University Mohamed Khider of Biskra
Faculty of Science and Technology
Department of Electrical Engineering

MASTER THESIS

Science and technology
Electronic
Embedded Systems Electronics

Réf. : ------------------------------------

Presented and supported by:

ATHMANI WAFA

Submitted on: 27/06/2022

Study and design of an assistance system for People with special needs with NAO ROBOT

Jury :

Pr.	Debilou Abderrazak	Prof	University of Biskra	President
Mr.	Rahmani Nacer-Eddine	MAA	University of Biskra	Supervisor
Dr.	Benelmir Okba	MCB	University of Biskra	Examiner

Academic year: 2021 - 2022

University Mohamed Khider of Biskra
Faculty of Science and Technology
Department of Electrical Engineering

MASTER THESIS

Science and technology
Electronic
Embedded Systems Electronics

Réf. : ------------------------------------

Study and design of an assistance system for People with special needs with NAO ROBOT

Date : June 2022

Presented by :
ATHMANI WAFA

Favorable Opinion of the Supervisor:
RAHMANI NACER EDDINE

Favorable Opinion of the President Jury

Pr. DEBILOU ABDERRAZAK

Stamp and signature

بسم الله الرحمان الرحيم

In the Name of Allah, the most Merciful, the Gracious

ACKNOWLEDGMENTS

First and foremost, I’m very beholden to almighty God for giving me the will, knowledge and strength to complete this work.

I am extremely thankful to my teacher and supervisor Mr. NACER EDDINE RAHMANI for his invaluable advices, support and precious guidance during the course of this thesis

like also to pay my gratitude to Pr. DEBILOU ABDERRAZAK and Dr. BENLMIR OKBA for their important comments on this thesis.

Finally, I would like to thank all teachers that worked hard to make my journey at Biskra University valuable and beneficial.

DEDICATION

To my parents, the persons I owe everything , the persons whom I stood tongueless to express how grateful I am for everything they did for me and for making me who I am today. May God Bless You Beyond Any Known Measure

I am also very thankful to my sister Yasmine for teaching me everything she knows, and without her guidance I wouldn't achieve what I have achieved

To my brother and sisters, the strongest bond of my life. To my source of joy, happiness, strength and will. the persons who stood for me and been my shield whenever I needed them. May The Lord Protect you.

My thanks also to my friends A.narimane and B.tourkia, for helping each in every step in realizing this thesis.

And special thanks for senior colleagues T.chaouki and H.mounir for their for their precious help

Table of Contents

Table of figure

Abbreviations List

- **API**: Application Programming Interface
- **ARM**: **Autonomous robot mobile**
- **CCD**: Charge coupled device
- **CNN**: Convolutional Neural Network
- **COG** center of gravity
- **CPU**: central processing unit
- **DOF** : Degree of Freedom
- **DSPIC**: Digital Signal Controllers
- **FOV**: Filed of View
- **HD**: High Definition
- I**P**: Internet Protocol
- **JSON**: Java Script Object Notation
- **MID**: mobile internet device
- **MIT** : Massachusetts Institute of Technology
- **NN**: Neural Network
- **PLC:** programable logic controller
- **PNG**:Portable Network Graphics
- **RAM** :Random-access memory
- **RISC**: reduced instruction set computer
- **Robocup**: robot soccer world cup
- **SDHC**: San Diego Housing Commission
- **USB:** Universal Serial Bus
- **VGA**: Video Graphics Array,
- **Wi-Fi**: wireless network protocols
- **XML**: Extensible Markup Language
- **ZMP**: zero momnent point

Abstract

This thesis aims are to get to know the field of robot, specifically Nao Robot and get to know its sensors, capabilities and internal and external parts and how to deal with it, and learn its platforms and software and how to program it in order to make it useful for people with special needs.

In our projects, we used the Python programming language and many of its libraries like opencv for object and face detection and TensorFlow library for deep learning project… we access to robot cameras and sensor like and sonar and tactile sensors and we use speech recognition and text to speech API.

The project we apply it on nao robot are android application to control the robot and navigation algorithm also human following and train a hand gesture recognition module addition to localization and path planning.

تهدف هذه المذكرة إلى التعرف على مجال الروبوتات وتحديداً Nao Robot والتعرف على مستشعراته وقدراته وأجزائه الداخلية والخارجية وكيفية التعامل معها ، ومعرفة منصاته وبرامجه و طريقة استعمالها ، لتطبيق هذه المعارف في انجاز مشروع لذوي الاحتياجات الخاصة .

في عملنا هذا ، استخدمنا لغة برمجة Python مع العديد من مكتباتها مثل opencv للتعرف على الاشياء والألوان والوجه ومكتبة TensorFlow لمشروع التعلم العميق ... كذلك استخدمنا كاميرات الروبوت وأجهزة الاستشعار مثل الحساس فوق صوتي وحساس اللمس واستخدمنا مكتبة التعرف على الكلام و تحويل النص إلى كلام؛ كما قمنا باستعمال هذه المكتبات في انجاز خاصية تتبع الإنسان وتدريب وحدة التعرف على إيماءات اليد بالإضافة الى تحديد الموقع وتخطيط المسار.

كما قمنا بانشاء تطبيق android للتحكم في الروبوت عن بعد باستعمال WIFI و Bluetooth.

Introduction

Robots have always been a subject of curiosity for technologists and engineers and all people alike. Humanoids, robots with multiple degrees of freedom, have become popular research platforms as they are considered to be the future of robotics. robotics aims to bring technologies that allow autonomous systems to assist and help humans performing tasks that are difficult, repetitive or unpleasant. The human like design and locomotion allows humanoid robots to perform complex motions. This includes balancing, walking, standing up, step over or avoid obstacles, reaching destinations, and to navigate through cluttered environments without colliding with objects These abilities would make humanoid robots ideal assistants specially for disable people who are more likely to get benefit from it from brining object, doing daily task and keeping company in this thesis, we focused on getting to know the Nao Robot, its characteristics and capabilities its hardware component and software algorithms and platforms, and apply some project on it to make the best use of it and in order to accomplish the study we divide this thesis to four chapter.

To get a better view to field of robotic and robot in chapter one we talk about the robot characteristic and his fields of study also the technologies from his sensors, electronic and mechanic part, pneumatic and hydraulic system to his software development also we talk about the type of robot and each type role in human life and we focused on humanoid robot.

In chapter two we focus on nao robot and we talk almost of every part in It from the top of his head to bottom of his feet its components, sensors, motors joint, electronic and mechanic part, from what it made of its technologies and his software also we talk about his choregraph frame work and webots simulated robot for nao and give the first steps to get start utilizing the robot.

The hardware, software and technologies that help us and get used from it will be presented in the third chapter.

In the last chapter we present the project we applied on nao robot and we will discuss their result, this chapter is divided to five parts, the first is about how we make an android application contain buttons to control nao robot also speech recognition commands and live streaming from the robot for the second part we applied and algorithm of autonomous navigation on the robot, and third we teach nao robot to follow human and react to deferent event. in the fourth part we trained a hand recognition module so we can control nao robot with hand gesture and the last part for localization and path planning.

CHAPTER I:INTRODUCTION TO ROBOTS

I. ROBOTS

The term of industrial robot (robotics) has appeared in the early of 1960s after a period of rapid change in industrial automation, In North America, there was much adoption of robotic equipment in the early 1980s, followed by a brief pull-back in the late 1980s. Since that time, the market has been growing, A major reason for the growth in the use of industrial robots is their declining cost. through the decade of the 1990s, robot prices dropped while human labor costs increased. Also, robots are not just getting cheaper, they are becoming more effective faster, more accurate, more flexible [1].

I.1 Definition:

According to the Robotics Institute of America's (RIA) definition, a robot is a reprogrammable multipurpose manipulator created to move objects like materials, tools, or specialized equipment using a variety of programmed motions. Robots are intricate, adaptable machines from an engineering perspective that have a mechanical framework, a sensing system, and an automatic control system. The results of study in mechanics, electrics, electronics, automated control, mathematics, and computer sciences serve as the theoretical foundation for robotics as well [2].

I.1.1 Autonomous Mobile Robot (AMR)

A mobile robot is a machine controlled by software that use sensors and other technology to identify its surroundings and move around its environment robots can mainly walk, roll, jump, run, slide, skate, swim, and fly. According to their locomotion system, mobile robots can be classified into the following major categories:

I.1.2 Autonomous mobile robot types

A "vision" of how the robot will look and function is frequently the starting point of a custom robot design. AMRs can take countless forms, however the following are the most common [3].

I.1.2.1 Legged ARM’S

A series of point interactions between the robot and the ground define legged locomotion. The main benefits include the capacity to adjust and move on difficult terrain. The quality of the ground between such spots is not important as long as the robot can maintain appropriate ground clearance because only a set of point interactions are necessary. A walking robot can also cross a chasm or hole if its reach is greater than the breadth of the obstruction. The ability to expertly move items in the environment is the final benefit of legged locomotion. [3].

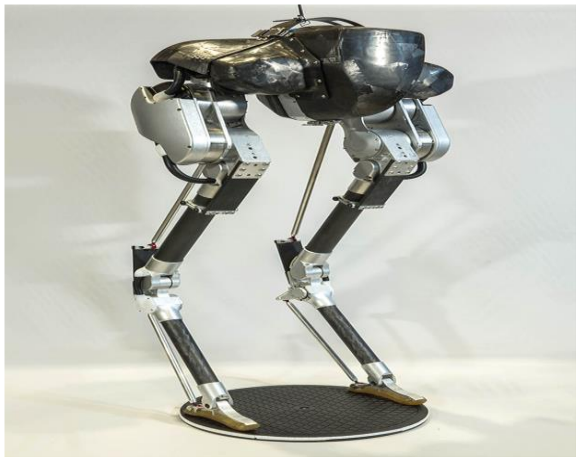


Figure 1 leges ARM's [4]

I.1.3 Aerial AMRs

Unmanned aerial vehicles (UAVs), known as drones, are aircraft that can fly autonomously or be remotely piloted from the ground but do not carry passengers or have pilots. Extrapolating from an English term that implies "drone," the word "drone" was created [3].

