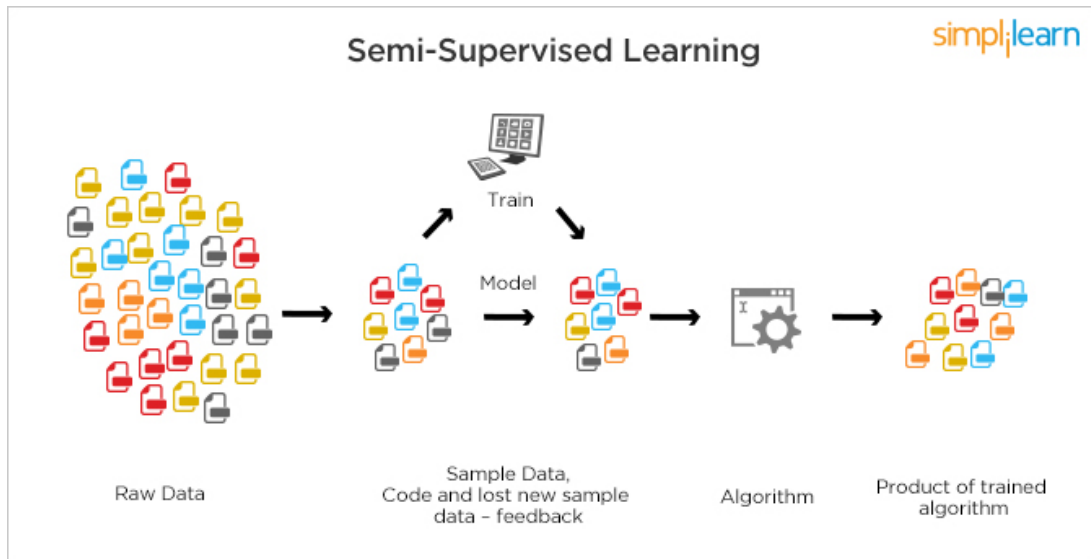


Figure II.8: Semi-supervised learning [26].

II.4.4. Reinforcement learning

Reinforcement Learning takes on a whole new perspective. In this context, the learning system is referred to as an agent since it can monitor the environment, select and conduct actions, and receive rewards in exchange (or penalties in the form of negative rewards). It must then figure out for itself what the optimal technique is for getting the highest reward over time, which is termed a policy. When an agent is in a certain scenario, a policy specifies the action it should do[16].

Many robots, for instance, use Reinforcement Learning algorithms to learn how to walk. DeepMind's AlphaGo software is another outstanding example of Reinforcement Learning: in May 2017, it made news by defeating world champion Ke Jie in the game of Go. It developed its winning strategy after studying millions of games and playing numerous games against itself. It's worth noting that learning was disabled during the games versus the champion; AlphaGo was just following the policy it had learnt[16].

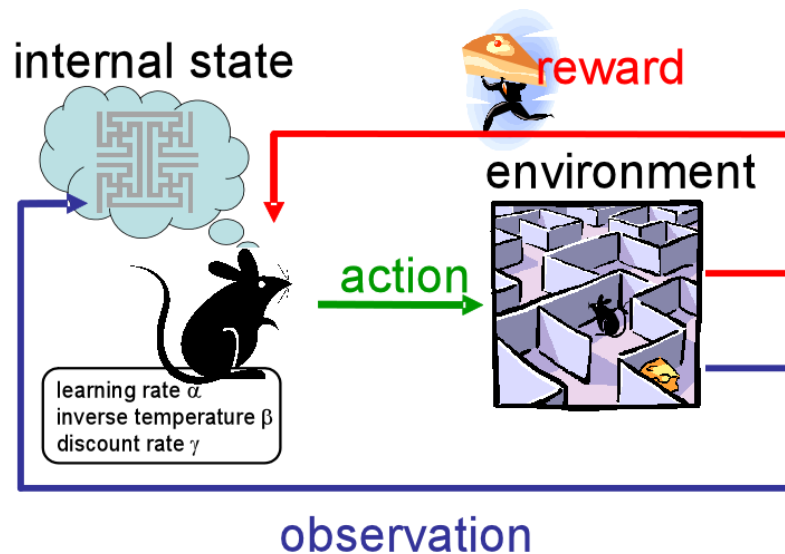


Figure II.9: Reinforcement learning [27].

II.5. Conclusion

Machine learning (ML) is the study of ways that leverage data to improve performance on some set of tasks. Machine learning algorithms create a model based on training data to make predictions or judgments without having to be explicitly programmed to do so. Data and neural networks are used in certain machine learning implementations to replicate the functioning of a biological brain. Machine learning is the process of computers learning from data in order to do certain jobs. For basic jobs, algorithms may be programmed that inform the machine how to carry out all of the steps required to address the problem at hand; the computer does not need to learn anything. In practice, assisting the computer in developing its own algorithm rather than having human programmers explain each required step can prove to be more productive.

Chapter III
Introduction to visible light
communication

III.1. Introduction

Over the last decade, the exponential growth of mobile devices and wireless services has produced a huge need for radio frequency-based technologies. Meanwhile, the increasing adoption of more cost-effective and efficient LED light bulbs has changed the lighting industry. Visible Light Communication (VLC) is an LED-based technology with a free spectrum and high data rate that might potentially augment current radio frequency standards in this respect[28].

III.2. VLC:

Visible light communication (VLC) is a type of data transfer that employs visible light between the wavelengths of 400 and 800 THz (780–375 nm). VLC is an optical wireless communications technology subset.

The technique utilizes conventional fluorescent bulbs (not special communications equipment) to carry signals at 10 kbit/s across short distances, or LEDs for up to 500 Mbit/s. Over lengths of 1–2 kilometers (0.6–1.2 mi), systems like RONJA may transmit at full Ethernet speed (10 Mbit/s)[d]. Signals from light sources are received by specially built electrical devices that usually comprise a photo-diode[e]. In other circumstances, though, a mobile phone camera or a digital camera can suffice.[f] In these devices, the image sensor is really an array of photodiodes (pixels), which in some situations may be preferable to a single photodiode. Multi-channel (down to 1 pixel Equals 1 channel) or spatial awareness of various light sources may be provided by such a sensor[29].

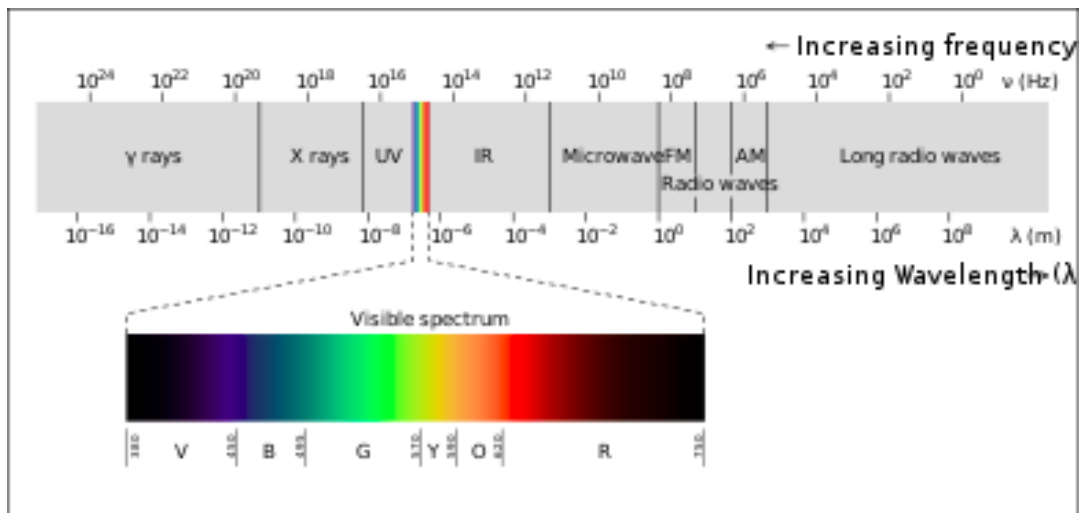


Figure III.1: Visible light is only a small portion of the electromagnetic spectrum.[30].

III.3. VLC architecture

Light is used to transfer information in Visible Light Communication. Furthermore, the purpose of VLC apps is to give both lighting and communication. As a result, VLC systems will always contain light transmitting and receiving components. LEDs are employed as transmitters in the great bulk of the literature on this subject. These LEDs are used to communicate data by modulating the intensity of light. Photo-sensors on the receiver side are in charge of immediately catching this light (Direct Detection) and transforming it into a data stream [28]. The type of LED has an influence on the operation of a VLC system since lighting illumination brightness is not altered by light manipulation when transferring information in VLC[6].

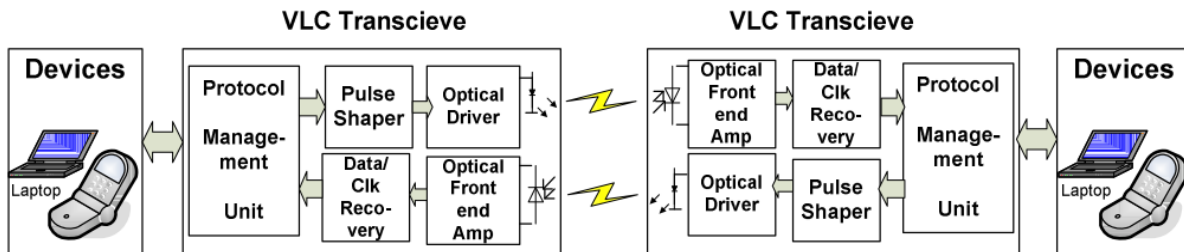


Figure III.2: Architecture of a full-duplex VLC communication system.[31].

III.3.1. Transmitter

VLC transition sources include LEDs and lasers. When both communication and lighting must be accomplished with a single device, the LED should be employed. One of the most appealing alternatives for use as a VLC source is white light based on LEDs and wavelength converters. Trichromatic light production is the most frequent way for producing white light using LEDs (red, green, and blue). The large bandwidth and hence high data rates of RGB LEDs make them ideal for white light production. The RGB LEDs disadvantage is its high complexity and modulation difficulties. The optical wireless channel has been characterized using a variety of techniques[6].Based on the channel model, the suitable LED is chosen[32].

III.3.1.1. LED

LED light bulbs have become the most common medium for Visible Light Communication due to factors such as pricing. Furthermore, LED light bulbs grew increasingly popular, integrating numerous contexts where using light as a means of communication would be helpful.[28]

LEDs may switch on and off in a fraction of a second. We can send information by producing a frequency by turning it on and off millions of times per second. When an LED is turned on, it sends a 1 bit; when it is turned off, it transmits a 0 bit. The frequency changes are so fast that the human eye cannot see them because it does not perceive flicker and only sees constant light. In terms of bandwidth, this equates to 1 Gbps, compared to roughly 100 Mbps for Wi-Fi.[28]

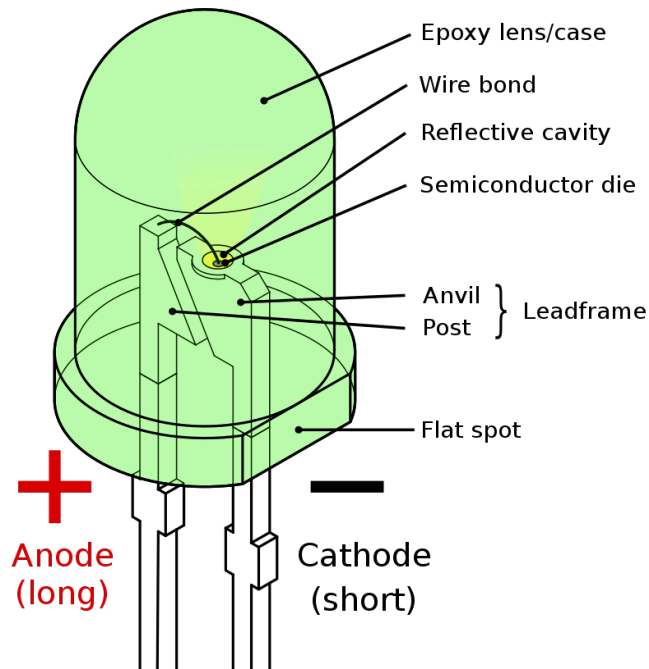


Figure III.3: Basic representation of a standard LED[33].

- **Phosphor Converted LEDs (pc-LEDs)** : PC-LEDs are frequently utilized since they are simple and inexpensive. They are made up of a blue LED chip with a phosphor layer that converts a portion of the blue light to green, yellow, and red while emitting a percentage of the blue light as white light. Due to the delayed reaction of phosphorus, this LED has a restricted band[28].
- **Multi-chip LEDs**: This LED's structure is made up of three or more chips that emit distinct colored lights. In order to generate white light, the various chips normally output RGB colors. The ability to adjust the colors emitted by the intensity of each chip is a significant benefit of this form of LED. It's worth noting that a sort of modulation called Color-Shift Keying was designed specifically for this type of LED[28].

- **Organic LEDs (OLEDs):** A series of thin organic layers placed between two conductors make up this type of LED. An electric current causes light to be emitted. They're common in smartphone screens[28].
- **μ -LEDs:** μ -LEDs are commonly used in displays to provide high-density parallel transmission at very fast rates[28].

III.3.2. Receiver

The data is extracted from the modulated light beam by the VLC receiver, which converts the light into an electrical signal that is decoded by the built-in decoder. The receiver's performance influences the VLC system's performance; the detected light is not only that of the transmitter, but also that of other light sources (natural or artificial), causing significant interference. Using an optical filter to eliminate undesirable spectral components can enhance the performance of the VLC receiver. Due to its great sensitivity, the photodiode is the most often used light sensor. Solar cells can be used to receive both solar electricity and visible light communication (VLC) signals at the same time. Furthermore, the modulated VLC optical signal may be transformed to an electrical signal without the need for additional power[34].

III.3.2.1. Photodiode

A photodiode is a device that transforms light into electricity. It operates on the photo-conduction principle, whereas LEDs operate on the electro-luminescence basis. A photodiode is a photodetector that transforms light into current or voltage[35].

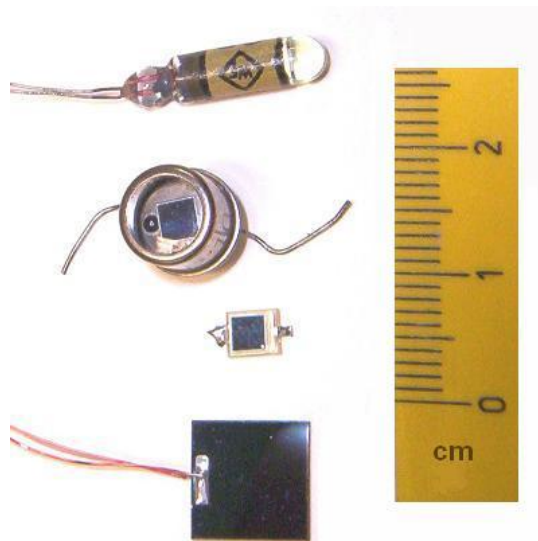


Figure III.4: Photodiode size and example[36].

III.3.3. Transmission Channel

Intensity Modulation / Direct Detection systems make up the majority of wireless optical systems. Information is transmitted by the intensity of the optical signal rather than frequency or phase in an IM/DD system. A light emitting diode performs the conversion between the electrical signal and the optical intensity $x(t)$ (LED). After then, the optical wave travels via the wireless optical channel. It is important to do the reverse conversion upon reception in order to return to the electrical domain. The photodiode is responsible for this function. Direct detection is achieved by the photodiode, which generates a photo-current $y(t)$ proportionate to the received optical intensity. The receiver's principal goal is to deduce the information contained in $x(t)$ from the signal y received (t)[37].

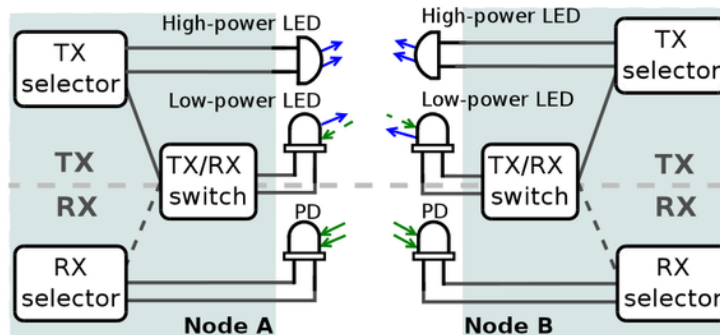


Figure III.5: Open-VLC 1.0 transceivers[38].