Figure 2 drone [5]

I.1.4 Wheeled AMRs

Wheeled mobile robots (WMR) are robots that move on wheels. Using wheels makes this design, which is fairly common, simpler to design, build, and program for mobility on flat and soft terrain. Additionally, WMR are simpler and more effective to control. The number of wheels on a WMR is unlimited [3].

Figure 3 wheeled ARM's [6]

I.2 Types of the robots

I.2.1 Industrial robots

painting and welding robots they distinguished with high degree of flexibility and high quality and precision so it’s can reach very difficult places.

I.2.2 Service Robots

- field robotics: machines that work in unstructured.
- environments (farm, factory, campiness..).
- professional cleaning.
- medical robots: for doing surgery and medical examination.
- entertainment: toys for kids.
- Mobile platforms: for chemical power plant, under see or remote areas and bombs fields.
- Humanoid robot: like Nao robot that we going to study in these thesis [7].

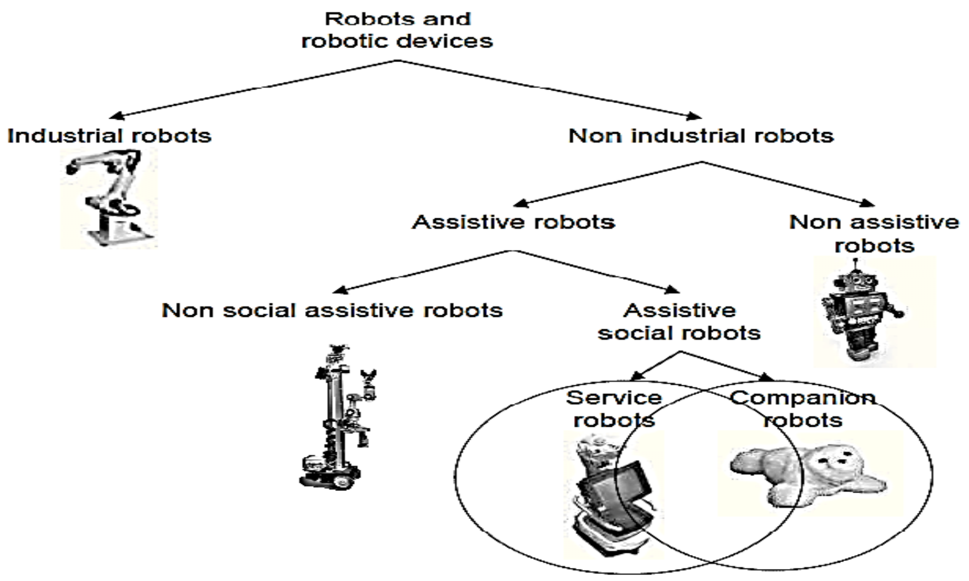


Figure I-4 types of robots [8]

I.3 robot characteristic:

- **Sensing**: light sensors like your eyes touch and pressure sensors like your hands chemical sensors (nose), Sonar sensors (ears) and taste sensors (tongue) will give your robot awareness of its environment.
- **Movement:** A robot must be able to move around in its surroundings. whether it is rolling on wheels, walking on legs, or propelled by thrusters.
- **Energy** Solar, electrical, or battery power could all be used to power a robot. Your robot's energy source will be determined by the tasks it must do.
- **Intelligence** and that by entering software and algorithms and artificial intelligence in the robot so it can do tasks without human interaction [10].

I.4 Technologies used in robot

I.4.1 pneumatic and hydraulic system:

consist in actuators with this kind of system there is input and output in the cylinder, through these we pump air for pneumatic system and clean filtered oil for hydraulic system to make the piston move outside and inside to provide us with linear force and motion which allows the robot to move [11].

I.4.2 mecanics:

reprsented in transmission system, it usually rotates and mechanism to transfer motion to all the necessary parts of a robot to create the motion, consist in Gears, Chains, Timing belts, Metal belts, cables and pulleys, Ball screws … [11].

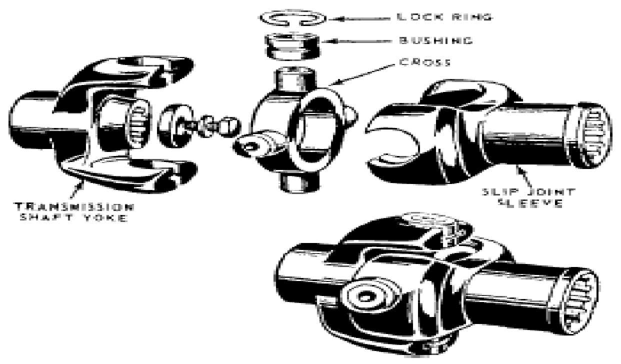


Figure I-5 mechanic parts [12]

I.4.3 Sensors:

which considered as the senses consist in switch, Force sensor, Potentiometer, Digital rotary Encoder is for measuring rotating degree, Tachometer (Essentially is a generator), Proximity sensors (A sensor is able to detect or recognize the presence of close objects), tactile sensor, cameras, microphones, ultrasonic sensors … [11].

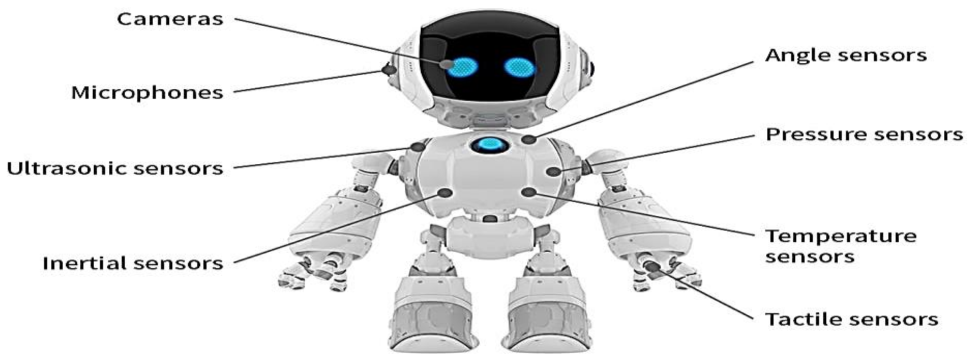


Figure I-6 robot sensors [13]

I.4.4 Electronic part:

Micro-controller and processors are the brain that controls the whole system, Programmable logic controller (PLC) has input and output that are used to create communication between sensors and actuators, Power Electronics are used for running motor drive and controlling the motor speed [11].

I.4.5 Algorithms and software

logic programming language through logical event sequence by planning the whole task at the beginning, then controlling the motors and actuators through using feedback signal that are obtained from sensors, also entering some smartness to the robot (object recognition, speech recognition deep learning and artificial intelligent) and give order to the other part to do the behavior [11].

I.5 Humanoid robot

In our thesis we are going talk humanoid robot , it consider type of legged autonomous mobile robot . about A humanoid robot is one that looks and functions like a person. The design might be for practical goals, such as engaging with human equipment and environs, or it could be for experimental objectives, such as studying bipedal walking. Humanoid robots feature a torso, a head, two arms, and two legs in general, while some may merely mimic a portion of the body, such as the waist up. Some humanoid robots have heads with eyes and lips that are made to seem like human characteristics. Humanoid robots known as Androids are designed to look like people [15].

Humanoid robots are now used as research tools in several scientific areas. Researchers study the human body structure and behavior (biomechanics) to build humanoid robots. On the other side, the attempt to simulate the human body leads to a better understanding of it. Human cognition is a field of study which is focused on how humans learn from sensory information in order to acquire perceptual and motor skills. This knowledge is used to develop computational models of human behavior [15].

CHAPTER II: NAO ROBOT

II. NAO ROBOT

humanoid robot for people to communicate with and packed with sensors, capable of mimicking human-like activity such as doing a specific task, recognizing faces and objects is used throughout the world in research, academics, industries and many more.

II.1 History

In 2005 Bruno Maisonnier create Aldebaran robotics, a manufacturer of humanoid robots targeted for customer, and build their first prototype of Nao based of strong principles

In 2006 NAO replace Sony's AiBo robot and chosen to be the official platform of Robocup (Robot soccer world cup) at least 5 robots of 16 teams ,At the end of 2008 more than 80 Robot were delivered ,Aldebaran upgrade its first generation of Nao in 2009 [16].

II.2 Nao hardware sensors

Nao is 58 cm tall and 4.5 kg of weight, Nao designed to be smooth with no sharp edges, round surfaces and no metallic part are visible, it has 25 degrees of freedom (DOF) five for each arm and one for each hand, two for the head, and eleven for the legs.

II.2.1 In Nao's head

The head is very sensitive part it holds:

- CPU (ATOM Z730, 1.6 GHz (Intel unveiled the Z530 CPU in early 2008 as an ultra-low-power 32-bit x86 microprocessor designed exclusively for Mobile Internet Devices (MID). The Z530 is made using a 45 nm technology which based on the Bonnell microarchitecture. This CPU has a 1.6 Ghz clock speed, , and a 220-mW average power [17].) with 1GB of RAM and 2GB of flash memory an 8GB micro SDHC.
- 2 1280x960 pixel color cameras the top in his forehead and the bottom in his chin.
- 4 microphones around the head with frequency in range of 20Hz-20kHz.
- 2 loud speakers in the ears with frequency in range of up to ~20kHz.
- Tree tactile sensers on the top of his head.
- The Wi-Fi dongle.
- The ethernet and USB connectors in the back of his head.
- Two infra-red emitters and receivers located one in each eye.
- Plenty of Leds around the eyes and the ears on the head.

- NAO can move the head by 239°horizontally and by 68° vertically [18].

Figure II-1 nao head from inside [20]

II.2.2 NAO's chest and back

- four ultrasonic or sonar sensors in the front.
- battery Lithium-Ion type with max charge voltage 24.9V and max charge current 3.0A / 2.0A with charging duration to 5h located in the back.
- button (for on/off and configuration).
- An inertial measurement unit computed by the main CPU, located in the trunk, can be used to detect disturbances that can affect the balance of the robot (push off, slope) receives the position command for each joint and transmits that information through a RS485 bus toward DSPIC microcontrollers located in the limbs [16] [18].