III.4. Applications of VLC communications

III.4.1. Indoor

III.4.1.1. Mobile connectivity (Li-Fi)

Harald Haas was the first to create the phrase "light fidelity" in 2011. (Li-Fi). Li-Fi is a bi-directional, fully linked, visible light wireless communication technology that is similar to Wi-Fi, which communicates using radio frequency. Wi-Fi frequencies can interfere with other RF signals, such as pilot navigational equipment signals in airplanes. As a result, in situations where electromagnetic radiation is sensitive (such as airplanes), Li-Fi may be a preferable option. A Li-Fi network can also help with the Internet of Things (IoT). Li-Fi can achieve speeds of up to 10 Gbits/s, which is 250 times faster than super-fast broadband[32].

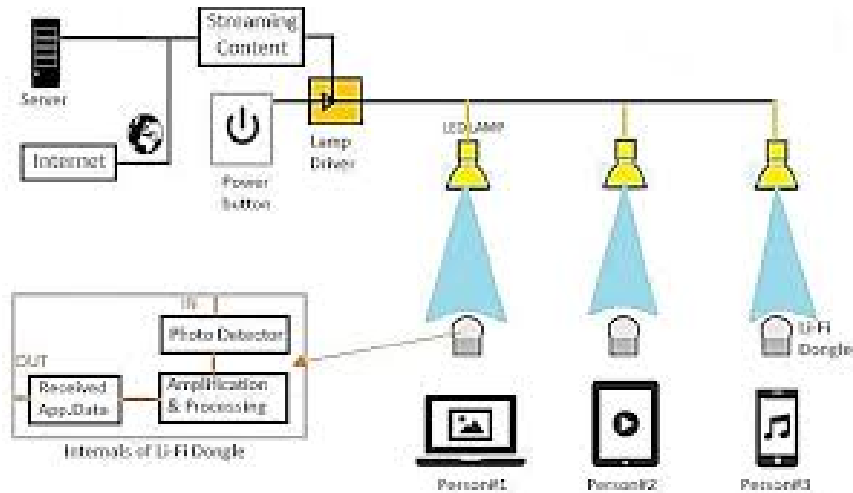


Figure III.6: The Working Of Li-Fi Communication[39].

III.4.1.2. Hospitals and Healthcare

Electromagnetic wave sensitive places (such as MRI scanners) in hospitals are likely to transition to VLC since it does not interfere with other devices' radio waves. A robot dubbed HOSPI was proposed for use in hospital transportation. HOSPI's control system was improved with the use of VLC, which was put in a building, and the robot's navigational sensors[32].

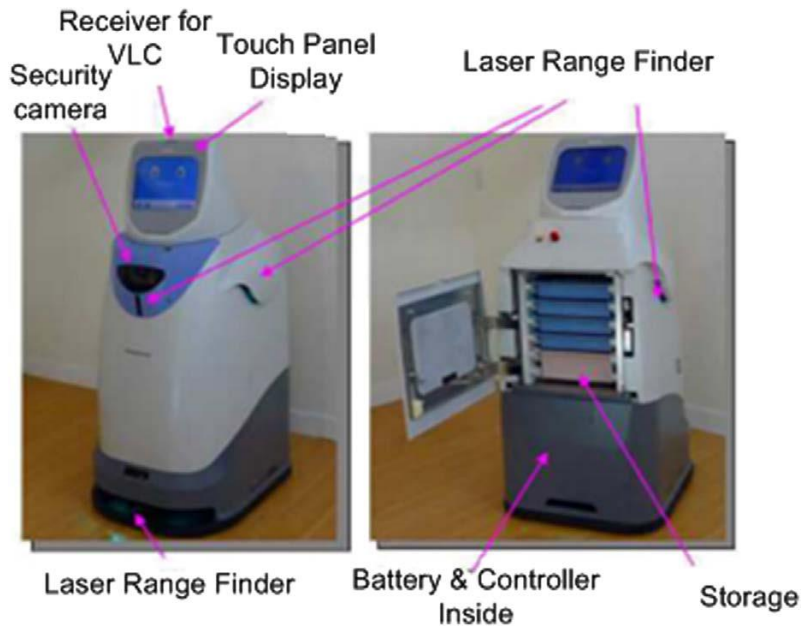


Figure III.7: Hospital delivery robot manufactured by Panasonic[32].

III.4.1.3. Wi-Fi Spectrum Relief

The increased demand for wireless data is outstripping Area networks. Fi's VLC has the ability to give faster data speeds than Wi-Fi. Because radio frequency (RF) components and antenna systems are eliminated, this may be done at a minimal cost.[6]

III.4.1.4. Aircraft

In aircraft passenger cabins, radio waves are unwelcome. LEDs are already utilized for illumination and can be used to give multi-media services instead of cables. This lowers the cost and weight of an aircraft's construction.[6]

III.4.1.5. Positioning

To conduct high-precision location using VLC, a receiver must gather up signals from LEDs in a room and determine the distance between them using different methods. RSS (received signal strength), which is widely employed in radio frequency systems, is one of these strategies. The signal intensity is reduced when the distance between the transmitters and receivers is increased. Obstacles that block or reflect waves may interfere with the RSS, decreasing the method's accuracy. Calculating the Time of Arrival is another extensively used technique (TOA). This approach, however, necessitates the transmission of rigorously synced signals between the transmitter and receiver, which may need the purchase of more expensive features. Finally, there's a technique that takes the Angle of Arrival into account (AOA). Because it necessitates a direct Line of Sight, this approach is not widely used in radio frequency systems (LOS). VLC systems, on the other hand, require LOS for communication, hence AOA approaches are also viable.[28]

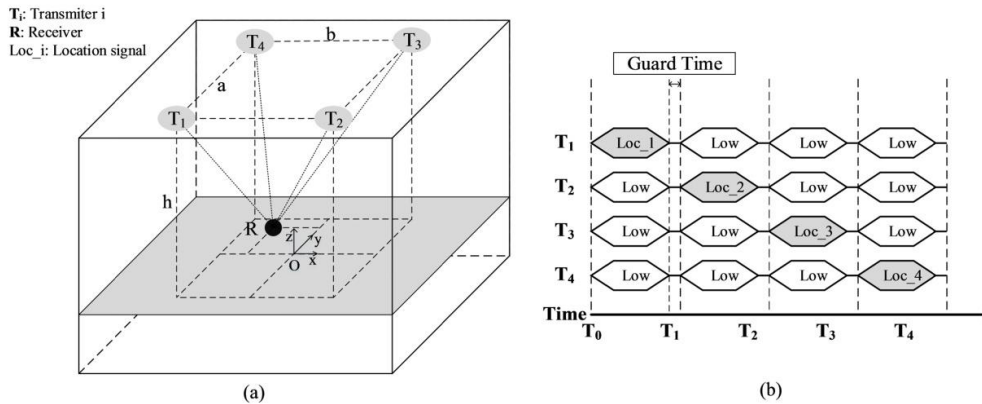


Figure III.8: Basic diagram on indoor positioning using VLC[6]

III.4.2. Outdoor

III.4.2.1. Vehicles and Transportation

The widespread use of LEDs in traffic applications presents a number of opportunities for vehicle-to-vehicle communication (VLC) systems. Since VLC links are visible, installation of roadside equipment is much easier. V2I and V2V communications using radio technology can be used simultaneously with VLC. VLC provides an additional feature if the receiver incorporates an image sensor or a camera.[40]

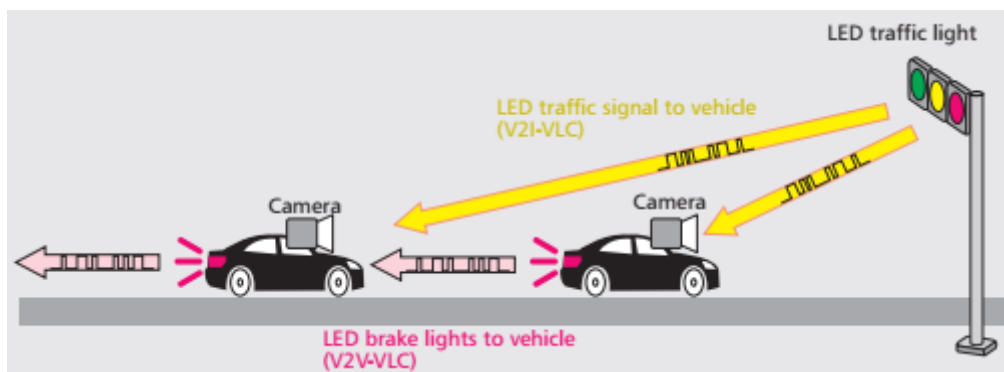


Figure III.9: V2V and I2V communication[40]

III.4.2.2. Underwater Communication

Because of its high conductivity, RF waves do not travel well in saltwater. As a result, underwater communication networks should adopt VLC communication. Another underwater communication use of the VLC is the Untethered Remotely Operated Vehicle (UTROV). UTROV may be used for a variety of tasks, including ocean observatory maintenance and ship deployment[32].

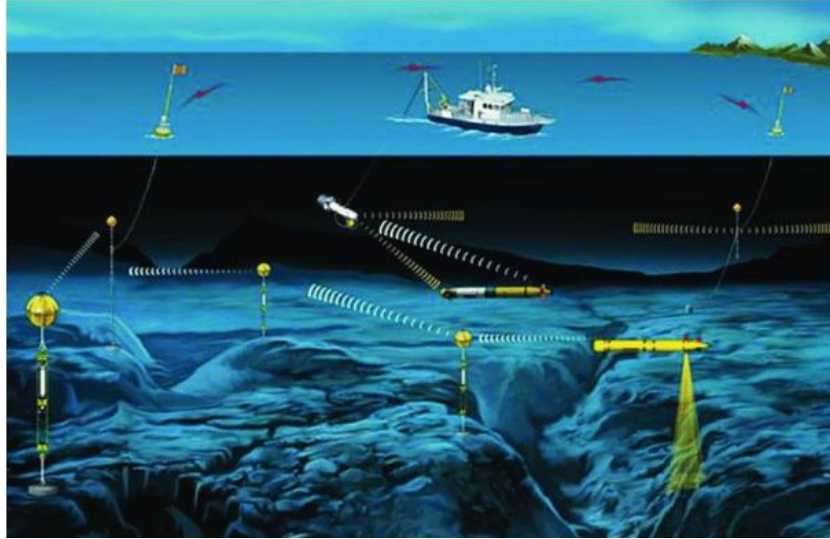


Figure III.10: Underwater Communication [32]

III.5. Conclusion

We outlined VLC and its working principle in this chapter, as well as its design, the best LEDs to use, optical link categorization, how to decrease noise and interference from other light sources, and its present possible applications. It's a terrific way to supplement current wireless infrastructure since it provides better performance, especially in close-quarters areas like workplaces and houses. Indoor location, underwater communication, and vehicle communication systems are just a few examples of applications that can make use of visible light. In conclusion, VLC is a large field of study that also piques the interest of scholars. Even yet, further investigation is required, which should occur in the next few years, given the growing popularity of the field and the expanding application of concepts such as the Internet of Things and Smart Lighting.

Chapter IV

Application

IV.1. Introduction

In this chapter, we split our project in two sections, the first is a simulation part, which we are going to make an intelligent system that detects objects, analysis the location and take decisions. The second part is the experimental part, where we are going to build semi-controller robot with VLC and finally we switch into complete self-controller robot. That's actually will be our final goal in this project

IV.2. Simulation part

Our first goal is create a smart system that can train the images from the camera and take decision based on that.

I started search about a good simulation idea or project that it's near from our needs. we found a simulation from Udacity in this website <https://georgeerol.github.io/>

In this simulation, I'll use the Udacity rover simulator to autonomously drive a rover which will look for samples in a terrain area. The simulator has been modeled after the NASA sample return challenge. The simulator has two modes – Training mode, where we can drive the rover using the keyboard and collect training data and Autonomous mode, where we will be driving the rover autonomously by modifying the functions present in the python driver codes. The **Github** repository has provided all the required python template codes. We have to implement our algorithm by modifying the codes

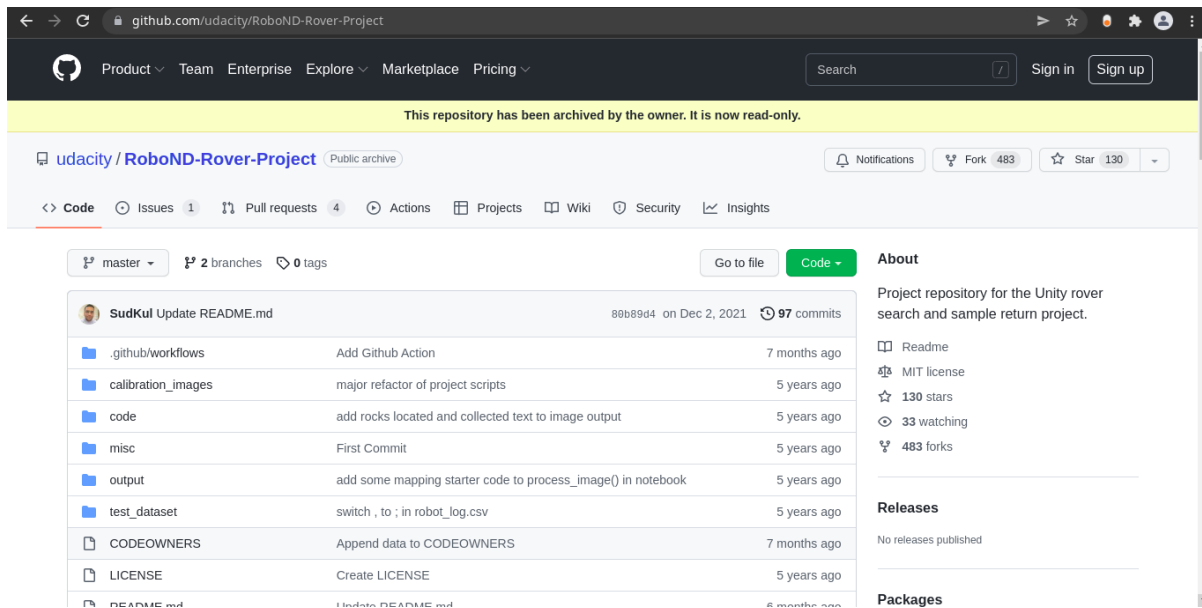


Figure IV.1: Github repo of RoboND-Rover-Project

- **Udacity**

Udacity, Inc. is an American for-profit educational organization founded by Sebastian Thrun, David Stavens, and Mike Sokolsky offering MOOC [41][42][43]



Figure IV.2: Udacity logo [44]

This project is based on NASA's sample return challenge and will provide you with hands-on experience with the three fundamental parts of robotics: perception, decision making, and actuation. This project will be completed in a Unity game engine-based simulated environment.[45]

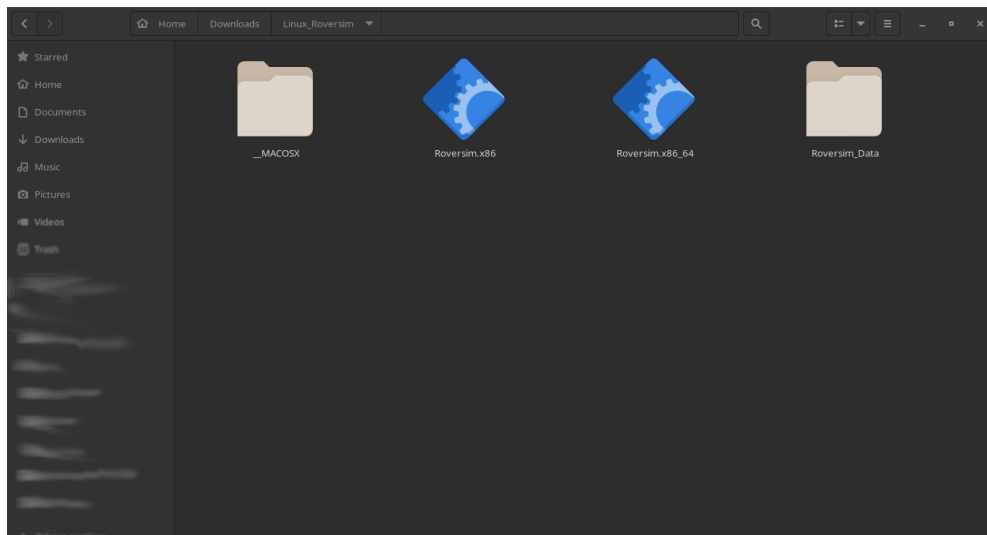


Figure IV.3: File of the simulation after the extraction

After we extract the file, we click on the Icon of the simulation to run. It will show a window like that to set up configuration for your preferences and ability

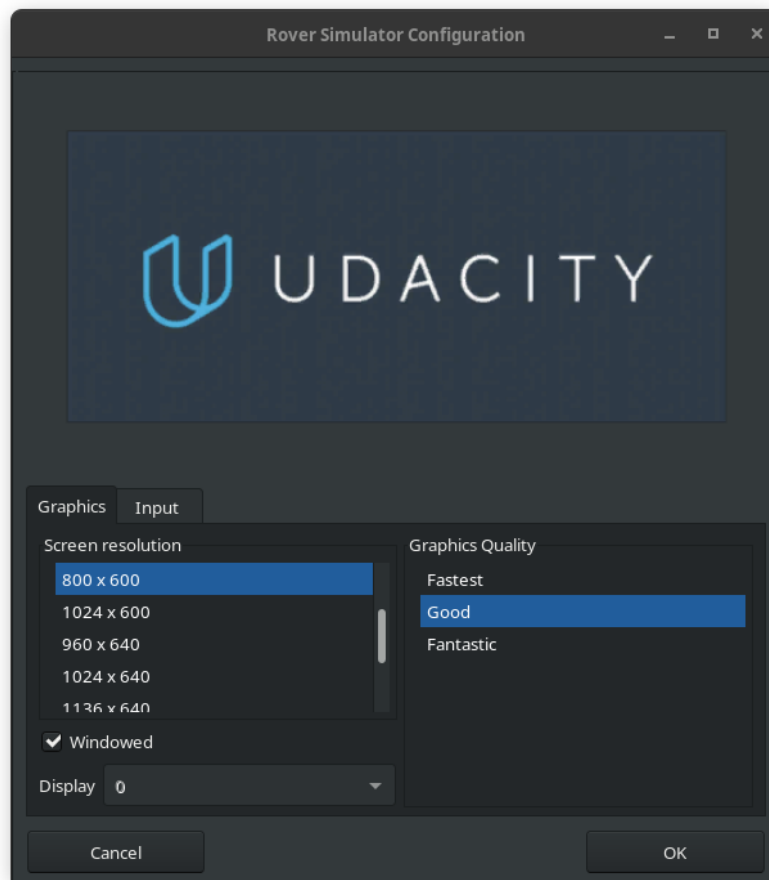


Figure IV.4: Rover Simulator Configuration

After that, the simulation will run. There are two mode in the Training Mode and Autonomous mode.