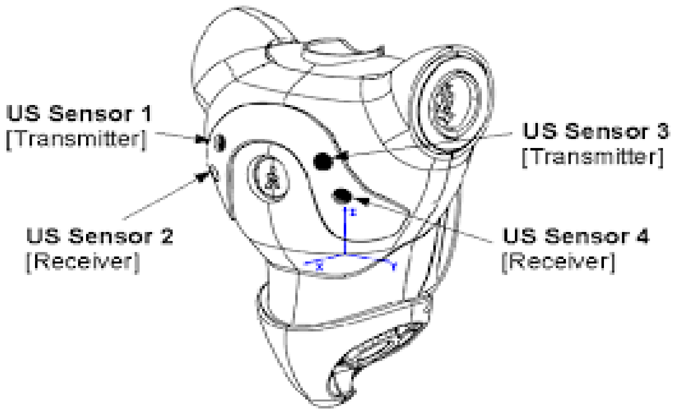


Figure II-2 Nao chest [19]

II.2.3 Nao’s hip, legs and feet

Each leg has 2 DOF at the ankle, 1 DOF at the knee, and 2 DOF at the hip. The pelvis is equipped by a unique mechanism consisting of 2 linked joints at each hip. The rotation axis of these two joints is inclined at 45◦ towards the body. the horizontal axis rotary joint at the waist and the rotary joints of vertical axis for each leg hip, the coupling of the pelvis joints prevents the trunk from rotating along the vertical axis (yaw rotation), although this is not a problem for walking and other motion behaviors. The proposed technique has a number of benefits. The pelvis can be driven by just one motor. This reduces construction costs and frees up room in the trunk's lower section. this structure aids in improved power distribution between the hip roll joint and the pelvic joint. and creates a specific motion style to the NAO humanoid [21].

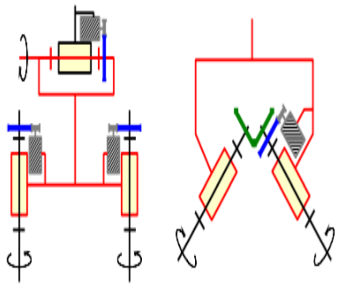


figure II-3: Left-hand side: classical set of three rotary joints, one horizontal axis rotary joint at the waist and two vertical axis rotary joints for the legs. Right-hand side: coupled inclined axis rotary joints for the NAO pelvis [21]

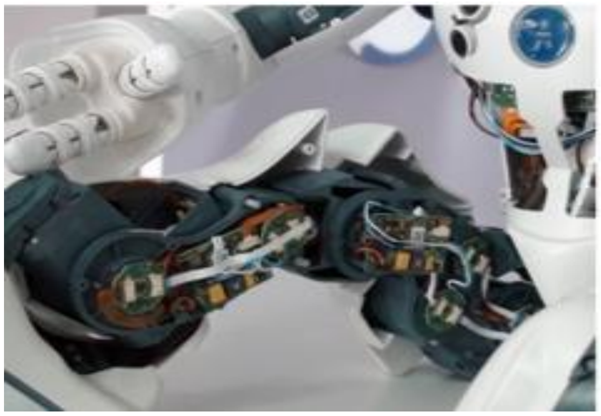


Figure II-4 motors in nao leg [20]

And one more sensor used to evaluate the current status of the robot are the pressure sensors located in the soles of each of the feet. The four force-sensing resistors under each foot allow the robot to detect if the foot is on the ground or not and can give an estimation of the position of the center of mass and of the center of pressure. Mechanical bumpers on the “toes” can also detect a collision of the foot with an obstacle [21].

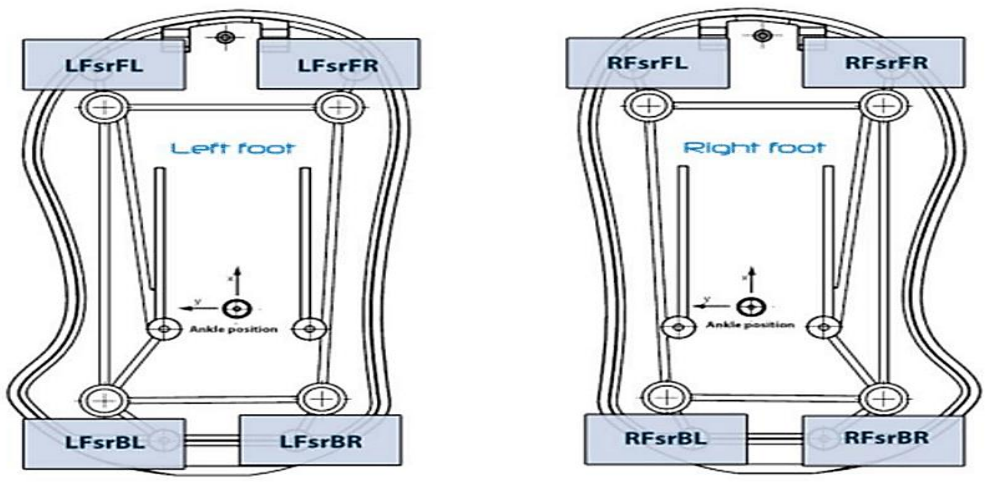


Figure II-5 NAO Feet [22]

II.2.4 Nao hands

The NAO robot has three touch sensors on each hand that will be used to collect input from the user. These sensors are depicted in the figure below. They can give back touch data to the robot, on which the robot's reactions can be based [23].

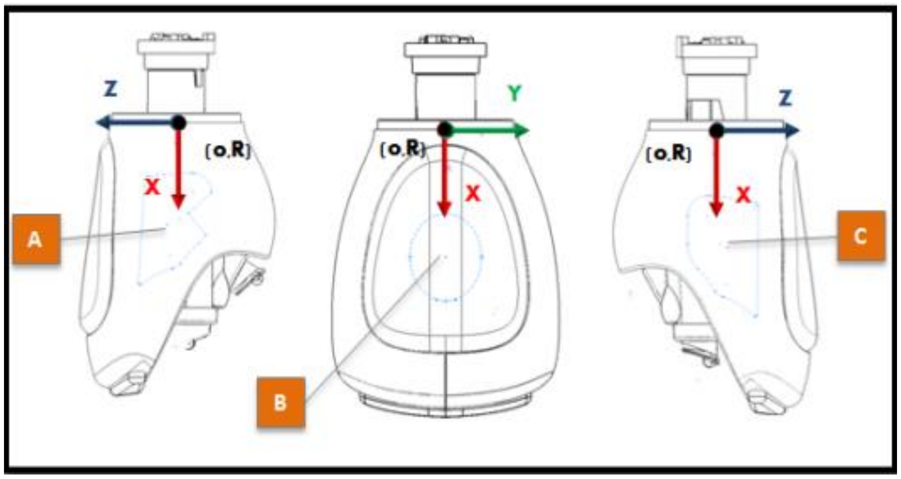


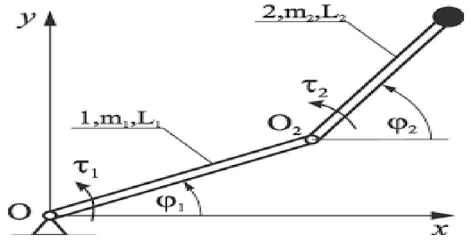
Figure II-6 nao hands [24]

II.2.5 Nao arms and shoulder

in addition, each arm features have 2 DOF at the shoulder, 2 DOF at the elbow, 1 DOF at the wrist and 1 DOF for the hand's grasping.

These parts using specific actuators, based on DC motors, gears, and position sensors were created.10 degrees of freedom in NAO rely on the same type of actuation module called the cylindrospheric module This module includes two motors that provide two perpendicular motions This module is very compact and makes the integration of motorization very easy and focused on the physical appearance of the robot [16] [21]

Figure II-7 Two DOF cylindrospheric module used in the neck, shoulder, and elbow of NAO



II.3 Nao joint and their names

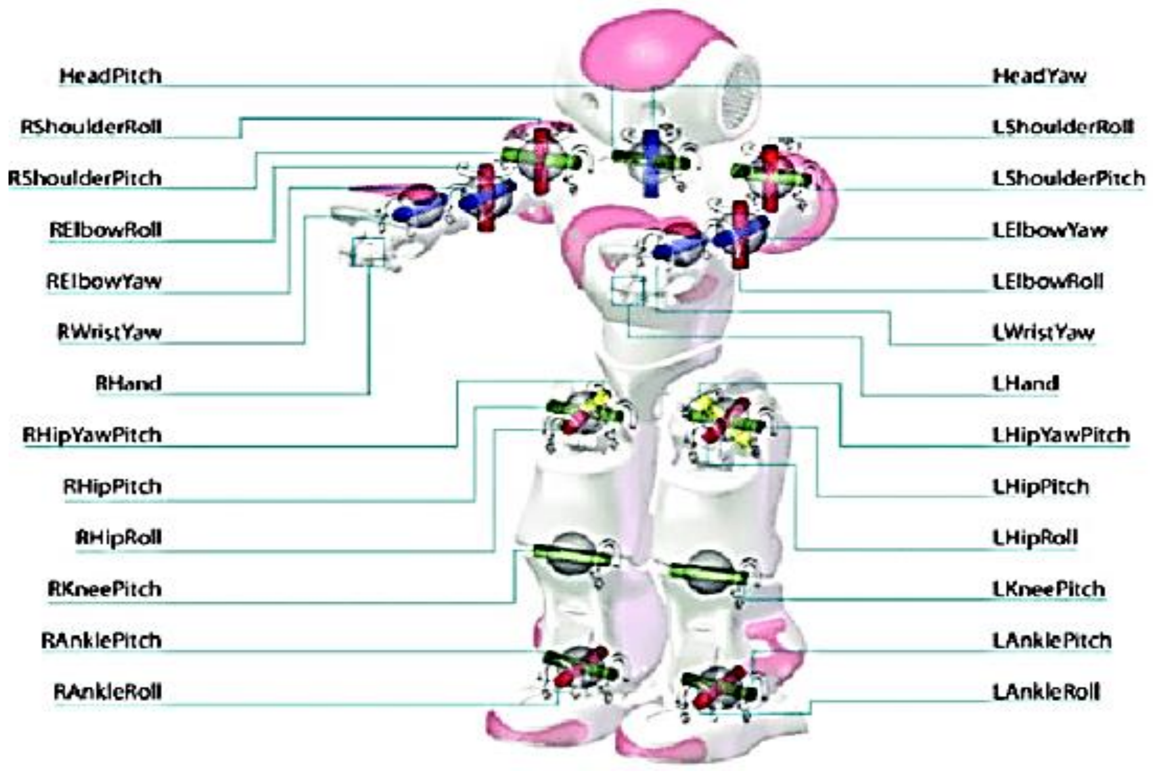
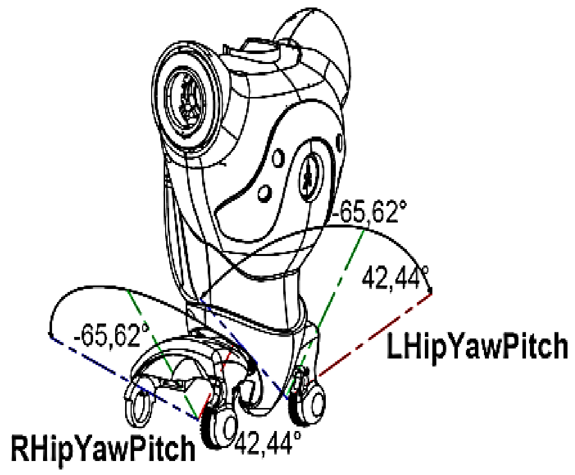
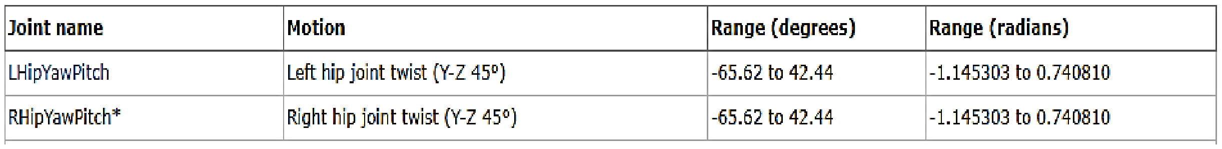


Figure II-8All joints in NAO robot and initial position [25]

II.3.1 The angles of the hip joints

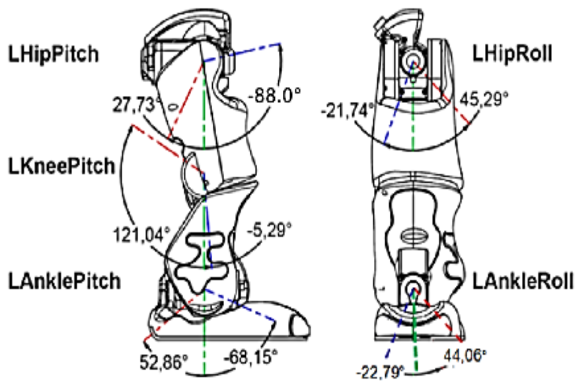


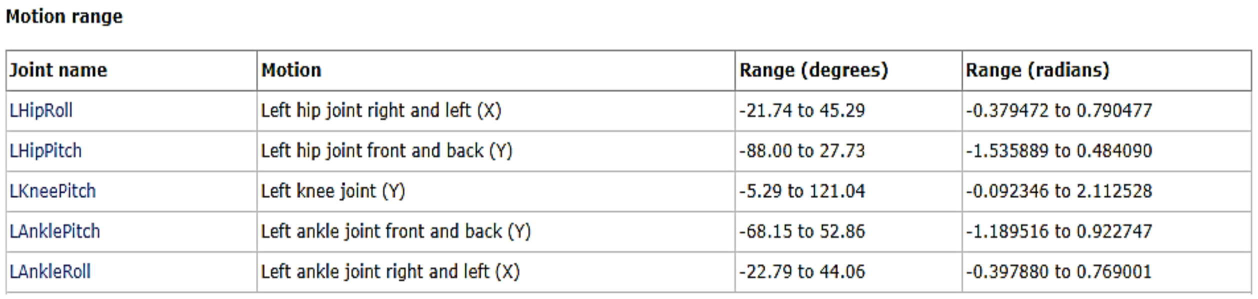


Joint name	Motion	Range (degrees)	Range (radians)
LHipYawPitch	Left hip joint twist (Y-Z 45°)	-65.62 to 42.44	-1.145303 to 0.740810
RHipYawPitch*	Right hip joint twist (Y-Z 45°)	-65.62 to 42.44	-1.145303 to 0.740810

Figure II-9 Pelvis angle [24]

II.3.2 Leg joint





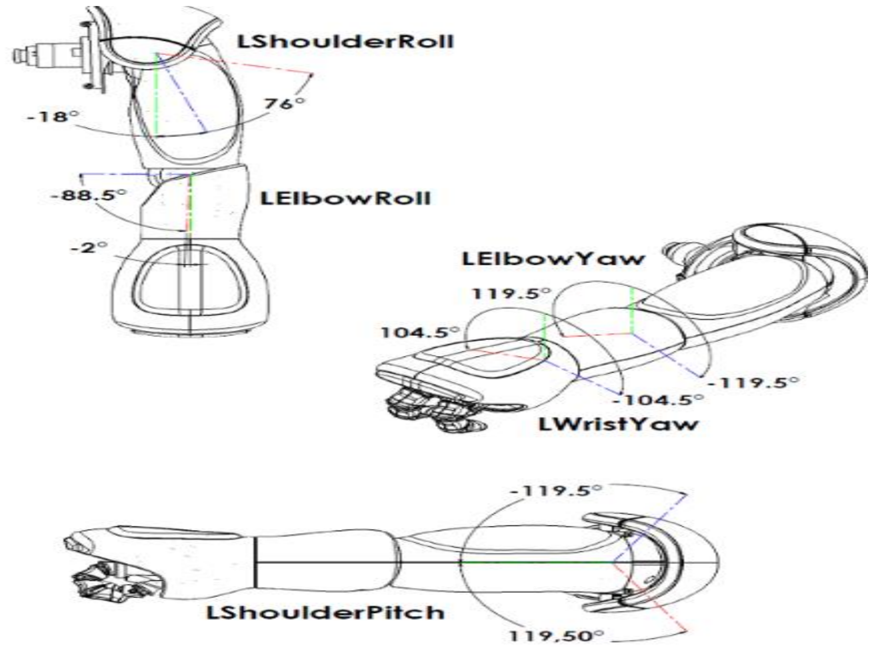
Motion range

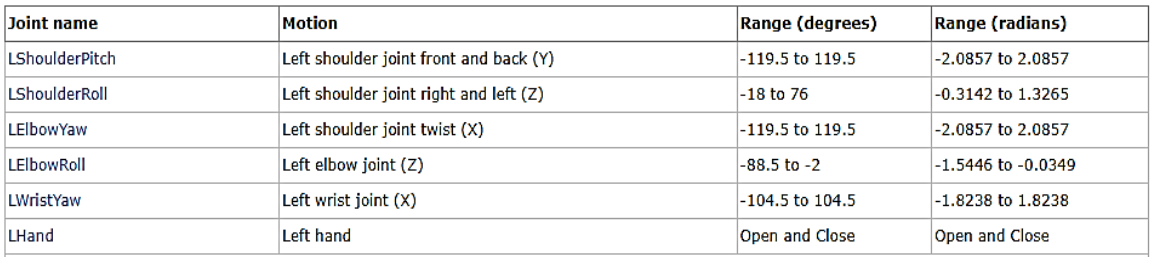
Joint name	Motion	Range (degrees)	Range (radians)
LHipRoll	Left hip joint right and left (X)	-21.74 to 45.29	-0.379472 to 0.790477
LHipPitch	Left hip joint front and back (Y)	-88.00 to 27.73	-1.535889 to 0.484090
LKneePitch	Left knee joint (Y)	-5.29 to 121.04	-0.092346 to 2.112528
LAnklePitch	Left ankle joint front and back (Y)	-68.15 to 52.86	-1.189516 to 0.922747
LAnkleRoll	Left ankle joint right and left (X)	-22.79 to 44.06	-0.397880 to 0.769001

Figure II-10leg angle [24]

II.3.3 The angles of Arms joints

The picture below represents the arm joint for the left shoulder





Joint name	Motion	Range (degrees)	Range (radians)
LShoulderPitch	Left shoulder joint front and back (Y)	-119.5 to 119.5	-2.0857 to 2.0857
LShoulderRoll	Left shoulder joint right and left (Z)	-18 to 76	-0.3142 to 1.3265
LElbowYaw	Left shoulder joint twist (X)	-119.5 to 119.5	-2.0857 to 2.0857
LElbowRoll	Left elbow joint (Z)	-88.5 to -2	-1.5446 to -0.0349
LWristYaw	Left wrist joint (X)	-104.5 to 104.5	-1.8238 to 1.8238
LHand	Left hand	Open and Close	Open and Close

Figure II-11 the angles of nao arm [26]

II.4 NAO LEDS

On NAO, there are over 80 LEDs.. These LEDs are utilized to inform the user of the robot's status as well as to light up the robot when a sensor is pushed or when an application makes use of them. The messages conveyed by the LEDs are critical and must be treated carefully. During startup and when the NAO is running, the chest button illuminates. During boot, a blue light shows that a firmware update has been requested, while a green steady light indicates that the boot process has been stuck. The chest light reveals the level of the battery charge or the status of the NAO once it is turned on NAOqi isn't up and operating. The LEDs on the eyes in animation mode show whether the robot is listening, storing, or accepting a position [23].

II.5 Motors and position sensors

- 25 motors (25 DOF in each joint there is a motor) with 21 Maxon coreless motors.

  - **Position sensors**

- 36 Magnetic Rotary encoders with Resolution of 12bit and sensors for each leg's joint: 1 motor axis, 1 joint [20].

II.5.1 Electric current sensors on each joint

- 20 embedded micro-controllers.
- 18 DSPic 40MPS/16 Bits (DSPic Digital Signal Controllers for high-performance and robust designs, offer low power consumption flexible peripherals, and a comprehensive ecosystem of software and hardware tools to simplify and speed up the project, from the technologies that used, motor control, digital power conversion, low-power security, advanced analog integration and functional safety [27].
- ARM9 96MHZ (ARM9 from ARM family (Advanced RISC) is a series of reduced instruction set computer (RISC) instruction set designs for computer processors [28].