The Training mode is manual controller using the the arrow keys and the mouse and you are full of control.

The Autonomous mode when we'll install the machine learning in its brain .

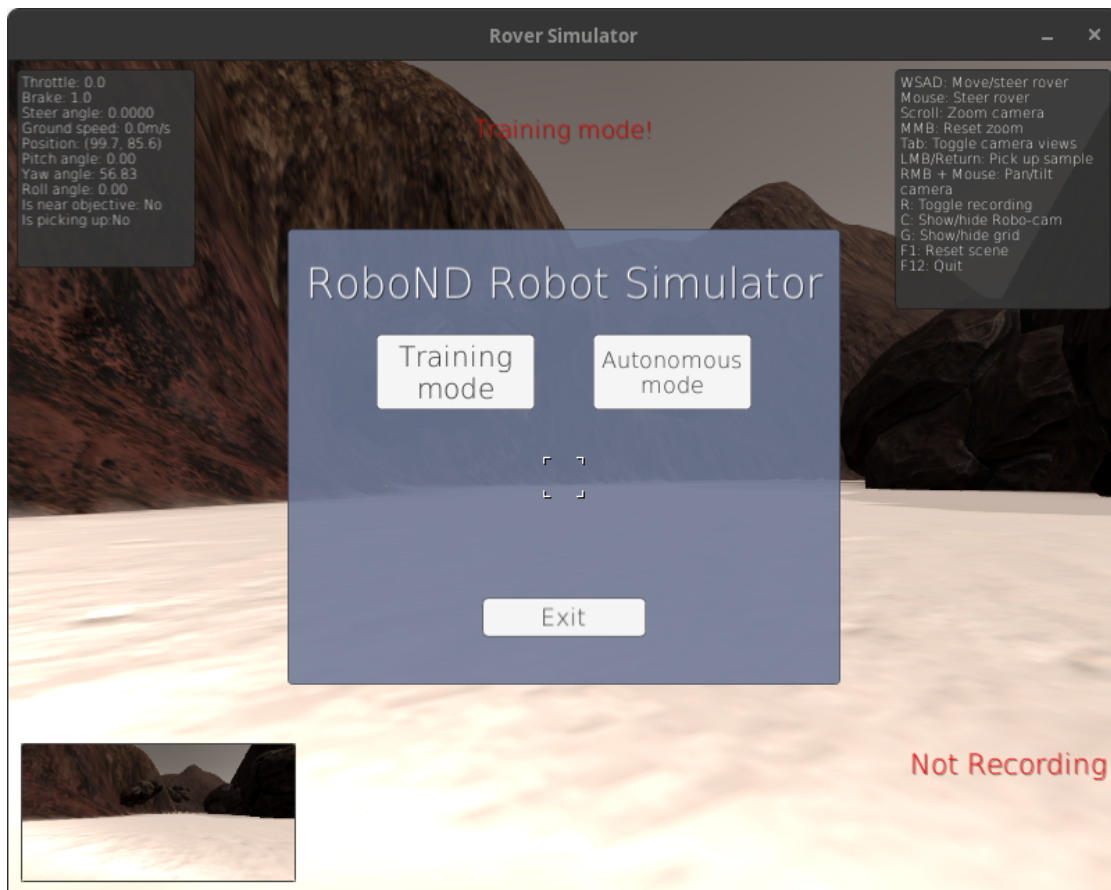


Figure IV.5: Simulator window

Our first step is using the training mode to discover what we are deal with, the environment , the elements and the controller and base on that we'll build a system to do our tasks as we want.

Chapter IV : Application



Figure IV.6: the start of the simulation

The simulation window shows our robot details on the top left about the location , top right we can see the key to do a certain action like quit by pressing F12, bottom left shows the camera view of the robot, and bottom right shows if we are recording or not.

To start recording press the key “r” and that will open a window asked you where you want to save the data from the recording.

Chapter IV : Application

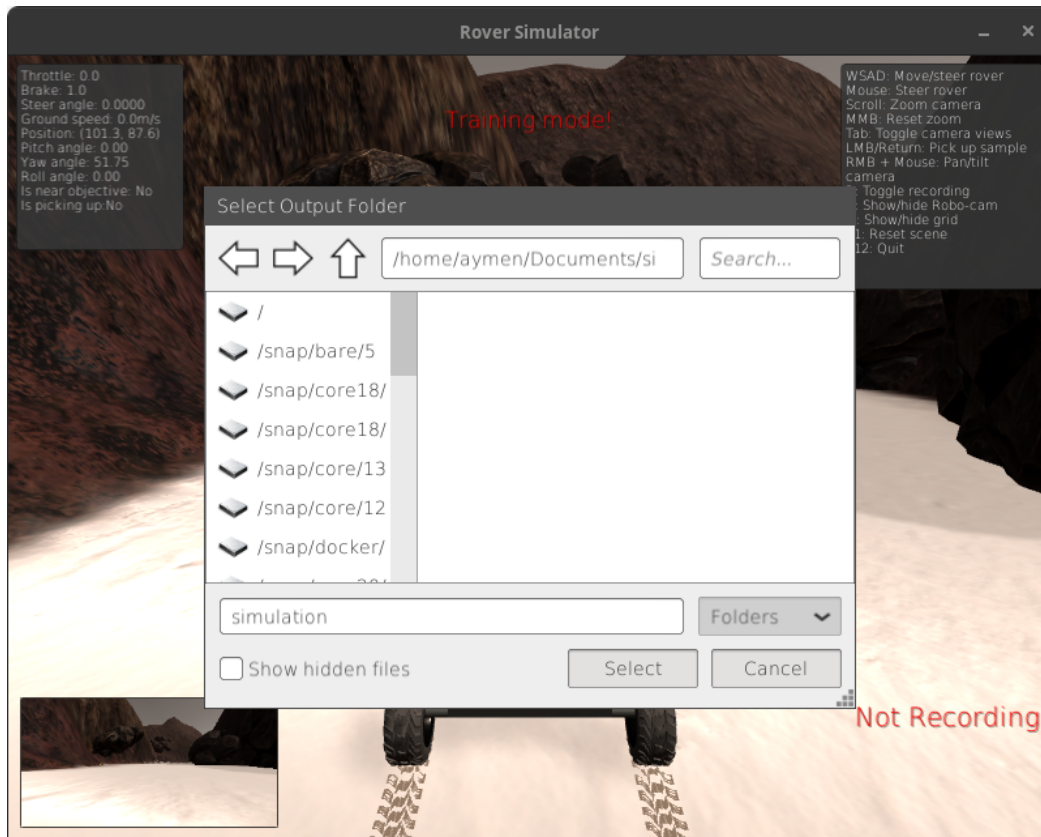


Figure IV.7: set the output

We start recording.

Chapter IV : Application



Figure IV.8: Collecting items in the environment

After we finish recording we close the simulation and we go to the document where we saved our output, we'll find two main document, the first is a folder contains all the image that has been recorded , the other is csv file contain a table of all of our image and other data about the robot

Chapter IV : Application

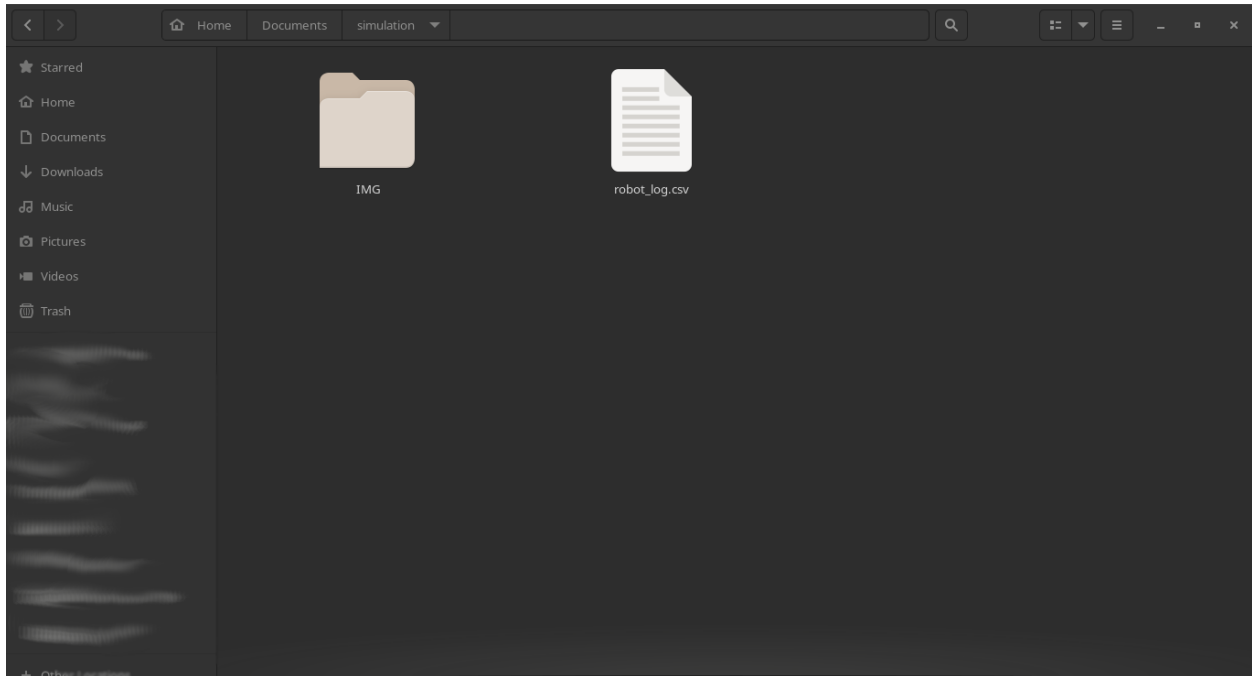


Figure IV.9: Output

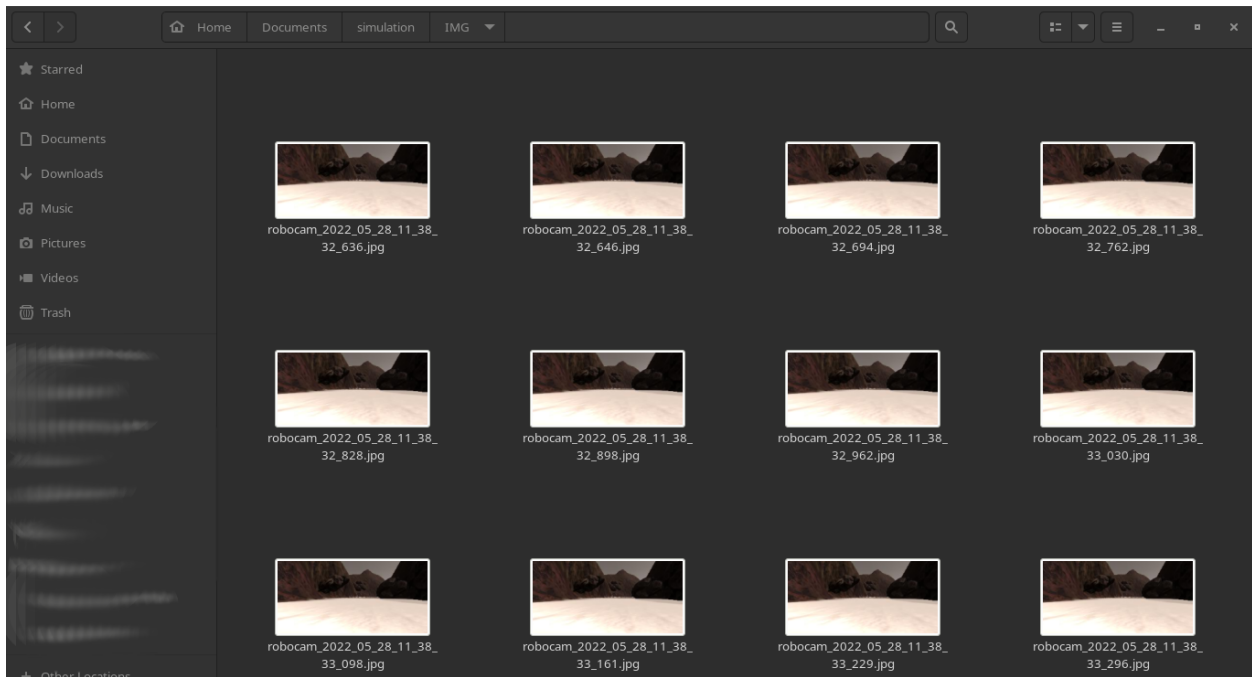


Figure IV.10: IMG folder

Chapter IV : Application

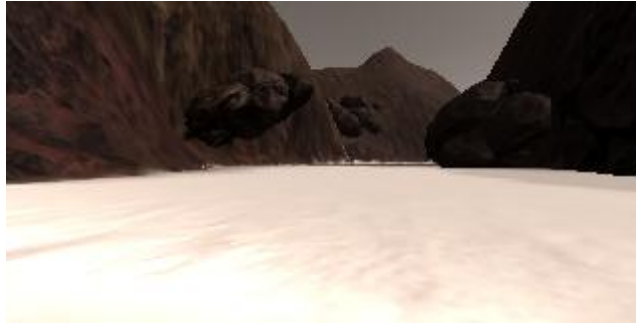
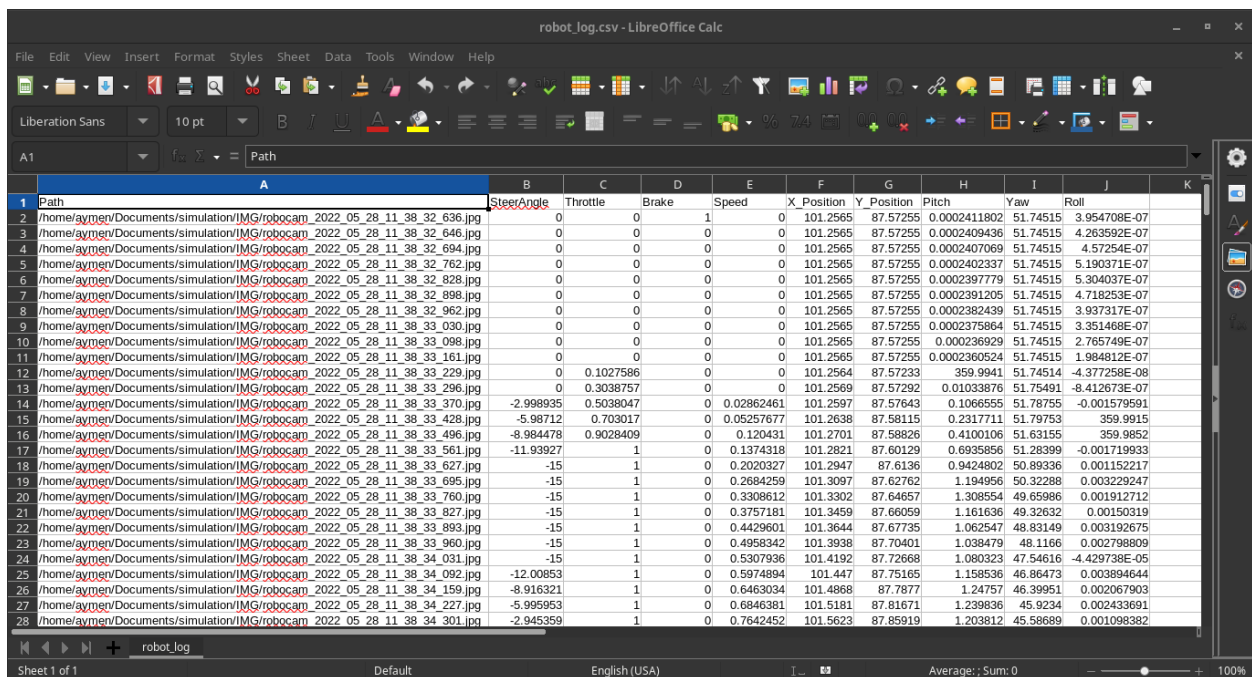