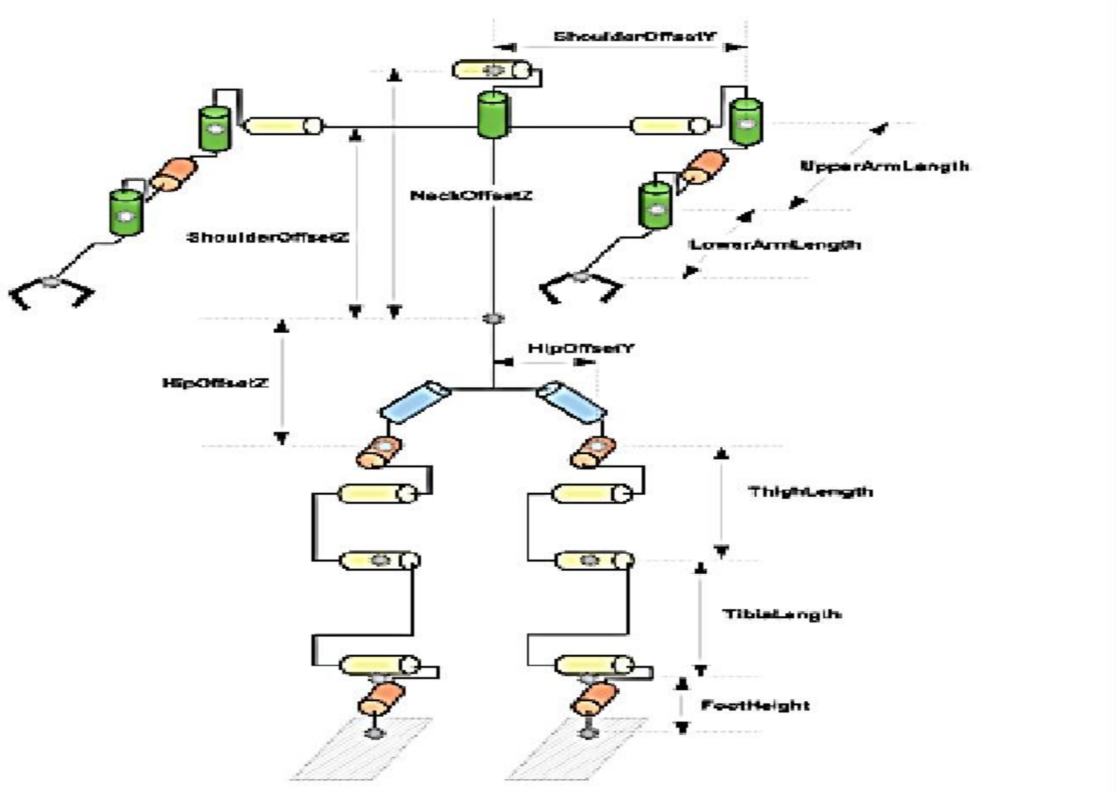


Figure II-12 motors and position sensor [20]

II.6 Electronic architecture

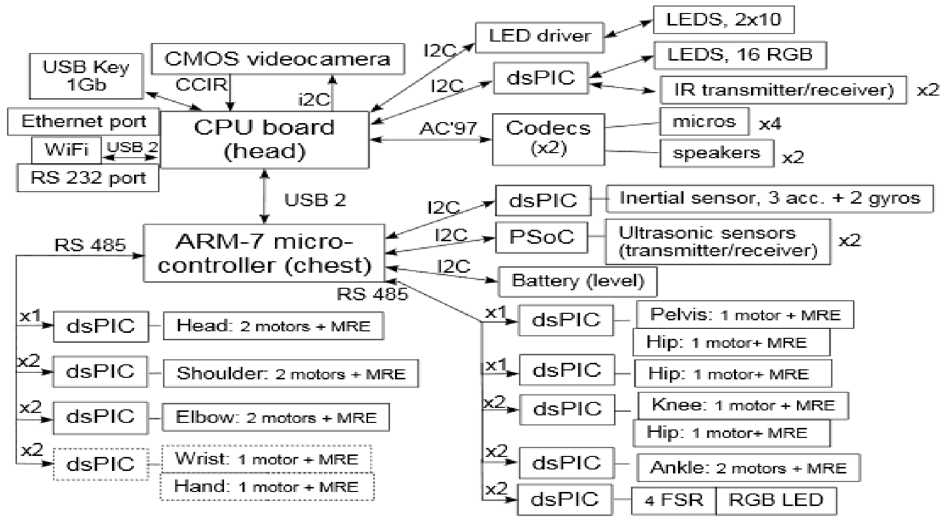

USB Key 1Gb
CMOS videocamera
CCIR
i2C
Ethernet port
WiFi
USB 2
RS 232 port
CPU board (head)
LED driver
LEDS, 2x10
I2C
dsPIC
LEDS, 16 RGB
IR transmitter/receiver) x2
AC'97
Codecs (x2)
micros x4
speakers x2
USB 2
ARM-7 micro-controller (chest)
RS 485
dsPIC
Inertial sensor, 3 acc. + 2 gyros
PSoC
Ultrasonic sensors (transmitter/receiver) x2
Battery (level)
x1 dsPIC Head: 2 motors + MRE
x2 dsPIC Shoulder: 2 motors + MRE
x2 dsPIC Elbow: 2 motors + MRE
x2 dsPIC Wrist: 1 motor + MRE
Hand: 1 motor + MRE
x1 dsPIC Pelvis: 1 motor + MRE
Hip: 1 motor+ MRE
x1 dsPIC Hip: 1 motor+ MRE
x2 dsPIC Knee: 1 motor + MRE
Hip: 1 motor+ MRE
x2 dsPIC Ankle: 2 motors + MRE
x2 dsPIC 4 FSR RGB LED

Figure II-13 electronic architecture [21]

Via an Ethernet port Audio, video, Wi-Fi, and other sophisticated modules are all controlled by the CPU. Information is distributed by a single ARM7-60MHz microprocessor in the torso. in each actuator module Microcontrollers (Microchip 16 bit) DSPic via RS485 bus links the ARM7 processor to the other. DSPic modules in the top section of the microcontroller one that links the ARM7 to the rest of the body Lower-body DSPic modules. This bus Increased data throughput is possible with partitioning. The CPU connects with the ARM-7 microcontroller. with a potential throughput of 11[M bits/s]. It used to regulate the stability of the robot. [21]

II.7 Walk process in nao robot

II.7.1 Zero moment point (ZMP)

is a concept related with dynamics and control of legged locomotion, e.g., for humanoid or quadrupedal robots. It specifies the point with respect to which dynamic reaction force at the contact of the foot with the ground does not produce any moment in the horizontal direction, i.e., the point where the sum of horizontal inertia and gravity forces is zero. The concept assumes the contact area is planar and has sufficiently high friction to keep the feet from sliding [29].

II.7.2 The linear inverted pendulum

Is a point-mass model that translate dynamics of a legged robot's locomotion [30].

II.7.3 Inverse kinematics

is the use of kinematic equations to determine the motion of a robot to reach a desired position. For example, to perform automated bin picking, a robotic arm used in a manufacturing line needs precise motion from an initial position to a desired position between bins and manufacturing machines.

II.7.4 The centre of gravity (COG)

of the human body is a hypothetical point around which the force of gravity appears to act. It is point at which the combined mass of the body appears to be concentrated. Because it is a hypothetical point, the COG need not lie within the physical bounds of an object or person.

- The NAO is provided with an open-loop walk engine, A basic ZMP trajectory is constructed using user-specified step values to build walk patterns online an inverted pendulum model is used to convert the ZMP trajectory into a centre of gravity (CoG) trajectory, Using inverse kinematics, this CoG trajectory is then monitored and maintained during single and double support phases [Gouaillier et al., 2008]. The swing leg's path is cycloid [29].

The NAO's low-level controller for joint position takes the shape seen in Figure bellow . The scaling factor Ks (It ranges from 0 to 100%) is added to the PWM duty cycle generated by the controller for the motor. This effectively adjusts the power available to each motor and the parameter Ks provides a sort of stiffness control; setting Ks to a lower value reduces the torque of the motor, resulting in softer trajectory tracking under load. The ALWalk engine provides the required joint locations. [29].

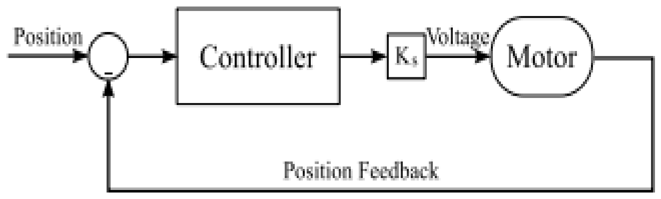


Figure II-14 walk process

II.7.5 Walk control

The data from NAO's joint sensors is used to adjust his walk. This makes the walk more resistant to small disruptions and absorbs frontal and lateral torso oscillations.

NAO can walk on a variety of floor surfaces, including carpet, tiles, and hardwood flooring.

foot planner used for by the walk process irrespective of the walk control functions , uses an ALMath::Pose2D multipication (math library) It is composed of a translation by pX and pY, then a rotation around the vertical z axis p Theta. and using clamp algorithm to avoid foot collisions during planning [24]

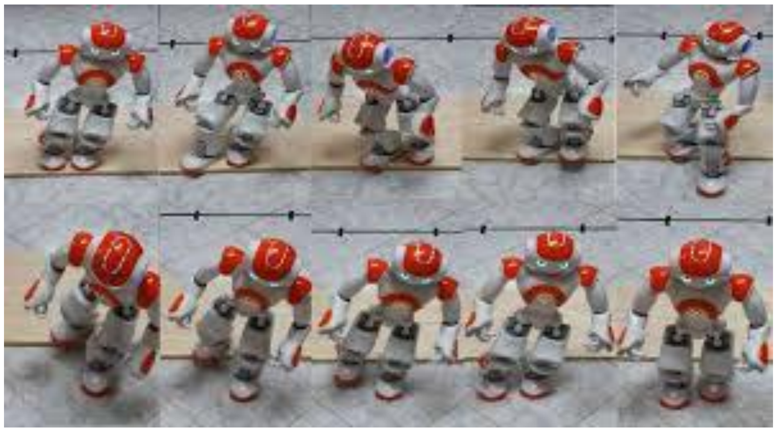


Figure II-15 walk stage

II.8 Enhanced audio and visual capabilities:

Camera: high sensitivity in VGA for better low light perception. For image processing work on the robot CPU, use up to 30 images/second in HD resolution, and his camera can see at 61° horizontally and 47°vertically. [18]

Object Recognition: Once the object is saved in Nao's software, if he sees it again, NAO is able to recognize and say what it is. [18]

Face Detection and Recognition: NAO can detect and learn a face in order to recognize it next time [18]

Text to Speech: NAO is able to speak up to 9 languages. By inserting the text and modify voice parameters in his software as you wish. NAO will say the text correctly [18].