Figure IV.11: image from IMG folder



Path	SteerAngle	Throttle	Brake	Speed	X Position	Y Position	Pitch	Yaw	Roll
/home/aymen/Documents/simulation/IMG/robocam_2022_05_28_11_38_32_636.jpg	0	0	1	0	101.2565	87.57255	0.0002411802	51.74515	3.954708E-07
/home/aymen/Documents/simulation/IMG/robocam_2022_05_28_11_38_32_646.jpg	0	0	0	0	101.2565	87.57255	0.0002409436	51.74515	4.283592E-07
/home/aymen/Documents/simulation/IMG/robocam_2022_05_28_11_38_32_694.jpg	0	0	0	0	101.2565	87.57255	0.0002407069	51.74515	4.57254E-07
/home/aymen/Documents/simulation/IMG/robocam_2022_05_28_11_38_32_762.jpg	0	0	0	0	101.2565	87.57255	0.0002402337	51.74515	5.190371E-07
/home/aymen/Documents/simulation/IMG/robocam_2022_05_28_11_38_32_828.jpg	0	0	0	0	101.2565	87.57255	0.0002397779	51.74515	5.304037E-07
/home/aymen/Documents/simulation/IMG/robocam_2022_05_28_11_38_32_898.jpg	0	0	0	0	101.2565	87.57255	0.0002391205	51.74515	4.718253E-07
/home/aymen/Documents/simulation/IMG/robocam_2022_05_28_11_38_32_962.jpg	0	0	0	0	101.2565	87.57255	0.0002382439	51.74515	3.937317E-07
/home/aymen/Documents/simulation/IMG/robocam_2022_05_28_11_38_33_030.jpg	0	0	0	0	101.2565	87.57255	0.0002375864	51.74515	3.351468E-07
/home/aymen/Documents/simulation/IMG/robocam_2022_05_28_11_38_33_098.jpg	0	0	0	0	101.2565	87.57255	0.000236929	51.74515	2.765749E-07
/home/aymen/Documents/simulation/IMG/robocam_2022_05_28_11_38_33_161.jpg	0	0	0	0	101.2565	87.57255	0.0002360524	51.74515	1.984812E-07
/home/aymen/Documents/simulation/IMG/robocam_2022_05_28_11_38_33_229.jpg	0	0.1027586	0	0	101.2564	87.57233	359.9941	51.74514	-4.377258E-08
/home/aymen/Documents/simulation/IMG/robocam_2022_05_28_11_38_33_296.jpg	0	0.3038757	0	0	101.2569	87.57292	0.01033876	51.75491	-8.412673E-07
/home/aymen/Documents/simulation/IMG/robocam_2022_05_28_11_38_33_370.jpg	-2.998935	0.5038047	0	0.02862461	101.2597	87.57643	0.1066555	51.78755	-0.001579591
/home/aymen/Documents/simulation/IMG/robocam_2022_05_28_11_38_33_428.jpg	-5.98712	0.703017	0	0.05257677	101.2638	87.58115	0.2317711	51.79753	359.9915
/home/aymen/Documents/simulation/IMG/robocam_2022_05_28_11_38_33_496.jpg	-8.984478	0.9028409	0	0.120431	101.2701	87.58826	0.4100106	51.63155	359.9852
/home/aymen/Documents/simulation/IMG/robocam_2022_05_28_11_38_33_561.jpg	-11.93927	1	0	0.1374318	101.2821	87.60129	0.6935856	51.28399	-0.001719933
/home/aymen/Documents/simulation/IMG/robocam_2022_05_28_11_38_33_627.jpg	-15	1	0	0.2020327	101.2947	87.6136	0.9424802	50.89336	0.001152217
/home/aymen/Documents/simulation/IMG/robocam_2022_05_28_11_38_33_695.jpg	-15	1	0	0.2684259	101.3097	87.62762	1.194956	50.82288	0.003229247
/home/aymen/Documents/simulation/IMG/robocam_2022_05_28_11_38_33_760.jpg	-15	1	0	0.3308612	101.3302	87.64657	1.308554	49.65986	0.001912712
/home/aymen/Documents/simulation/IMG/robocam_2022_05_28_11_38_33_827.jpg	-15	1	0	0.3757181	101.3459	87.66059	1.161636	49.32632	0.00150319
/home/aymen/Documents/simulation/IMG/robocam_2022_05_28_11_38_33_893.jpg	-15	1	0	0.4429601	101.3644	87.67735	1.062547	48.83149	0.003192675
/home/aymen/Documents/simulation/IMG/robocam_2022_05_28_11_38_33_960.jpg	-15	1	0	0.4958342	101.3938	87.70401	1.038479	48.1166	0.002798809
/home/aymen/Documents/simulation/IMG/robocam_2022_05_28_11_38_34_031.jpg	-15	1	0	0.5307936	101.4192	87.72668	1.080323	47.54616	-4.429738E-05
/home/aymen/Documents/simulation/IMG/robocam_2022_05_28_11_38_34_092.jpg	-12.00853	1	0	0.5974894	101.447	87.75165	1.158536	46.86473	0.003894644
/home/aymen/Documents/simulation/IMG/robocam_2022_05_28_11_38_34_159.jpg	-8.916321	1	0	0.6463034	101.4868	87.7877	1.24757	46.39951	0.002067903
/home/aymen/Documents/simulation/IMG/robocam_2022_05_28_11_38_34_227.jpg	-5.995953	1	0	0.6846381	101.5181	87.81671	1.239836	45.9234	0.002433691
/home/aymen/Documents/simulation/IMG/robocam_2022_05_28_11_38_34_301.jpg	-2.945359	1	0	0.7842452	101.5623	87.85919	1.203812	45.58689	0.001098382

Figure IV.12: robot_log.csv

At this point we have been recorded our data and now it's time to train it but now I'll give you a brave introduction about what tools we'll use for that.

- **Jupyter Notebook** The Junketer Notebook is the original web application for creating and sharing computational documents. It offers a simple, streamlined, document-centric experience[46].



Figure IV.13: Jupyter Logo [47]

Chapter IV : Application

- **Python** Python is the main programming language that we'll use in our project because it's so simple and it's the language of the artificial intelligence.
- **Bash** is the Unix shell and we use it in linux terminal.

I use linux as my primary Operating system , my linux distro is Debian version 11 so I'll be using the Terminal to access to Jupyter notebook Of course it would be easier for windows users than linux to set up all the settings needed to start that project.

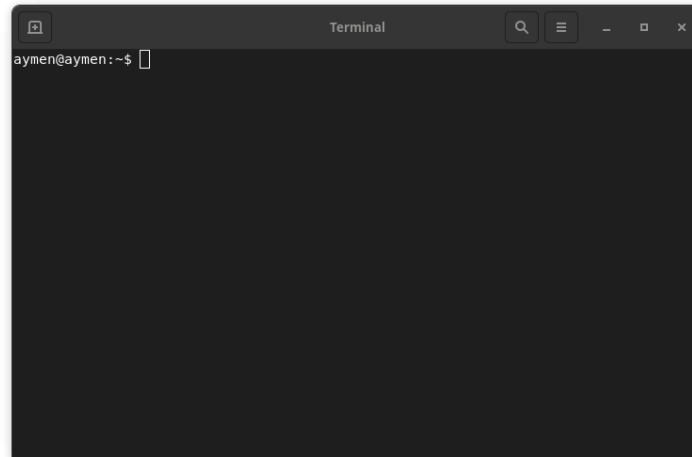


Figure IV.14: Terminal window

Our Jupyter notebook software needed to run using the “su” command; during a login session, the su command is used to switch users. Su becomes the superuser if it is invoked without a username. The optional argument - can be used to create an environment that is comparable to what a user would see if they signed in directly.[48]

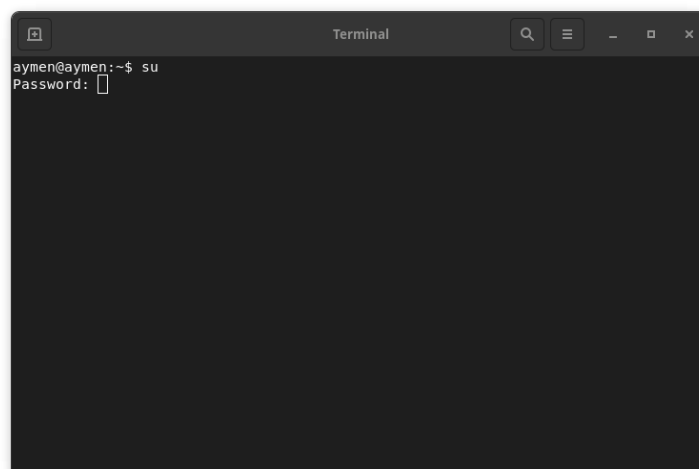
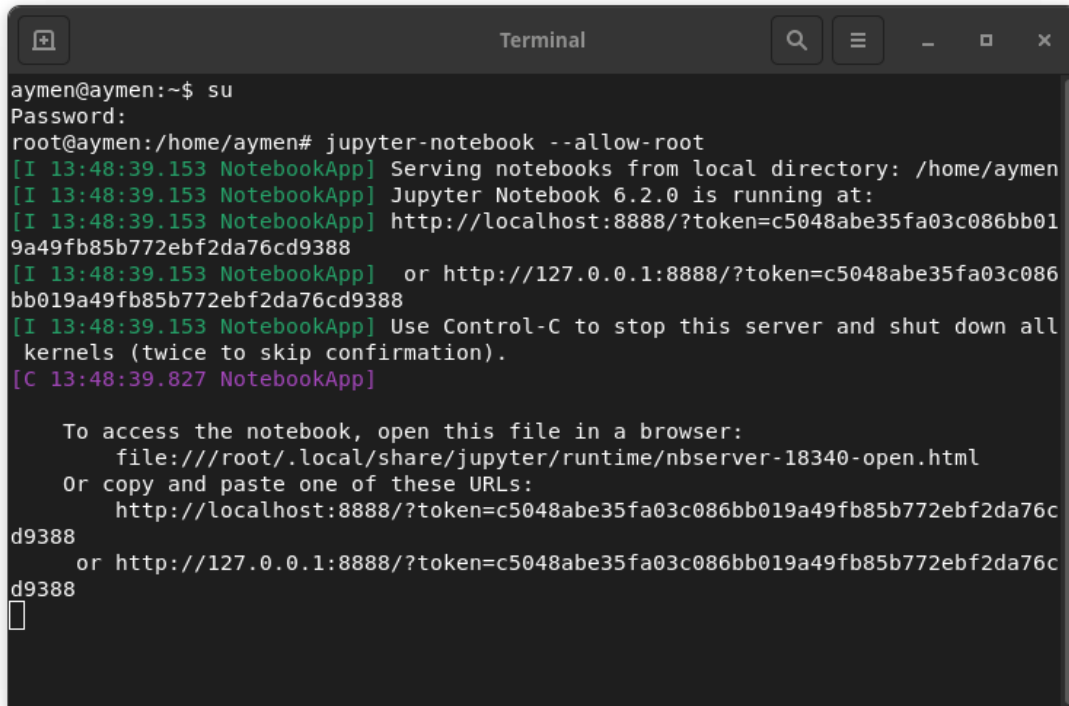


Figure IV.15: Terminal window, and type “su” command

Chapter IV : Application

After that you'll be in the "root" which you can add more command so we open a localhost:8888 and it gave us token to access the jupyter and also to keep us secure.



```
aymen@aymen:~$ su
Password:
root@aymen:/home/aymen# jupyter-notebook --allow-root
[I 13:48:39.153 NotebookApp] Serving notebooks from local directory: /home/aymen
[I 13:48:39.153 NotebookApp] Jupyter Notebook 6.2.0 is running at:
[I 13:48:39.153 NotebookApp] http://localhost:8888/?token=c5048abe35fa03c086bb019a49fb85b772ebf2da76cd9388
[I 13:48:39.153 NotebookApp] or http://127.0.0.1:8888/?token=c5048abe35fa03c086bb019a49fb85b772ebf2da76cd9388
[I 13:48:39.153 NotebookApp] Use Control-C to stop this server and shut down all kernels (twice to skip confirmation).
[C 13:48:39.827 NotebookApp]

To access the notebook, open this file in a browser:
file:///root/.local/share/jupyter/runtime/nbserver-18340-open.html
Or copy and paste one of these URLs:
http://localhost:8888/?token=c5048abe35fa03c086bb019a49fb85b772ebf2da76cd9388
or http://127.0.0.1:8888/?token=c5048abe35fa03c086bb019a49fb85b772ebf2da76cd9388
```

Figure IV.16: Do some command to open server for jupyter notebook

Jupyter notebook will be accessible on the browser I'm using here chromium



Figure IV.17: Jupyter notebook browser

Now we create a Jupyter file. I chose the file we created by the simulation to make access to data easier. Also I used python instead of Nodejs, because it's the language of AI. And it's easy to read.

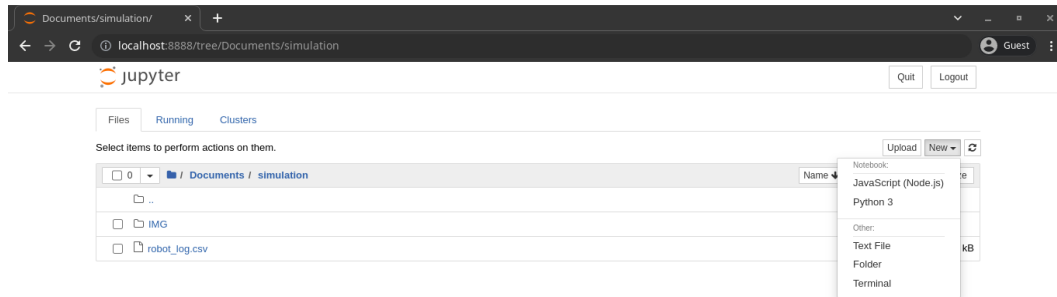


Figure IV.18: Create file in Jupyter notebook

Now we are ready to go.