Automatic Speech Recognition: NAO is able to hear you from 2 meters away, recognize a complete sentence or just few words in the sentence [18].

Sound Detection and Localization: NAO is able to detect and localize in the space thanks to microphones all around his head [18].

Fall Manager: NAO can protect himself with his arms and stand up when he almost falls [18].

II.9 Nao robot software

Nao robots have a wide range of functions and capabilities that may be used as desired by the operator. To keep the sensors and hardware in order

the NAO comes with embedded software which run on the motherboard and let the behavior act independently, and Desktop software, which is created by the user, enables for remote execution of new behavior on the robot, OpenNAO is a bespoke operating system created by Aldebaran for the robot. It's a Linux-based operating system, therefore it uses the Linux platform for navigation and code execution. the main development software for the robots is NAOqi [23].

II.10NAOqi

NAOqi is built to address the common robotics needs which include: parallelism, resources, synchronization, and events. Parallelism refers to the ability of NAOqi to manage lot of data through breaking down a program into smaller parts and processing them at the same time.

Synchronization occurs when tasks assigned to robots begin at a predetermined time rather than when the task is requested. This is accomplished via the NAOqi's internal clock, which gives more predictable timing and software behavior than asynchronous timing.

events are internal flags that are raised whenever a certain robot activity happens. This allows programmers to locate such behavior without having to go through data from multiple sensors and joints [23] [30] [31].

NAOqi offers uniform communication across audio, motion, and video modules, consistent programming, and homogeneous information exchange, NAOqi is cross-platform which means it can be developed in C++ and Python on the most of the cases the behaviors developed in Python and services in C++ Naoqi's consider as broker which performs two main roles:

- directory service that allows the access to attach modules.
- network access: allowing the methods of to be called from outside [23] [30] [31]

II.10.1.1broker process

A broker is an executable and a server that can listen to remote commands on IP and port

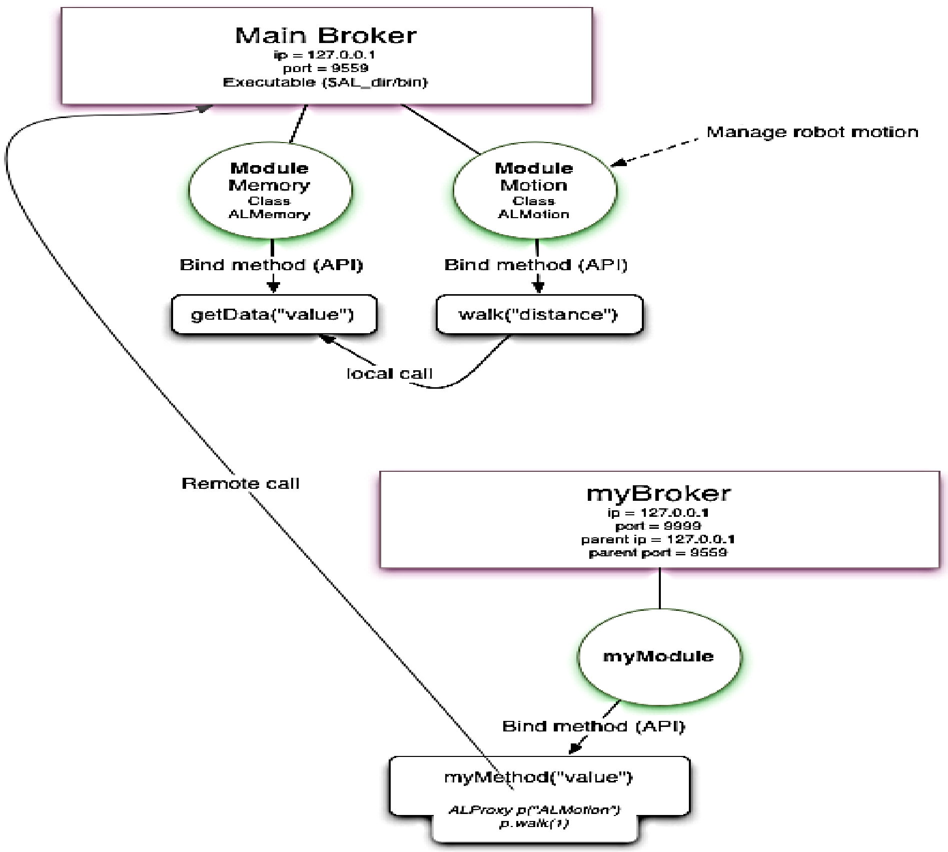


Figure II-16broker process [32]

II.10.2 Naoqi Modules:

Typically, each Module is a class within a library In the constructor of a class that derives from ALModule, Each module contains various methods, there is two kind of modules [33].

II.10.2.1 Local modules:

Local modules are two (or more) modules launched in the same process. They communicate using only one broker and we can create event to them [24].

II.10.2.2 Remote modules:

are modules which communicate using the network. A remote module needs a broker to speak to other modules [24].

II.10.2.3 Proxy:

The Proxy is an object containing all module methods

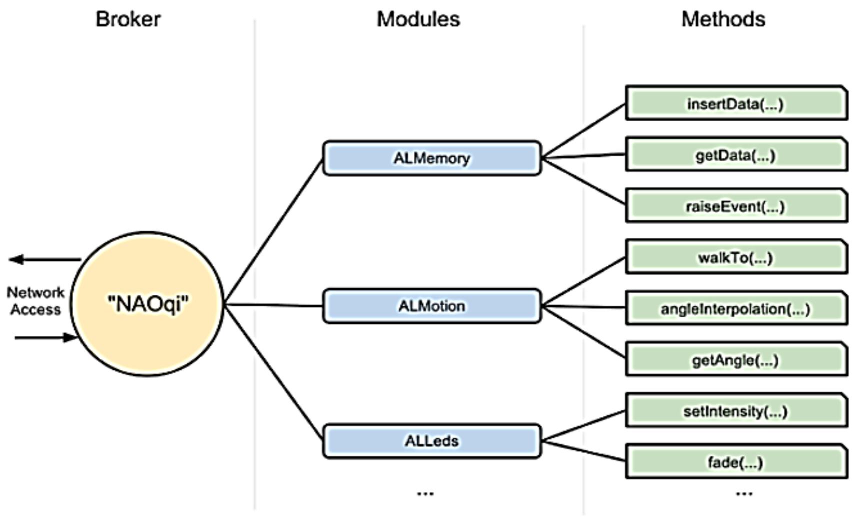


Figure II-17 NAOqi process [24]

II.10.2.4 Basic modules:

Naoqi framework allows homogeneous communication between different modules (motion, audio, video), homogeneous programming and homogeneous information sharing

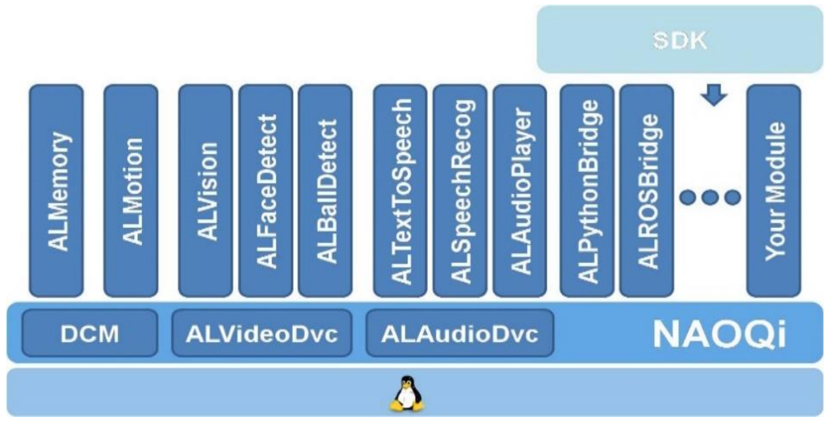


Figure II-18 Software architecture [20]

II.10.2.4.1 ALMemory

in the robot memory. Users can store data inside ALMemory the modules use write and read methods to access data by name. The data can be a binary or an analog value, The value will change permanently, and an event will be triggered at the end of each sample period. When the data goes through a threshold, a dedicated data is set to 1 in ALMemory. This data will be used as trigger by the application., a dedicated module called Device Communication Manager (DCM) is in charge of interfacing ALMemory with sensors and actuators [24].

II.10.2.4.2 ALmotion

Almotion module is the class that provide methods that allow Nao to move and change position and contain 4 main functions

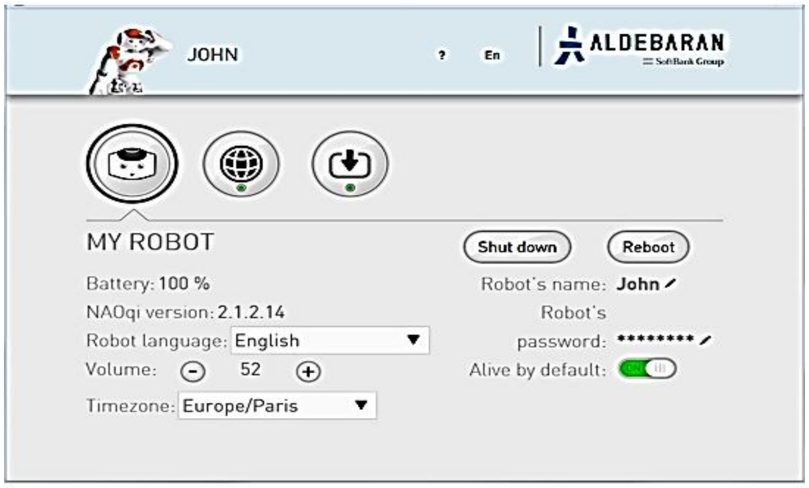
II.10.2.4.2.1 Joint stiffness functions (basically motor On-Off) [24].

II.10.2.4.2.2 joint position functions (interpolation, reactive control) [24].

II.11 The first step with Nao

II.11.1 The real robot:

We need a network cable a computer , the robot than We plug the cable in the ethernet connector in the back of Nao head and plug the other side in the electrical socket or in the ethernet connector of the computer and the we press long at the button in Nao's chest until he wakes up, he will say numbers, its his IP address Write down those numbers in your web page and u will access directly to your robot web page, it would have the information of the status of the robot



MY ROBOT

Battery: 100 %
NAOqi version: 2.1.2.14
Robot language: English
Volume: 52
Timezone: Europe/Paris
Robot's name: John
Robot's password: ********
Alive by default:

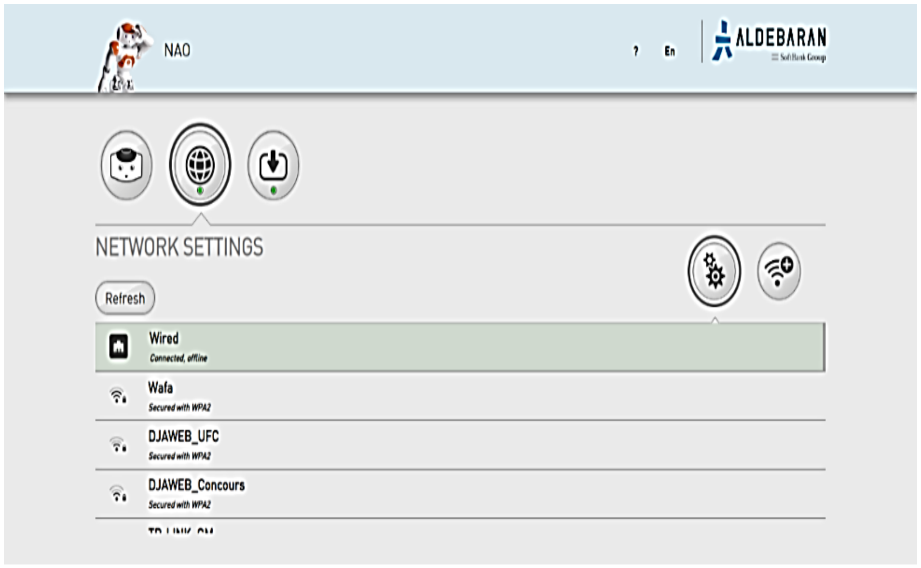
Figure II-19 nao web page [24]

II.11.2 Connect the robot to Wi-Fi

1- Remove the hatch behind the head of the robot to access to the Ethernet socket.
2- Plug an Ethernet cable.
3- Connect the Ethernet cable to your Internet box.

4- Access the NAO Web page and log-in.

5- In the Network settings page, choose and configure a WiFi network.



Wired
Connected, offline
Wafa
Secured with WPA2
DJAWEB_UFC
Secured with WPA2
DJAWEB_Concours
Secured with WPA2

Figure II-20 nao robot wifi connexion *[24]*

II.11.3 The simulated robot

To test your project and be sure of the safety of your robot or when u don’t have a robot Nao Webots robot consider the official simulated robot that deal with Aldebaran company

II.11.4 Webots

Webots is a multi-platform open-source desktop program for simulating robots. It comes with everything you need to create, program, and simulate robots. It extensively used in business, education, and research. Webots has been the primary product of Cyberbotics Ltd. since 1998. [36], Webots is a realistic physical world. Create a virtual environment where it can regulate the dynamic (speed, weight) and collisions with more than 50 items customizable by manipulating mass and inertia. With the console, inspect warnings and problems, and immediately obtain sensor data in Robot View. [37]

II.11.5 Naoqism controller

The controller API is the programming interface that gives you access to the simulated sensors and actuators of the robot. For example, including the webots/distance_sensor.

Naoqism is controller used to connect a Webots-simulated NAO robot to the NAOqi programming interface, including the Choregraphe graphical programming interface. [38]

II.11.6 Nao modules in webots

- The version module corresponds to the real Nao version. The supported versions are "3.3", "4.0" and "5.0". The main difference between these models is the different calibration of the physics. The field of view of the cameras is slightly different too. Note that each version has a different weight repartition in their bodies, the best contact properties in the simulated world aren't always the same.
- The degreeOfFreedom module corresponds to the degree of freedom of the real Nao. For versions 3.3 and 4.0 of Nao, the supported degreeOfFreedom values are 25 and 21. This corresponds to a model respectively with and without articulated fingers. [39]

II.11.7 Nao worlds in webots

• The "nao demo.wbt" world is a basic C controller that controls the motions of a Nao robot via keys. [39]

• The "nao indoors.wbt" environment contains a Nao robot in an apartment with normal living room items

• In the "nao robocuo.wbt" environment, a Nao robot stands on a soccer pitch, ready to score a goal

• The "challenge.wbt" environment contains the scenario from Aldebaran Robotics' 2013-2014 Nao Challenge contest. The robot must pick up a key from a door and place it in a key pot [39].

• The "get the candies.wbr environment comprises one of Aldebaran Robotics' Nao Challenge situations for the 2014-2015 round. The robot must transport sweets iron a shelf to a wagon [39].

• A robot soccer game scenario based on the Robocup's Standard Platform League (SP/) regulations can be found in the "robotstadium nao vs robotis-op2.wbt" world. It pits five Nao robots against five ROBOTIS OP2 robots [39].

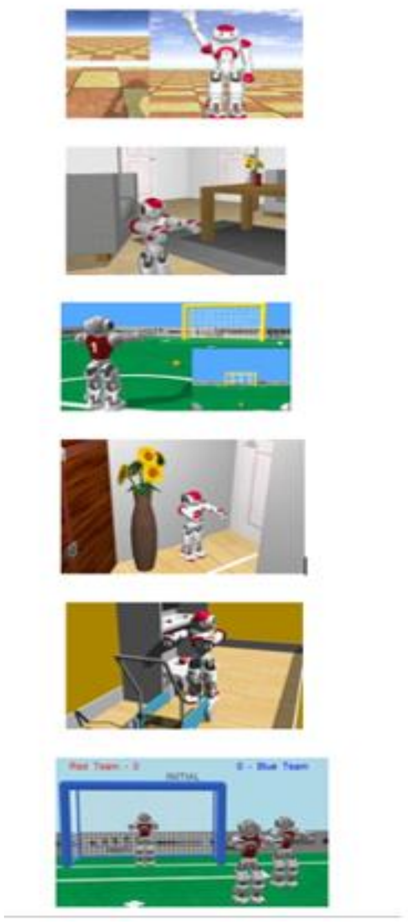
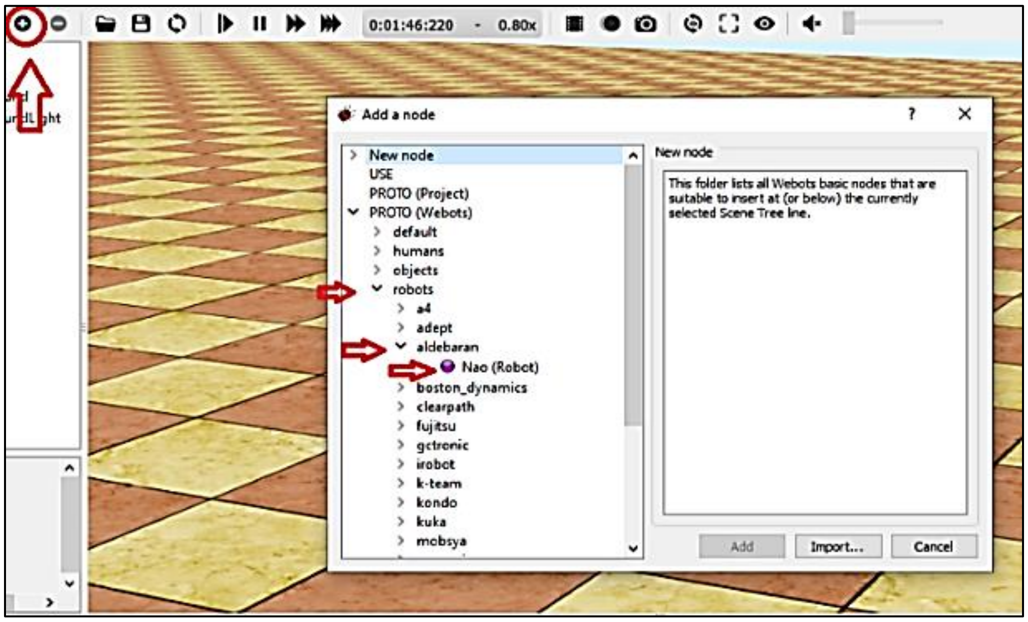


Figure II-21webots world for nao [33]

II.11.8 Starting with webots

- First enter nao to the demo simple world to
- By clicking on add button than ROBOT(Webots) than Aldebaran than Nao

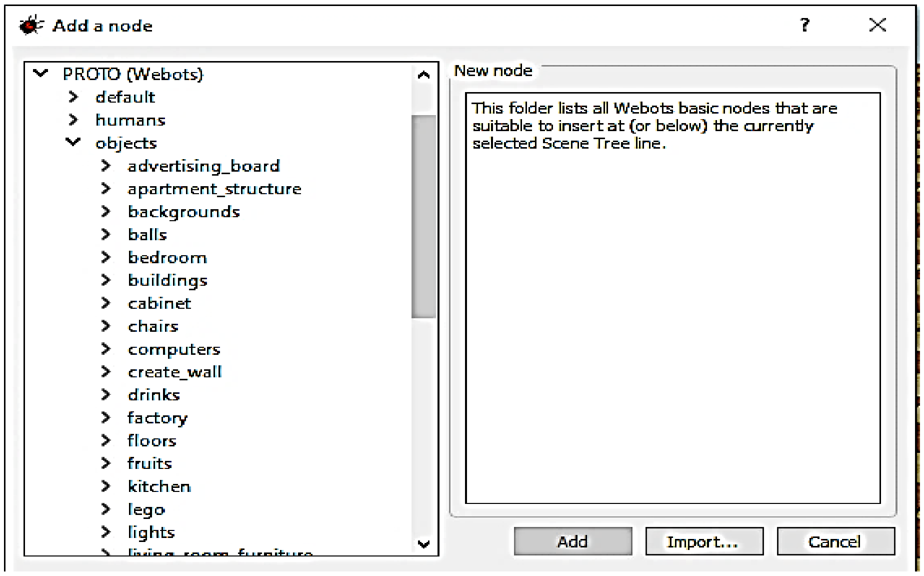


New node

This folder lists all Webots basic nodes that are suitable to insert at (or below) the currently selected Scene Tree line.