- **Applying Artificial intelligence**

First thing is importing the necessary libraries.

```
import cv2 # for perspective transform
import glob # to reading in a list of images from a folder
import numpy as np # to calculate numerical operation
import matplotlib.image as mpimg # to plot images
import matplotlib.pyplot as plt #to plot
import pandas as pd # to read our data
```

First we check how many image we have

```
path = './IMG/*'
img_list = glob.glob(path)
print(len(img_list)) # Know how many image we have : result = 1546
```

We'll plot 4 image to get better vision about the project to we create a helpful plotting function.

```
def plot_images(img1, img2):  
    fig = plt.figure(figsize=(15, 10))  
    plt.subplot(121)  
    plt.imshow(img1)  
    plt.subplot(122)  
    plt.imshow(img2)
```

After that we import several image to study, I chose random image with just one condition that some of those image need to have stones and I plotted the all together.

```
img1 = mpimg.imread(img_list[len(img_list)-1])  
img2 = mpimg.imread(img_list[630])  
img3 = mpimg.imread(img_list[650])  
img4 = mpimg.imread(img_list[0])  
fig = plt.figure(figsize=(20, 10))  
plot_images(img1, img2)  
plot_images(img3, img4)
```

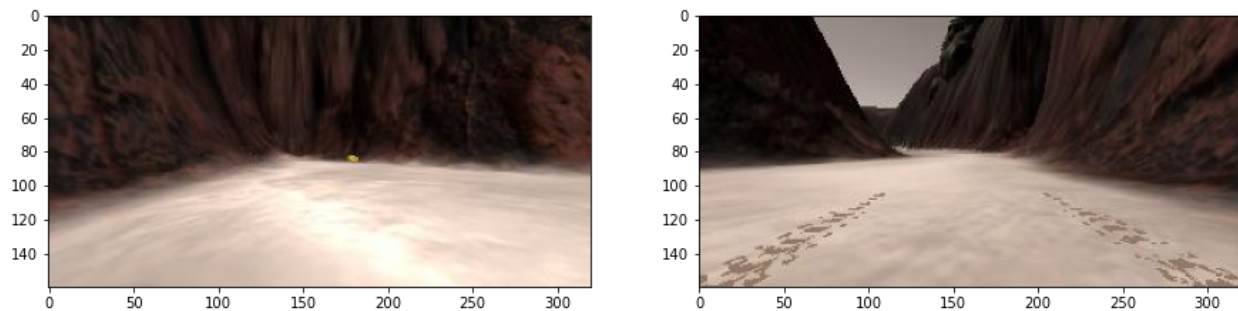


Figure IV.19: image 1 ,2 took by robot camera for train

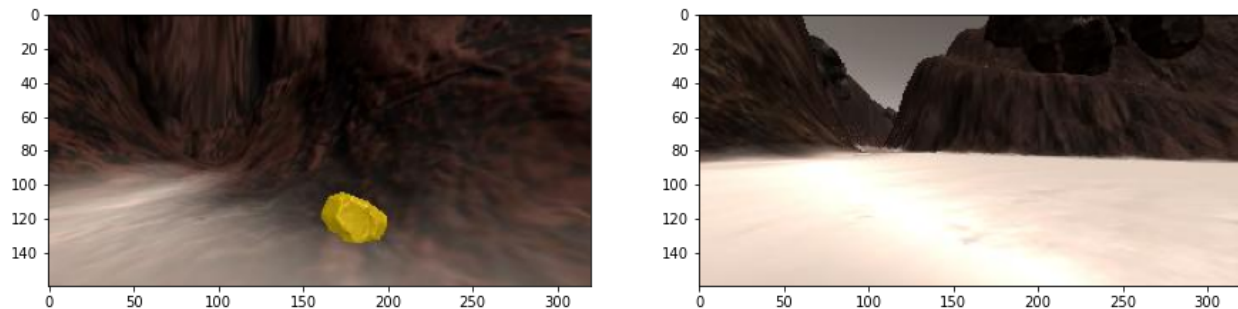


Figure IV.20: image 3,4 took by robot camera for train

- **Build a filter**

After we plot we need to make the robot knows what's in the picture We must find the max and min thresh of each image so we can split ground from mountain from rocks , so we give HLS color of each element The hue, lightness, and saturation (HLS) model is very identical to the HSV model, except that instead of an inverted hex-cone, a double hex-cone is utilized. In this paradigm, both systems' hue and saturation are defined in the same way. The lightness and value are slightly different[49].

```
ground_thresh_min = (0, 100, 70)
ground_thresh_max = (255, 255, 255)

rock_thresh_min = (0, 100, 0)
rock_thresh_max = (255, 255, 70)

mountain_thresh_min = (0, 0, 0)
mountain_thresh_max = (255, 100, 255)
```

After we identify the degree of each color now we'll build the filtre.

```
def filter_hls(img, thresh_min, thresh_max, height = None):
    hls_img = img.copy() # make a copy
    cv2.cvtColor(hls_img, cv2.COLOR_BGR2HLS) # convert image from rgb to hls
    # create a zero matrix same xy size exactly like the image
    binary_img = np.zeros_like(img[:, :, 0])

    within_thresh = (hls_img[:, :, 0] >= thresh_min[0]) & (hls_img[:, :, 0] <= thresh_max[0])
    & \
    (hls_img[:, :, 1] >= thresh_min[1]) & (hls_img[:, :, 1] <= thresh_max[1]) &
    (hls_img[:, :, 2] >= thresh_min[2]) & (hls_img[:, :, 2] <= thresh_max[2])

    binary_img[within_thresh] = 1
    if height is not None:
        binary_img[:height, :] = 0
    return binary_img
```

After that we create function to plot the result of the filter anso that function will filter the environment (rock, ground , mountain) from each other.

```
def show_filtered_result(img):  
  
    ground_bin = filter_hls(img, ground_thresh_min, ground_thresh_max, height = 70)  
  
    rock_bin = filter_hls(img, rock_thresh_min, rock_thresh_max)  
  
    mountain_bin = filter_hls(img, mountain_thresh_min, mountain_thresh_max)  
  
    fig = plt.figure(figsize = (20,3))  
  
    plt.subplot(141)  
  
    plt.imshow(img)  
  
    plt.subplot(142)  
  
    plt.imshow(ground_bin, cmap='gray')  
  
    plt.subplot(143)  
  
    plt.imshow(rock_bin, cmap='gray')  
  
    plt.subplot(144)  
  
    plt.imshow(mountain_bin, cmap='gray')
```

Now, we plot our images.

```
show_filtered_result(img1)  
  
show_filtered_result(img2)  
  
show_filtered_result(img3)  
  
show_filtered_result(img4)
```

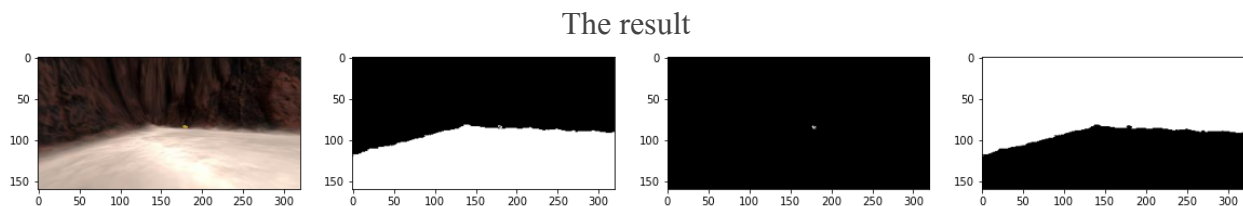


Figure IV.21: Result of image 1 in the filtre (normal,ground,rock and mountain)

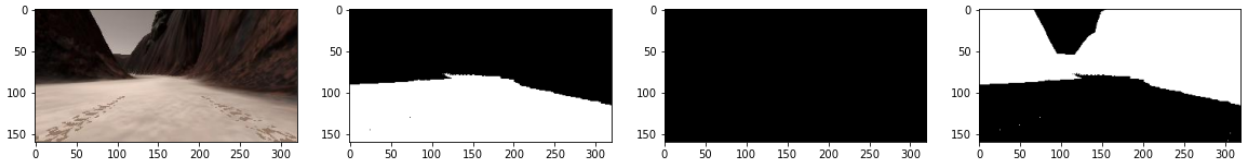


Figure IV.22: Result of image 2 in the filtre (normal,ground,rock and mountain)

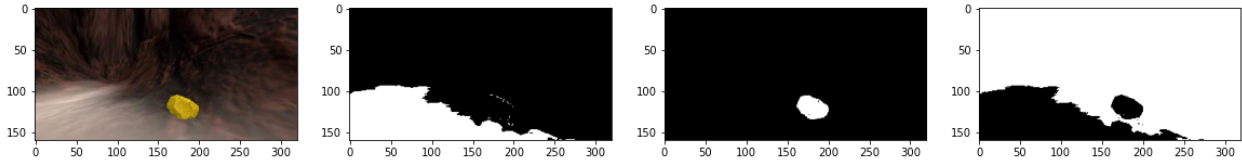


Figure IV.23: Result of image 3 in the filtre (normal,ground,rock and mountain)

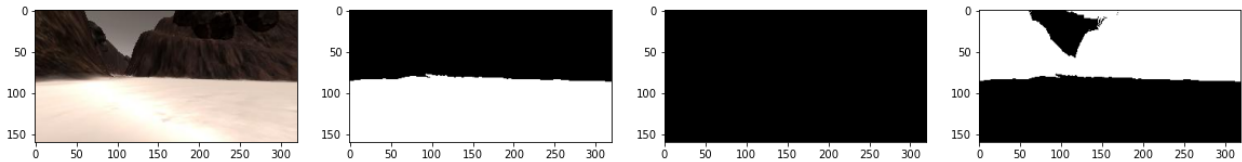


Figure IV.24: Result of image 4 in the filtre (normal,ground,rock and mountain)

- **Discuss the Results**

The plotted results shown a splitted images that we decided to study according to the degree of color that we identify for element we have in the environment In the left we can see the normal camera image(img1,img2,img3,img4) The second image present the image that the artificial intelligent identified as a ground so as you can see the white part is the ground and the black part is anything else The next image representing the rock that has been identified in the image

We can see in the image 1 too that there is a small white point that shows that there is a rock but in far distance, in image 4 we can see the white spot more large because the rock is actually as it's visible in the image so near from the camera

Image 2 and 4 shows that it's completely black that's because there are no rocks on those images.

- **Get one single image of the filter**

Now, we just have to create one single picture , so we'll just create a new function get the same pictures and then change their colors and plot them all together in on single image.

```
def coded_image(img):  
  
    ground_bin = filter_hls(img, ground_thresh_min, ground_thresh_max, height = 70)  
  
    rock_bin = filter_hls(img, rock_thresh_min, rock_thresh_max)  
  
    mountain_bin = filter_hls(img, mountain_thresh_min, mountain_thresh_max)  
  
    y_ground, x_ground = ground_bin.nonzero()  
    y_rock, x_rock = rock_bin.nonzero()  
    y_mountain, x_mountain = mountain_bin.nonzero()  
  
    coded_img = img * 0  
    coded_img[y_ground, x_ground, 0] = 255  
    coded_img[y_rock, x_rock, 1] = 255  
    coded_img[y_mountain, x_mountain, 2] = 255  
    return coded_img
```

Plotting the same and see the results.

```
plot_images(img1, coded_image(img1))  
  
plot_images(img2, coded_image(img2))  
  
plot_images(img3, coded_image(img3))  
  
plot_images(img4, coded_image(img4))
```

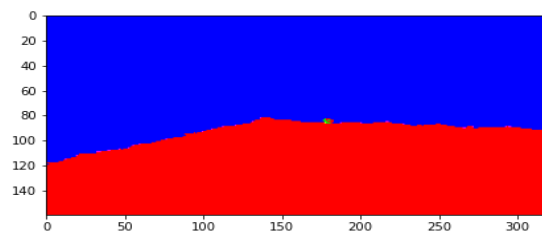
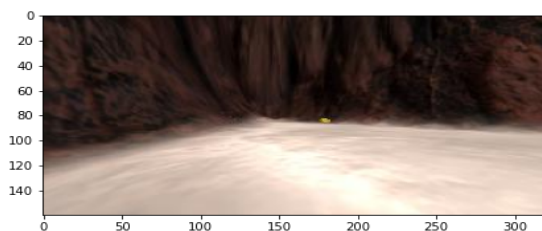


Figure IV.25: Result of image 1 after coded

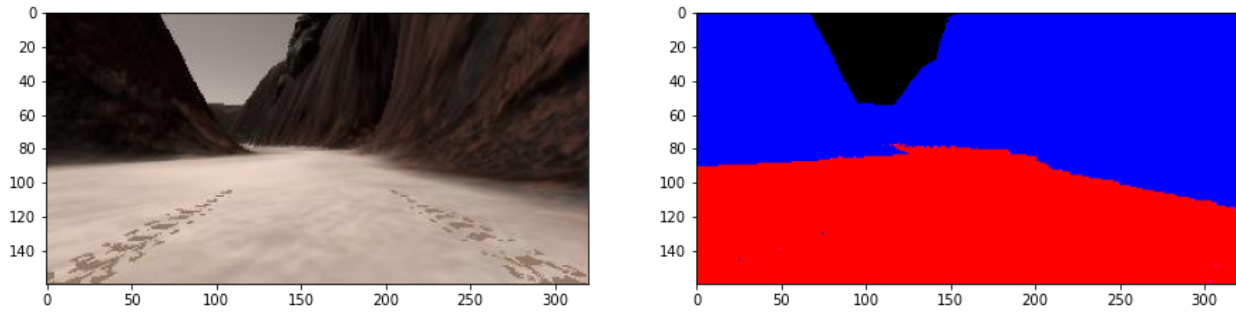


Figure IV.26: Result of image 2 after coded

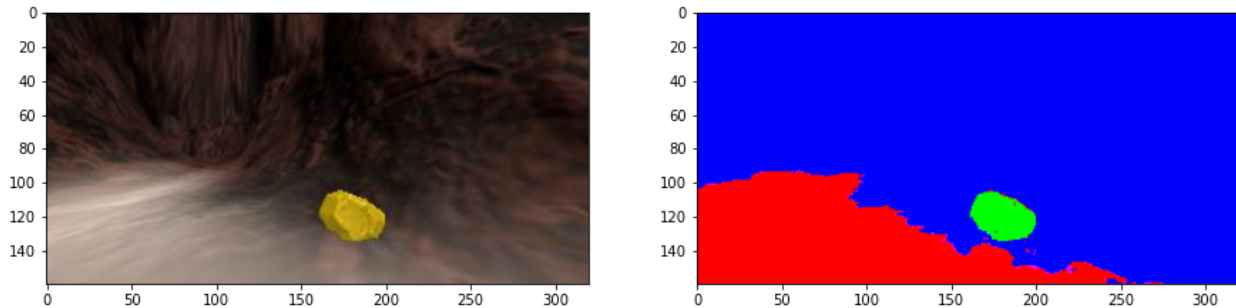


Figure IV.27: Result of image 3 after coded

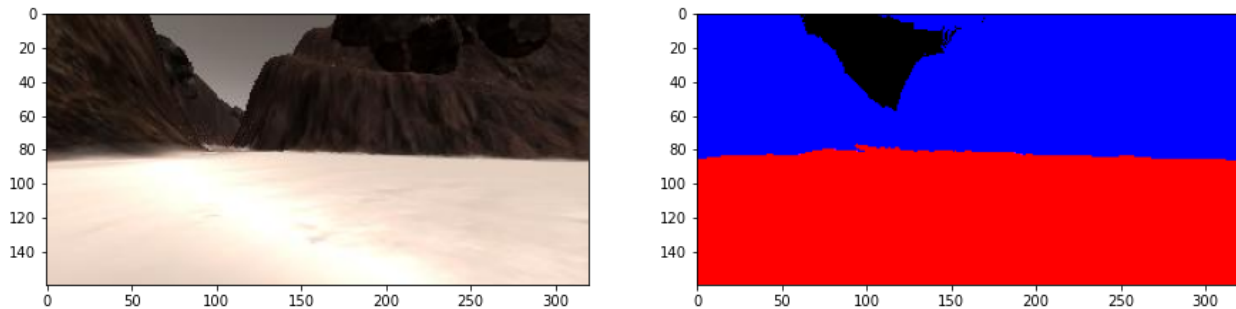


Figure IV.28: Result of image 3 after coded

- **Discuss the results**

After we plot all the results we can see that the blue color presents the mountain and red is the ground and green presenting the rocks but we can notice also black color founded in the images that would presenting anything except the identified field in this case we can see that the black color is presenting the sky.

- **Warning**

The black color can be anything not only the sky , if you put unknown shape that has a different color than the rock,mountain, and the ground the it will shows black

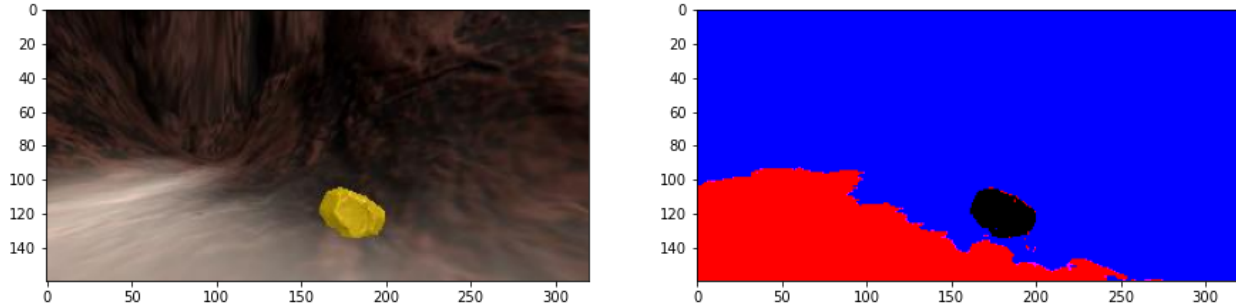


Figure IV.29: Result of image 3 after we remove the rock property

- **Make the robot see the distance**

To create a map base on the environment , we need to apply the perspective transform, we need to create a view from above

We started by returning to the simulation and run in training mode then we press ‘g’ that we’ll activate the grid on the map and we record after that just to create one image,

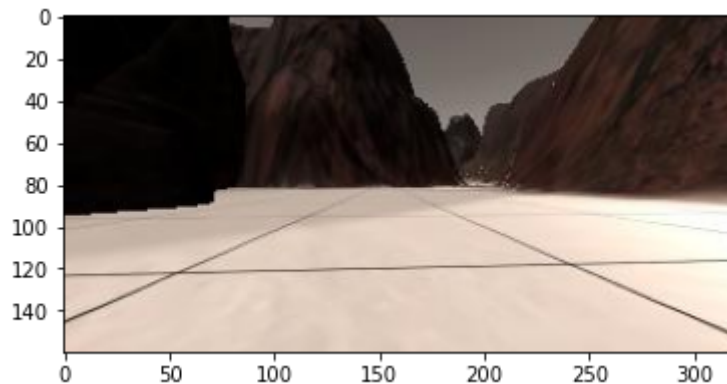


Figure IV.30: Image with grid

To preference the above view we need to use **cv2.getPerspectiveTransform()** and also **cv2.warpPerspective()** those are two function in opencv which is a very popular computer vision library in python

The simple idea in opencv is like that : A 3x3 transformation matrix is required for perspective transformation. Even after the change, straight lines will stay straight. You'll need 4 points on the

input image and equivalent points on the output image to find this transformation matrix. Three of these four points should not be collinear. The transformation matrix can then be retrieved using the `cv.getPerspectiveTransform` method. Then, given this 3x3 transformation matrix, use `cv.warpPerspective`. [50]

```
def perspect_transform(img, src, dst):  
  
    transform_matrix = cv2.getPerspectiveTransform(src, dst)  
    dimensions = (img.shape[1], img.shape[0])  
  
    warped_img = cv2.warpPerspective(img, transform_matrix, dimensions)  
  
    return warped_img
```

We need also to identify the destination and the source .

```
dest_size, bottom_offset = 5, 6  
source = np.float32([[14, 140], [301, 140], [200, 96], [118, 96]])  
  
h, w = grid_img.shape[0], grid_img.shape[1]  
  
p1, p2 = w / 2 - dest_size, w / 2 + dest_size  
  
p3, p4 = h - bottom_offset, h - 2 * dest_size - bottom_offset  
  
destination = np.float32([  
    [p1, p3], [p2, p3], [p2, p4], [p1, p4],  
    ])
```

Now we have just to run the code

```
warped = perspect_transform(grid_img, source, destination)  
  
plot_two_images(grid_img, warped)
```

Chapter IV : Application

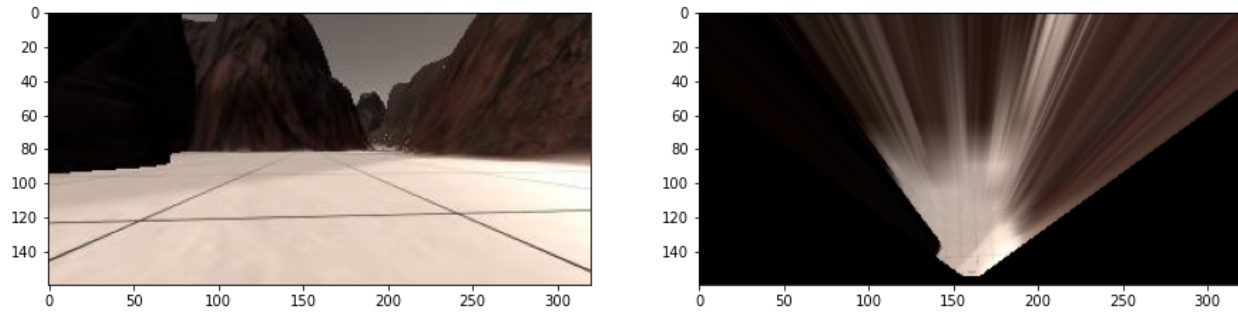


Figure IV.31: The grid picture in normal and after the perspective transform from z index view

Now, we just apply that for the other images so have the map view.

```
warped_1 = perspect_transform(img1, source, destination)
warped_2 = perspect_transform(img2, source, destination)
warped_3 = perspect_transform(img3 ,source, destination)

warped_4 = perspect_transform(img4, source, destination)
plot_two_images(img1,warped_1)
plot_two_images(img2,warped_2)

plot_two_images(img3,warped_3)
plot_two_images(img4,warped_4)
```

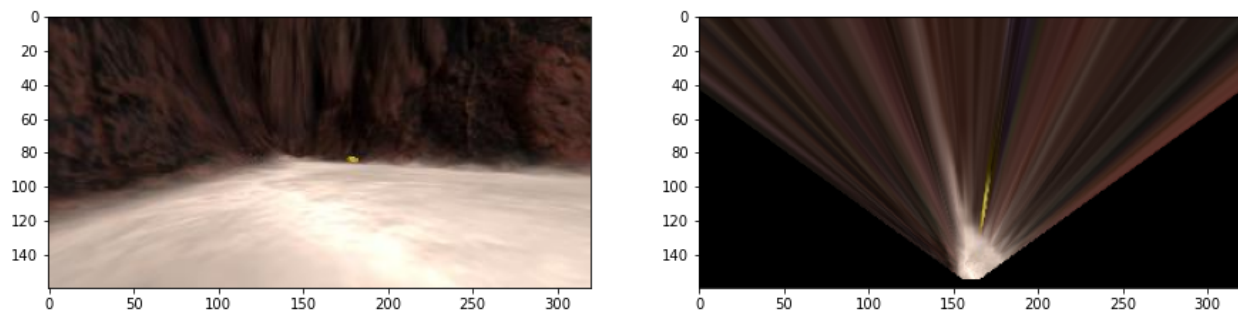


Figure IV.32: Result of image 1 normal and with perspective transform

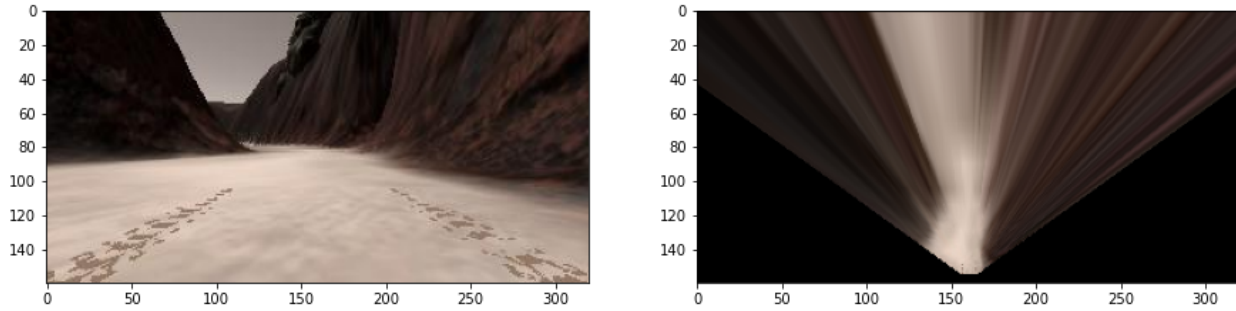


Figure IV.33: Result of image 2 normal and with perspective transform

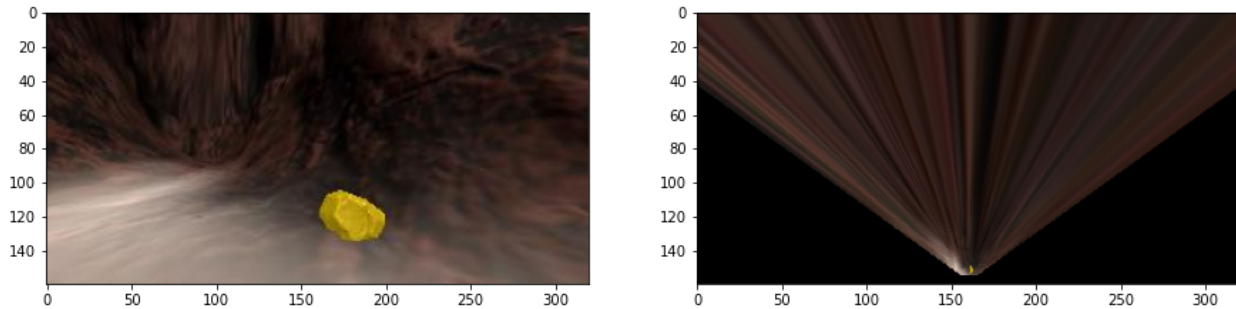


Figure IV.34: Result of image 3 normal and with perspective transform

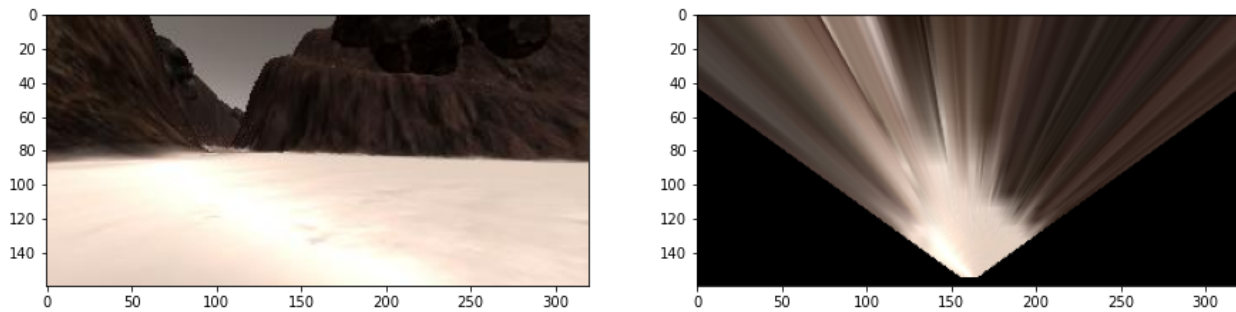


Figure IV.35: Result of image 4 normal and with perspective transform

- **Discuss the results**

The perspective transform function that allowing us to have an upper view of the environment and that what we can see in the testing images the result presenting and upper view of the image we can see that in all the pictures what we can found interesting is the rock now we can see that the yellow color longer and not in shape and thats returned to the nature of the perspective transform it doesn't recognize the shapes or the distance.

Chapter IV : Application

Perspective transform is often using in image stabelization in modern smart phone to keep the image fixed during filming.

- **Colorize the new view**

```
plot_two_images(warped, get_coded_image(warped))
plot_two_images(warped_1, get_coded_image(warped_1))

plot_two_images(warped_2, get_coded_image(warped_2))
plot_two_images(warped_3, get_coded_image(warped_3))

plot_two_images(warped_4, get_coded_image(warped_4))
```

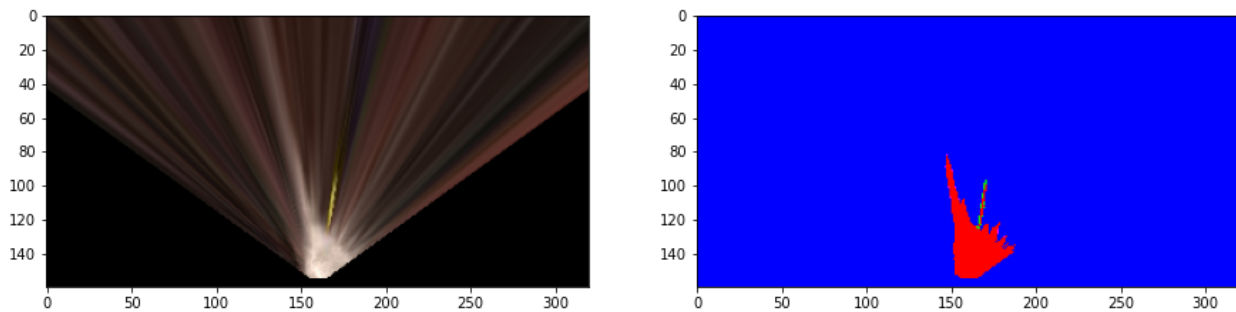


Figure IV.36: Perspective transform of image 1 with coded

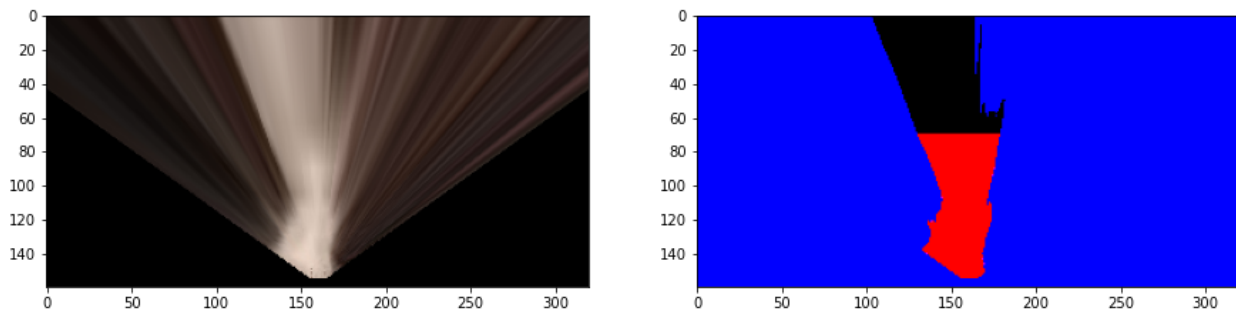


Figure IV.37: Perspective transform of image 2 with coded

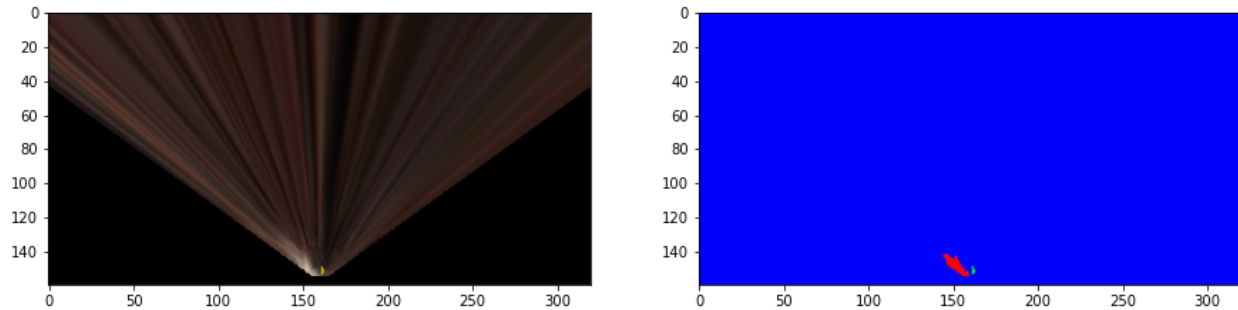


Figure IV.38: Perspective transform of image 3 with coded

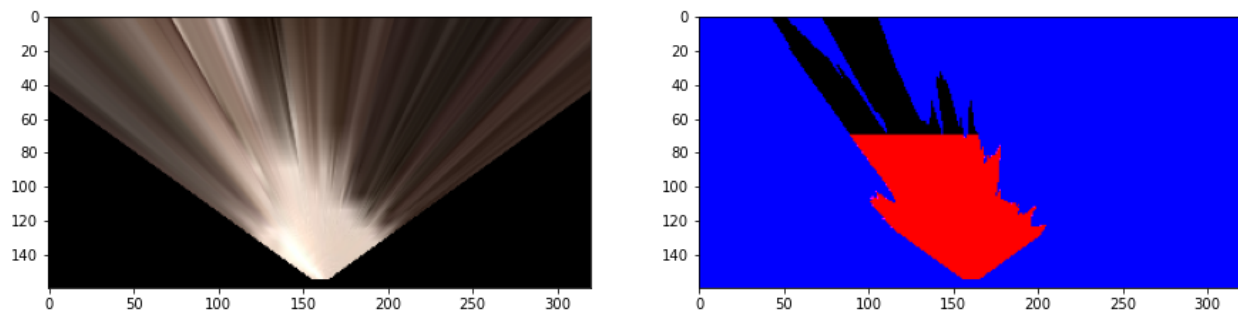


Figure IV.39: Perspective transform of image 4 with coded

- **Discuss the results**

We can see after we implement colorization on the prespective transform images more visibility of the ground on all the images and that make us have a better view of the directions and based on that we can take decisions.

- **Make a decision**

Now, after we prepare the data to learn , it's time to use the csv data. So we'll import it and to read later from it the location of the map.

```
df = pd.read_csv('./robot_log.csv', delimiter=';', decimal=','.)
data_xs = df["X_Position"].values

data_ys = df["Y_Position"].values
data_yaws = df["Yaw"].values
```

```
WORLD_SIZE, SCALE = 200, 30

# Those are just some example of Yaws, X and Y positions
print("yaws:")
print(data_yaws[1])
print(data_yaws[100])
print(data_yaws[1000])
print()

print("x positions:")
print(data_xs[1])
print(data_xs[100])
print(data_xs[1000])
print()

print("y positions:")
print(data_ys[1])
print(data_ys[100])
print(data_ys[1000])
print()
```

```
yaws:
51.74515
29.94607
259.1309
```

```
x positions:
101.2565
110.2521
102.9781
```

```
y positions:
87.57255
96.48149
189.0413
```

Now we will import the map , to have better vision of the location

```
ground_truth_bin = mpimg.imread('./data/map.png')
ground_truth_ys, ground_truth_xs = ground_truth_bin.nonzero()
ground_truth_img = np.zeros((200, 200, 3))
```

```
ground_truth_img[ground_truth_ys, ground_truth_xs, :] = (0, 1, 0)
fig = plt.figure(figsize=(16, 10))

plt.imshow(ground_truth_img)
```

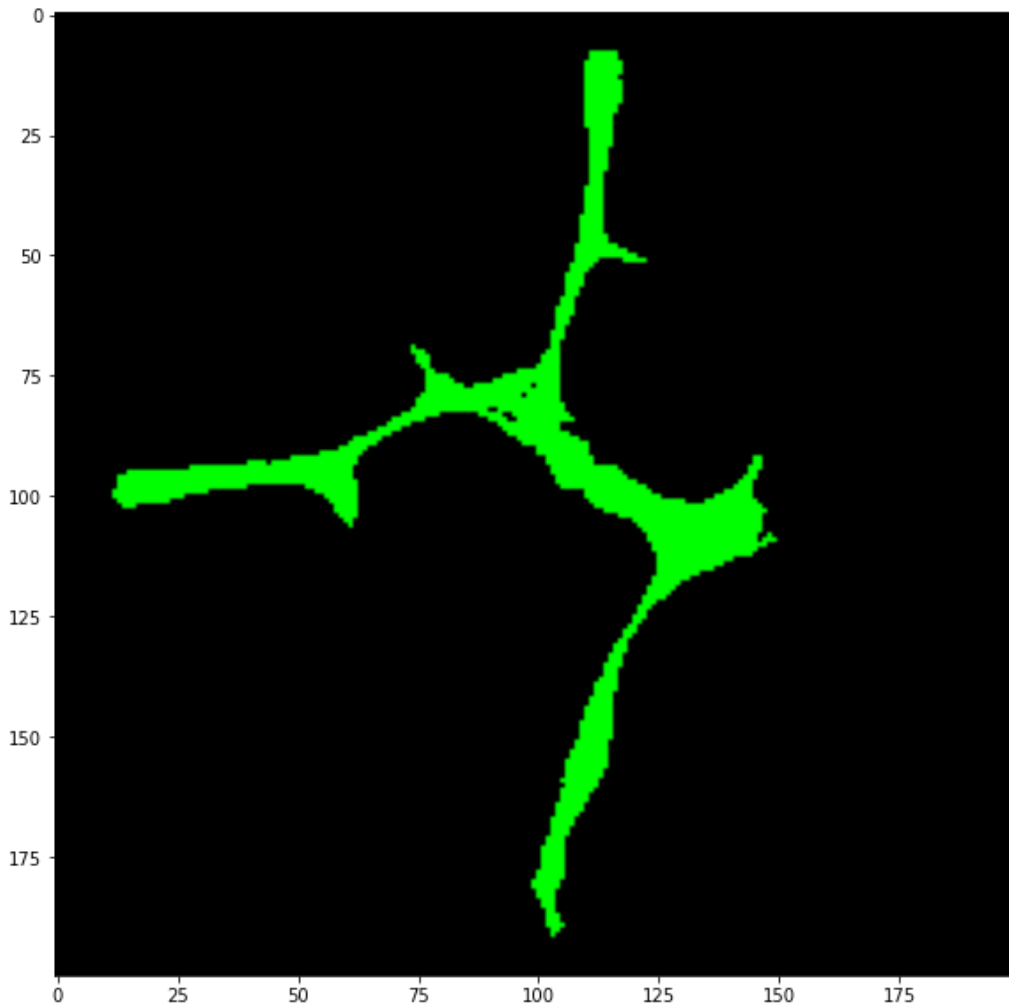


Figure IV.40: Map of the envirmnt [51].

After we import the map it's time now to get the rover coordinates

```
def rover_coordinates(binary_img):
    ys, xs = binary_img.nonzero()
```

```
xs_rover = -(ys - binary_img.shape[0])
ys_rover = -(xs - binary_img.shape[1] / 2 )
```

```
return xs_rover, ys_rover
```

```
def convert_to_polar(xs, ys):
```

```
    distances = np.sqrt(xs**2 + ys**2)
    angles = np.arctan2(ys, xs)
    return distances, angles
```

Now, we treat all the what we have in one function.

```
def perception_pipeline(img):
```

```
    ground_bin = filter_hls(img, ground_thresh_min, ground_thresh_max, height = 70)
    warped_bin = perspect_transform(ground_bin, source, destination)
    warped_coded_img = get_coded_image(perspect_transform(img, source, destination))
    xs, ys = get_rover_coordinates(warped_bin)
    distances, angles = convert_to_polar(xs, ys)
    mean_direction = np.mean(angles)
    fig = plt.figure(figsize = (20,4))
    plt.subplot(141)
    plt.imshow(img)
    plt.subplot(142)
    plt.imshow(ground_bin, cmap = 'gray')
    plt.subplot(143)
    plt.imshow(warped_coded_img)
    plt.subplot(144)
    plt.plot(xs, ys, '!')
```



```
plt.ylim(-160, 160)
plt.xlim(0, 160)

arrow_length = 100
x_arrow = arrow_length * np.cos(mean_direction)

y_arrow = arrow_length * np.sin(mean_direction)

plt.arrow(0, 0, x_arrow, y_arrow, color = 'red', zorder = 10, head_width = 10, width = 2)
```

```
perception_pipeline(img1)
perception_pipeline(img2)
perception_pipeline(img3)
perception_pipeline(img4)
```

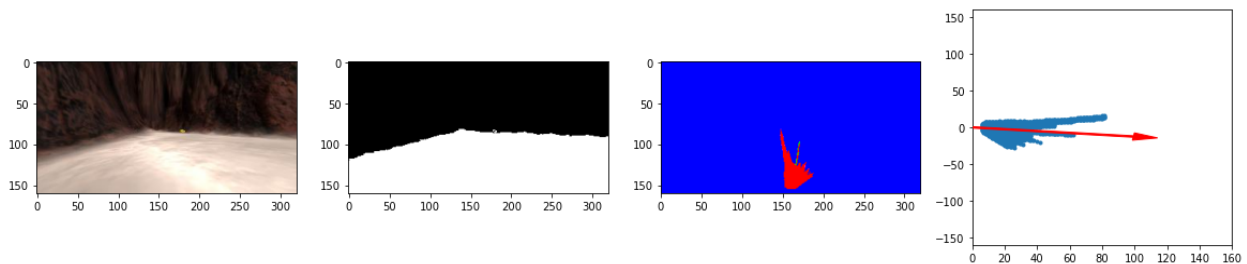


Figure IV.41: Normal,ground filtre, coded color of perspective transform and decision making(image 1)

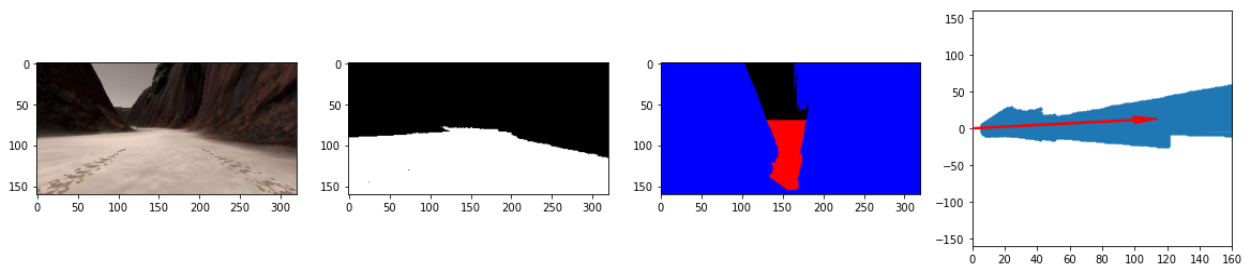


Figure IV.42: Normal,ground filtre, coded color of perspective transform and decision making(image 2)

Chapter IV : Application

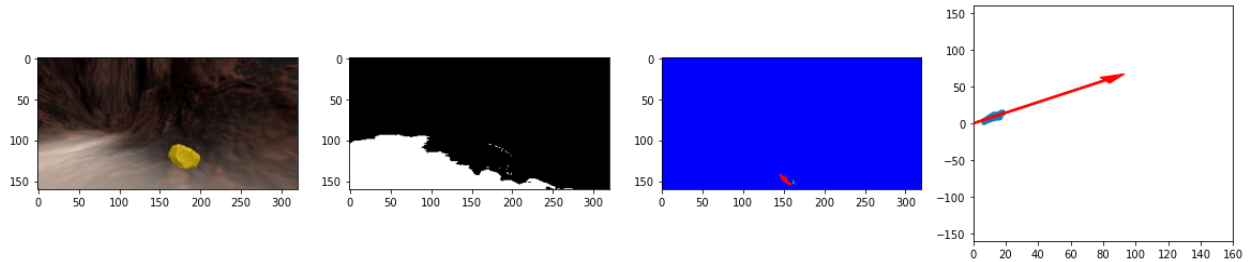


Figure IV.43: Normal,ground filtre, coded color of perspective transform and decision making(image 3)

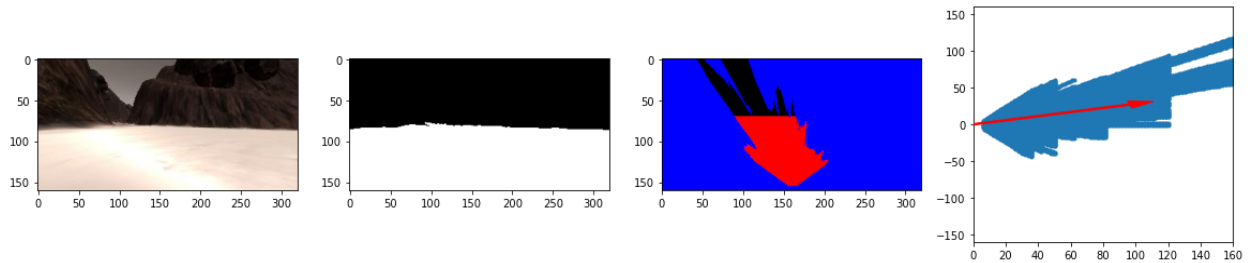


Figure IV.44: Normal,ground filtre, coded color of perspective transform and decision making(image 4)

- **Discuss the results**

After making the decision making we can see that the robot takes the straight forward way to and that what we can see in the last column for each image.

- **Recognize the location**

```
def convert_rover_to_world_coordinates(xs, ys, x_pos, y_pos, yaw, world_size, scale):
```

```
    a = yaw * np.pi / 180
```

```
    xs_rotated = (xs * np.cos(a)) - (ys * np.sin(a))
```

```
    ys_rotated = (xs * np.sin(a)) + (ys * np.cos(a))
```

```
    xs_world = xs_rotated / scale + x_pos
```

```
    ys_world = ys_rotated / scale + y_pos
```

```
xs_world = np.clip(np.int_(xs_world), 0, world_size - 1)
ys_world = np.clip(np.int_(ys_world), 0, world_size - 1)

return xs_world, ys_world
```

```
def mapping_pipeline(img, x_pos, y_pos, yaw):

    world_size, scale = WORLD_SIZE, 10

    ground_bin = filter_hls(img, ground_thresh_min, ground_thresh_max, height = 70)
    rock_bin = filter_hls(img, rock_thresh_min, rock_thresh_max)
    blocked_bin = filter_hls(img, blocked_thresh_min, blocked_thresh_max)

    warped_ground_bin = perspect_transform(ground_bin, source, destination)
    warped_rock_bin = perspect_transform(rock_bin, source, destination)
    warped_blocked_bin = perspect_transform(blocked_bin, source, destination)

    ground_rover_xs, ground_rover_ys = get_rover_coordinates(warped_ground_bin)
    rock_rover_xs, rock_rover_ys = get_rover_coordinates(warped_rock_bin)
    blocked_rover_xs, blocked_rover_ys = get_rover_coordinates(warped_blocked_bin)

    gxs, gys = convert_rover_to_world_coordinates(ground_rover_xs, ground_rover_ys, x_pos,
    y_pos, yaw, world_size, scale)

    rxs, rys = convert_rover_to_world_coordinates(rock_rover_xs, rock_rover_ys, x_pos, y_pos,
    yaw, world_size, scale)

    bxs, bys = convert_rover_to_world_coordinates(blocked_rover_xs, blocked_rover_ys, x_pos,
    y_pos, yaw, world_size, scale)
```

```
world_ys, world_xs = ground_truth_bin.nonzero()
map_img = np.zeros((200, 200, 3))
map_img[world_ys, world_xs, :] = (1, 1, 1)

with_rover_img = np.zeros((200, 200, 3))
with_rover_img[gys, gxs, 0] = 1
with_rover_img[rys, rxs, 1] = 1

with_rover_img[bys, bxs, 2] = 1

map_with_rover_img = cv2.addWeighted(map_img, 0.3, with_rover_img, 1, 0)

coded_img = get_coded_image(img)
warped_coded_img = perspect_transform(coded_img, source, destination)

fig = plt.figure(figsize = (20,4))

plt.subplot(141)
plt.imshow(coded_img)

plt.subplot(142)
plt.imshow(warped_coded_img)

plt.subplot(143)
plt.plot(blocked_rover_xs, blocked_rover_ys, '!')
plt.plot(ground_rover_xs, ground_rover_ys, '!')

plt.ylim(-160, 160)
plt.xlim(0, 160)

plt.subplot(144)
plt.imshow(map_with_rover_img)
```

Chapter IV : Application

```
mapping_pipeline(img1, data_xs[1545], data_ys[1545], data_yaws[1545])
mapping_pipeline(img2, data_xs[630], data_ys[630], data_yaws[630])
mapping_pipeline(img3, data_xs[650], data_ys[650], data_yaws[650])
mapping_pipeline(img4, data_xs[0], data_ys[0], data_yaws[0])
```

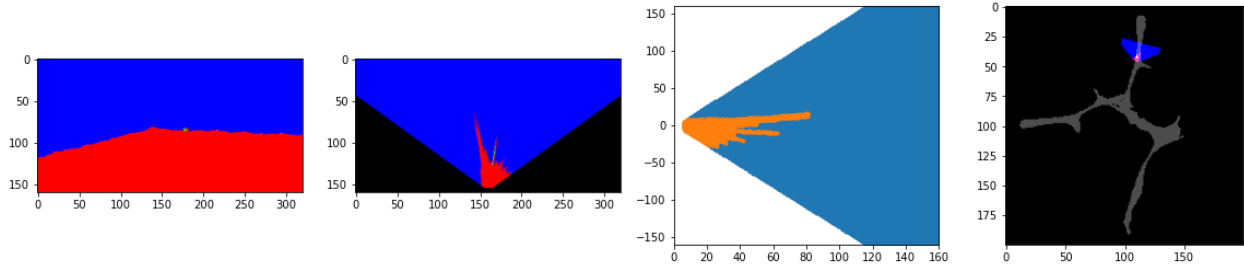


Figure IV.45: Coded ground filtre, coded color of perspective transform, blocked coded color of perspective transform and position in the map (image 1)

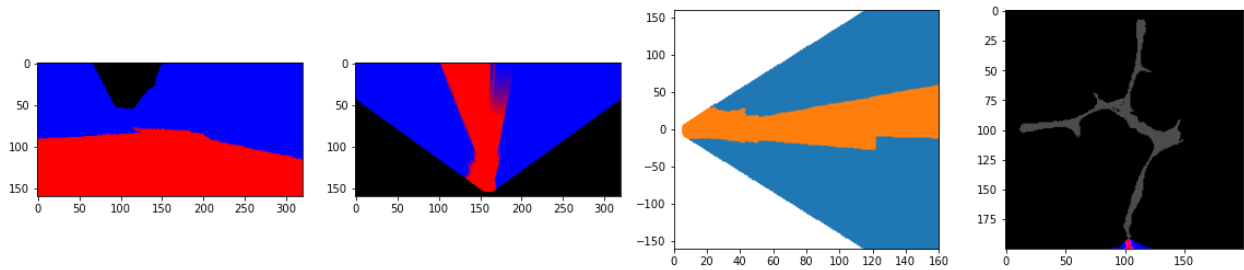


Figure IV.46: Coded ground filtre, coded color of perspective transform, blocked coded color of perspective transform and position in the map (image 2)

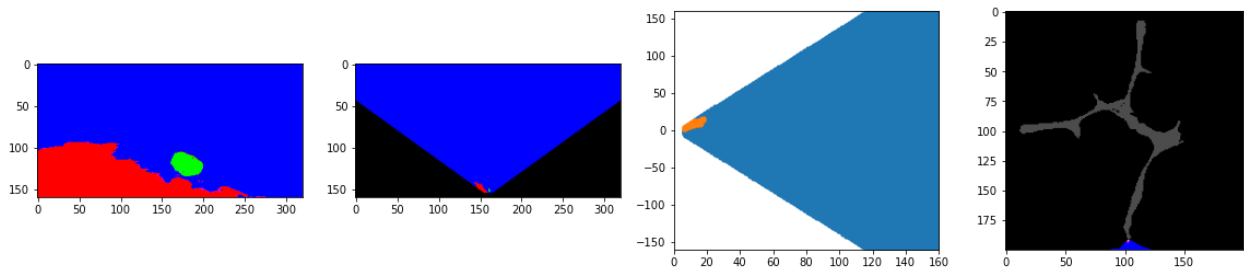


Figure IV.47: Coded ground filtre, coded color of perspective transform, blocked coded color of perspective transform and position in the map (image 3)

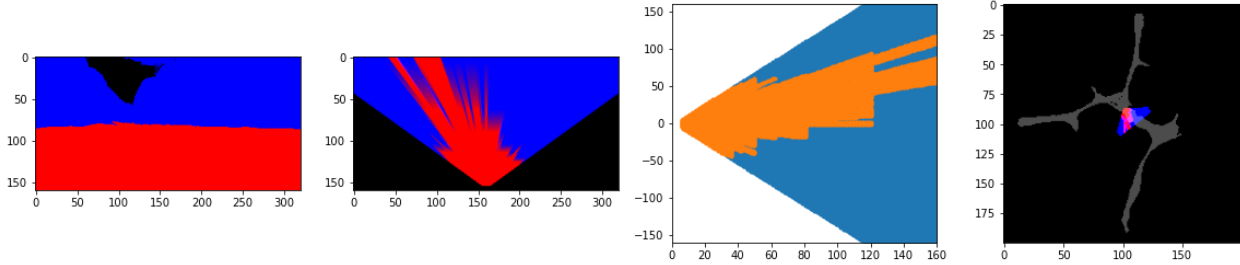


Figure IV.48: Coded ground filtre, coded color of perspective transform, blocked coded color of perspective transform and position in the map (image 4)

- **Discuss the results**

The Result of our four image shown in the first column the coded image, second column the perspective transform , the third one the blocked position and the last column shown the position of the robot in the map in the last column we can see the position of the robots in the map it's not really that exact according ro the ability of the simple camera that can detects.

- **Suggestion for anybody who wants to continue in that project**

Due I don't have enough time to continue in the simulation there is several improvement that can be really good if you want to take a step and working on that project.

- **Recreate the environment :** First of all that doesn't related with my tinny time but for my experience with 3d I don't know too much experience in 3D special Unity and C#, but if you have an idea you can make a completely new environment with multiple rock colors and then give orders to the robot to bring you certain rocks in certain places. If you don't have any idea from where you should start then try to check "gazebo, ROS,ROS2,blender,unreal engine,FreeCAD" .
- **Use a PID controller** Using PID controller is really good idea to avoid oscillating behavior and to make the project looks more realistic and smother
- **Use Reinforcement learning** In the time that I'm writing this is not really popular with the RL but it would be a great choice to make that robot learn from the environment, and send you data when it discovers something new. If you want to use RL check first if your laptop bear it.
- **Create a web app for the robot** Making a robot connected with the net is really advance topic but it need to know more about web development, at simulation used flask to open server so I assumed that the best choice for that is using that framework "Flask is python framework to create server side app" an alternative for Flask is django, the robot

is saving the data in csv from so best type of database is SQL”Structured Query Language”, also you may create restful api instead of graphql.

I believe that everything about the simulation we’ll go now to make the robot alive.

IV.3. Expiiermental part

- **Semi-controller Robot**

- **Case :** This robot is design to enter to place with tinny places to detect the nature of the product in that tinny place, it can be a unreachable places in mines or caves, the robot will be communicate with the pilot (human) with VLC and the robot will send images to the pilot with full of stats

To start working in that project we have already a wheeled robot from lasts year that built by MSc. **SELLAOUI Raid Housseem**

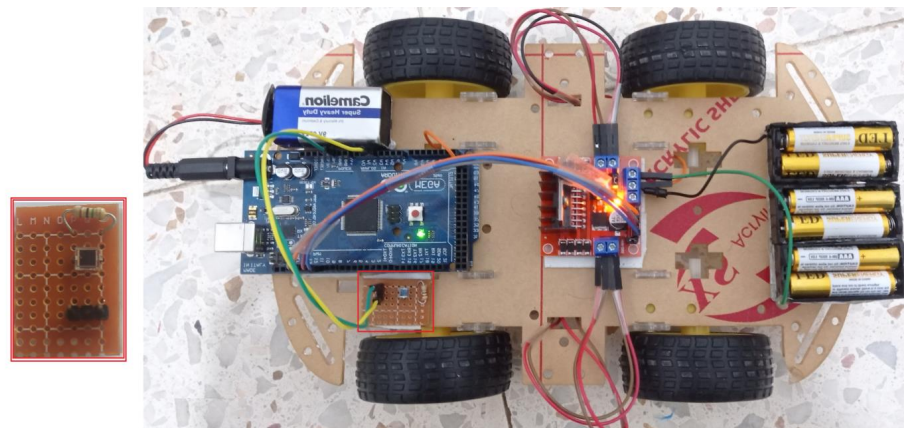


Figure IV.49: VLC-Car[6].

The Robot is built with arduino table that comunicate with VLC, we can see that half of the project was already done from previous colleague, Our goal now is build and AI and make it interect with this robot using raspberry pi 4

- **Raspberry pi**

The Raspberry Pi Foundation, in collaboration with Broadcom, created a series of miniature single-board computers in the United Kingdom. The Raspberry Pi project was created with the goal of encouraging the teaching of fundamental computer science in schools and disadvantaged nations.

Chapter IV : Application

- **Be aware** Before you start working on that project make sure that your python version is upper than version 2 (version 3) some libraries won't work in version 2 for example "numpy" which is very important library in artificial intelligence and it will be more important in this project .
- **Deadline**
Unfortunately, we had only a little time to work on this interesting project but the deadline has arrived and I'm in this part, I'll give several steps for anyone who wants to continue this project from here,
 - Check the last version of Raspberry OS.
 - Check if your Raspberry pi card bears all .

Conclusion

In this simulation we have created a smart software using python programming languages and the open cv library to treat the image, also we have been using jupyter notebook to get our code cleaner and to have better vision of our result.

Using easier operating system can be helpful for beginners, we took several image to make our study on it we can see the results are crystal clear on the outputs,

We started with making the filter that'll split up the different degree of our environments' items that will simplify our work, using HLS format better than the RGB because that format helps to change the degree of the brightness and not the colors themselves

Perspective transform is a transform often using in the smart phone to stabilize the images but it's also have other abilities, we used it to make a 3d view from and finally we using our data to make decision about our location, we use the map to see the current location.

In the experimental part we wanted to create a semi-controller robot using raspberry pi (3 or 4) and arduino, The robot will be wheeled, we send the orders to the robot using LED.

General conclusion

In this project we created a smart software that allows robots to train images to know the items in the current environment and make transforms to get better vision of the location as well as taking decision to know which direction it can drive, that's all according to a data that has been taken by a camera on the robot. We wanted to make this idea real by implemented in the real world, but the time was not beside us

Artificial intelligence is very interesting topic when it returns to robotics, and building a smart robot that would send response and receive request with VLC technology seems that needs a lot work and the short amount of time make it even more harder, so I hope for anyone who wants to continue in that project to verify if he/she has enough time to make a new improvement on that project

References

- [1] Robotics Modeling, Planning and Control by Bruno Siciliano, Lorenzo Sciavicco, Luigi Villani, Giuseppe Oriolo, 2009
- [2] Robotics, Etymology, 2022, https://en.wikipedia.org/wiki/Robotics#cite_note-1
- [3] Wikipedia, Robotics, 2022, https://en.wikipedia.org/wiki/Robotics#/media/File:Shadow_Hand_Bulb_large.jpg
- [4] Learn Robotics Programming Build and control autonomous robots using Raspberry Pi 3 and Python by Danny Staple, 2018
- [5] Type of Robots, Manning free content center, <https://freecontent.manning.com/types-of-robots/>
- [6] SELLAOUI Raid Housseem Eddine Control of a robot based on VLC technology, University of Guelma 8th May 1945
- [7] 6 among the most amazing robots in 2020, broken house company, 7 october 2020, <https://www.brokenhousecompany.it/blog/en/blog/2020/10/07/6-among-the-most-amazing-robots-in-2020/>
- [8] Top 5 countries using industrial robots in 2018: IFR, September 19, 2019, the robot report, <https://www.therobotreport.com/top-5-countries-using-industrial-robots-2018/>
- [9] Wheeled Robot Chassis : Buying & DIY Guide, Abhishek Ghosh, August 27, 2015, The Customize Windows, <https://thecustomizewindows.com/2015/08/wheeled-robot-chassis-buying-diy-guide/>
- [10] ANYmal X: Ex-Proof Inspection Robot - ANYbotics, <https://www.anybotics.com/anymal-ex-proof-inspection-robot/>
- [11] Medical robotics in China: the rise of technology in three charts, 24 June 2020, Sarah O'Meara, <https://www.nature.com/articles/d41586-020-01795-7>
- [12] CENTAURO – Robust Mobility and Dexterous Manipulation in Disaster Response by Full Body Telepresence in a Centaur-like Robot <https://www.centauro-project.eu/>
- [13] Consumer drones in conflict: where do they fit into IHL? - Humanitarian Law & Policy Blog, Faine Greenwood, 2022, <https://blogs.icrc.org/law-and-policy/2022/03/15/consumer-drones-conflict-ihl/>
- [14] U.S. Air Force photo/Lt Col Leslie Pratt, https://en.wikipedia.org/wiki/Military_robot#/media/File:Twuav_13_02.jpeg
- [15] Japan to create legal framework for level 4 self-driving cars, KrASIA, Nikkei Asia, 23 Dec 2021, <https://kr-asia.com/japan-to-create-legal-framework-for-level-4-self-driving-cars>
- [16] Hands-On Machine Learning with Scikit-Learn, Keras, and Tensorflow Concepts, Tools, and Techniques to Build Intelligent Systems by Aurélien Géron
- [17] What Is Machine Learning? <https://www.expert.ai/blog/machine-learning-definition/>
- [18] Simplifying the Difference: Machine Learning vs Deep Learning - Singapore Computer Society, Singapore Computer Society, <https://www.sap.com/insights/what-is-machine-learning.html>
- [19] Machine Learning: 6 Real-World Examples <https://www.salesforce.com/eu/blog/2020/06/real-world-examples-of-machine-learning.html>
- [20] Image detection, recognition and image classification with machine learning. , by

- Renukasoni , AITS Journal , Medium , <https://medium.com/ai-techsystems/image-detection-recognition-and-image-classification-with-machine-learning-92226ea5f595>
- [21] Neural networks and speech recognition - Machine Learning,25 May 2020,Diego Gosmar,<https://www.gosmar.eu/machinelearning/2020/05/25/neural-networks-and-speech-recognition/>
- [22] Medical Diagnosis Machine Learning Discount, 50% OFF , www.playadivingcenter.com, <https://campar.in.tum.de/twiki/pub/Chair/ProjectImageSynth/Project01.png>
- [23] Statistical Arbitrage: Defined & Strategies - Analyzing Alpha <https://analyzingalpha.com/wp-content/uploads/2022/04/statistical-arbitrage.jpg>
- [24] Supervised learning , Diego Calvo,Mar 23, 2019, <https://www.diegocalvo.es/en/supervised-learning/>
- [25] Unsupervised learning , Diego Calvo,Mar 24, 2019,<https://www.diegocalvo.es/en/learning-non-supervised/>
- [26] Machine Learning: What it is and Why it Matters, Decypher , May 16, 2018,<https://www.decypher.com/machine-learning-matters/>
- [27] The very basics of Reinforcement Learning ,by Aneek Das , Becoming Human: Artificial Intelligence Magazine,Mar 26, 2017,<https://becominghuman.ai/the-very-basics-of-reinforcement-learning-154f28a79071>
- [28] L. E. M. Matheus, A. B. Vieira, L. F. M. Vieira, M. A. M. Vieira and O. Gnawali, "Visible Light Communication: Concepts, Applications and Challenges," in IEEE Communications Surveys & Tutorials, vol. 21, no. 4, pp. 3204-3237, Fourth Quarter 2019, doi:10.1109/COMST.2019.2913348.
- [29] VLC Consortium,http://www.vlcc.net/modules/pico2/?content_id=28
- [30] Philip Ronan, Wikipedia, https://en.wikipedia.org/wiki/Visible_light_communication#/media/File:EM_spectrum.svg
- [31] N. Kumar, L. A. Nero and R. L. Aguiar, —Visible Light Communication for Advanced Driver Assistance Systems, the work is part of FCT project VIDAS–PDTC/EEA-TEL/75217, 2006.
- [32] Latif Ullah Khan, —Visible light communication: Applications, architecture, standardization and research challenges” Digital Communications and Networks, Volume 3, Issue 2, 2017, Pages 78-88, ISSN 2352-8648.
- [33] Parts of a conventional LED. The flat bottom surfaces of the anvil and post embedded inside the epoxy act as anchors, to prevent the conductors from being forcefully pulled out via mechanical strain or vibration.[https://en.wikipedia.org/wiki/Light-emitting_diode#/media/File:LED,_5mm,_green_\(en\).svg](https://en.wikipedia.org/wiki/Light-emitting_diode#/media/File:LED,_5mm,_green_(en).svg)
- [34] Chaabna A., Babouri A., Zhang X. —An Indoor Positioning System Based on Visible Light Communication Using a Solar Cell as Receiver.‖ In: Hatti M. (eds) Artificial Intelligence in Renewable Energetic Systems. ICAIRES 2017. Lecture Notes in Networks and Systems, vol 35. Springer, Cham. PP 43- 49. 2018.
- [35] Durgesh Gujjari —VISIBLE LIGHT COMMUNICATION‖ Submitted in partial fulfillment of the requirements for the degree of Master of Applied Science Dalhousie University Halifax, Nova Scotia August 2012.
- [36] versch Si-Fotodioden, oben eine Ge-Fotodiode

<https://en.wikipedia.org/wiki/Photodiode#/media/File:Fotodio.jpg>

[37] N. Barbot, S. Sahuguede, A. Julien-Vergonjanne, and J.-P. Cances, —LDPC and Fountain Code Performances over Mobile Wireless Optical Channel, *Transactions on Emerging Telecommunications Technologies*, 30/08/2013.

[38] Qing Wang, Domenico Giustiniano, and Omprakash Gnawali.. Low-Cost, Flexible and Open Platform for Visible Light Communication Networks. In Proceedings of the 2nd International Workshop on Hot Topics in Wireless (HotWireless '15). Association for Computing Machinery, New York, NY, USA, 31–35. 2015.

[39] Yusuf Perwej, *Journal of Computer Networks*, 2017, Vol. 4, No. 1, 20-29 Available online at <http://pubs.sciepub.com/jcn/4/1/3> Science and Education Publi,

[40] T. Yamazato et al., "Image-sensor-based visible light communication for automotive applications," in *IEEE Communications Magazine*, vol. 52, no. 7, pp. 88-97, July 2014, doi: 10.1109/MCOM.2014.6852088.

[41] "Stanford Takes Online Schooling To The Next Academic Level". *All Things Considered, National Public Radio*. 23 January

2012, <https://www.npr.org/blogs/alltechconsidered/2012/01/23/145645472/stanford-takes-online-schooling-to-the-next-academic-level>

[42] Cava, Marco della. "Online pioneer Udacity lands \$105 million round and a \$1 billion valuation". *USA TODAY*. Retrieved 2021-07-08.

<https://www.usatoday.com/story/tech/2015/11/11/online-pioneer-udacity-lands-105-million-round-and-1-billion-valuation/75544526/>

[43] Anderson, Stuart. "Sebastian Thrun: Udacity Would Not Exist Without Immigrants". *Forbes*. Retrieved 2021-07-

08. <https://www.forbes.com/sites/stuartanderson/2019/03/14/sebastian-thrun-udacity-would-not-exist-without-immigrants/>

[44] Logo of Udacity in SVG format, extracted from its website in 2018.

<https://en.wikipedia.org/wiki/Udacity>

[45] Github project <https://github.com/udacity/RoboND-Rover-Project>

[46] Official Project Jupyter webpage, Jupyter Notebook: The Classic Notebook Interface, 2022, <https://jupyter.org/>

[47] Official Project Jupyter webpage, main logo, 2022, <https://jupyter.org/>

[48] Debian website , 2022, <https://manpages.debian.org/testing/login/su.1.en.html>

[49] HSV HexCone,

https://support.ptc.com/help/mathcad/en/index.html#page/PTC_Mathcad_Help%2Fhls_and_hsv_color_representations.html%23

[50] Perspective Transformation, Geometric Transformations of Images,

https://docs.opencv.org/3.4/da/d6e/tutorial_py_geometric_transformations.html

[51] Simulation map https://github.com/udacity/RoboND-Rover-Project/blob/master/calibration_images/map_bw.png