Figure II-22 entering Nao robot to webots environment

U can modify the environment as u want by doing almost the same process and u can move the robot or any object by the three axis



New node

This folder lists all Webots basic nodes that are suitable to insert at (or below) the currently selected Scene Tree line.

Figure II-23 list of environments in webots

II.12 Programing Nao Robot :

To make Nao walk speak and whatever u wand they provide us with two ways to do so either with his software called Choregraphe who utilize Naoqi development software and there is supported programing languages C++, python, java, MATLAB, Net

II.12.1 Choreographe

Choregraphe is a desktop program that runs on a variety of platforms. It gives youthe ability to:

- Make animations and actions.
- Put them to the test on a virtual robot or a real one.
- Control and monitor NAO. Without writing a single line of code, Choregraphe allows you to design extremely complicated actions (such as interacting with people, dancing, sending e-mails, and so on). It also allows you to customize a Choregraphe behavior by adding your own Python code. [37]

II.12.1.1 To connect robot to choreograph:

II.12.1.1.1 Simulated robot:

Choreographe provide a simulated robot without the outside environment to test the actions Click on green connection icon or connection in the top of the tool bar

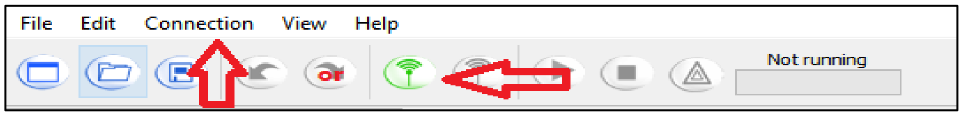
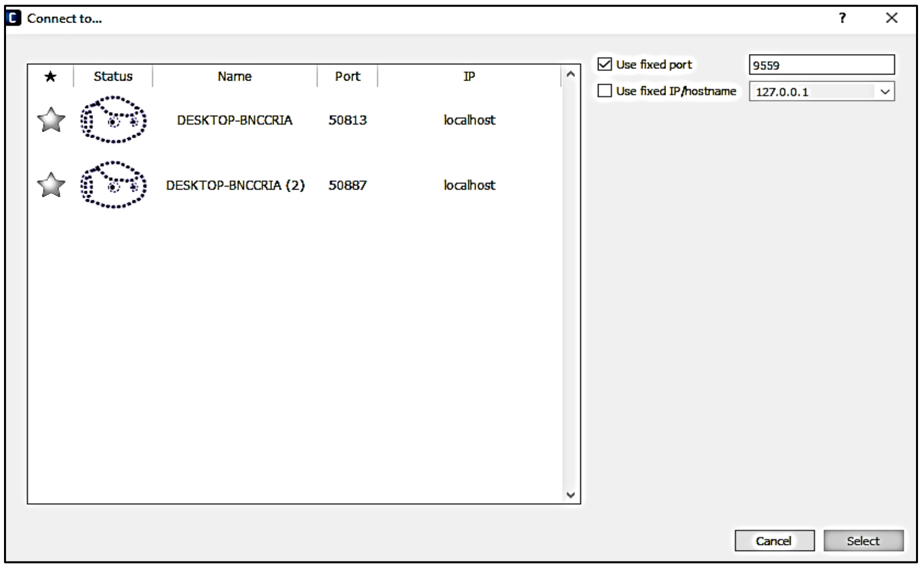


Figure II-24how to connect Nao robot

- When u see a robot with bleu broken lines its mean that is a simulated robot
- It shows the IP address and the name of the robot
- choose one of them and you are ready to test the action



Connect to...

★	Status	Name	Port	IP
☆		DESKTOP-BNCCRIA	50813	localhost
☆		DESKTOP-BNCCRIA (2)	50887	localhost

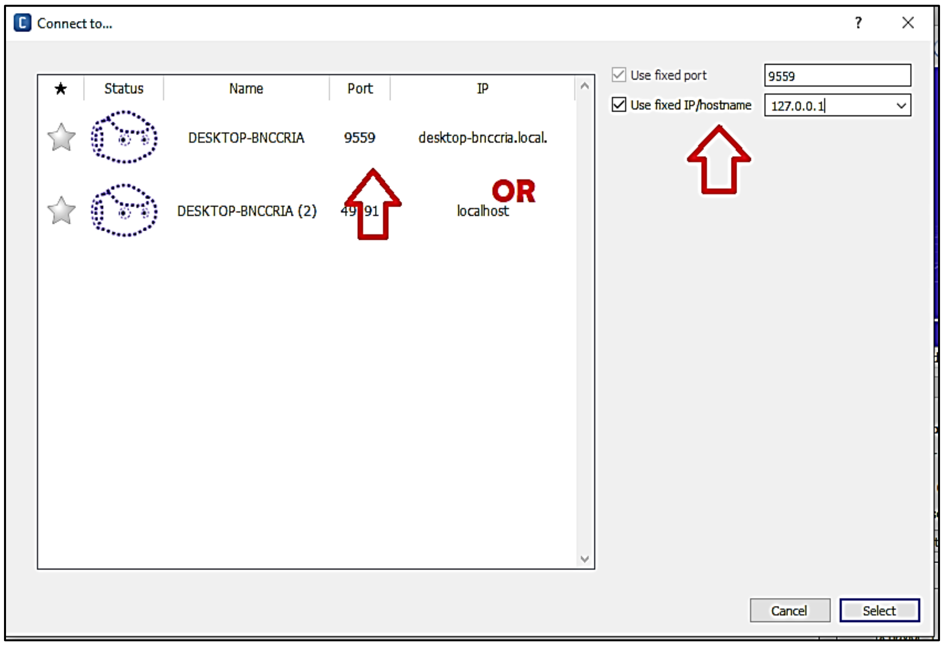
☑ Use fixed port: 9559

☐ Use fixed IP/hostname: 127.0.0.1

Figure II-25how to connect simulated Nao in choregraph

II.12.1.1.2 Connect the Webots Nao and real robot to choreograph

- First run the webots and choreograph and make sur the simulation is running
- Webots Nao robot have a local IP address
- Go to choreographe and click connection and u will see a simulated robot with 9559 IP address or click on fixed IP address



Connect to...

★	Status	Name	Port	IP
☆		DESKTOP-BNCCRIA	9559	desktop-bnccria.local.
☆		DESKTOP-BNCCRIA (2)	49 91	localhost

☑ Use fixed port: 9559

☑ Use fixed IP/hostname: 127.0.0.1

Figure II-26webots Nao in choregraph

- U can see the robot in the robot view and his camera viewing in the video monitor

- Click wakeup icon and the robot is ready to be tested

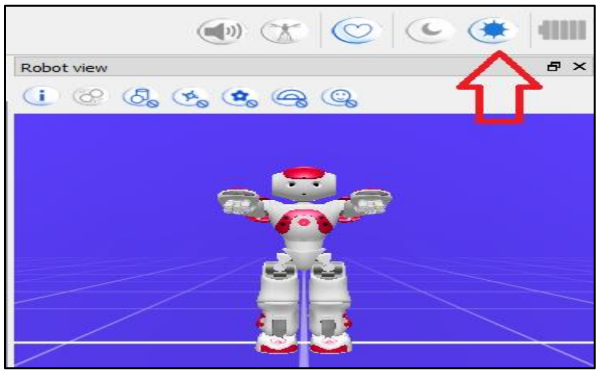


Figure II-27 wake up nao in webots

<u>Note</u>: for the real robot you will see a green robot in choregraph that is the robot

II.12.1.2 How to do a behavior in Choregraph

There is a box library contain a lot of actions and behaviors drag and drop the action in the flow diagram panel u can edit on that box on how u want the robot do that behavior connect the box on start of the flow diagram panel

By clicking twice on the box u can modify or do changes in the program written in python

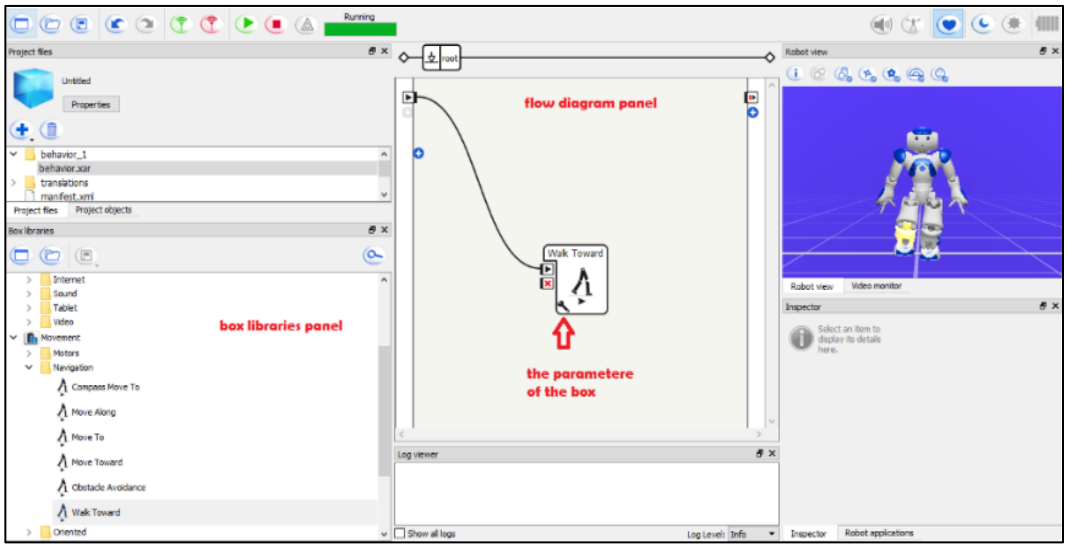
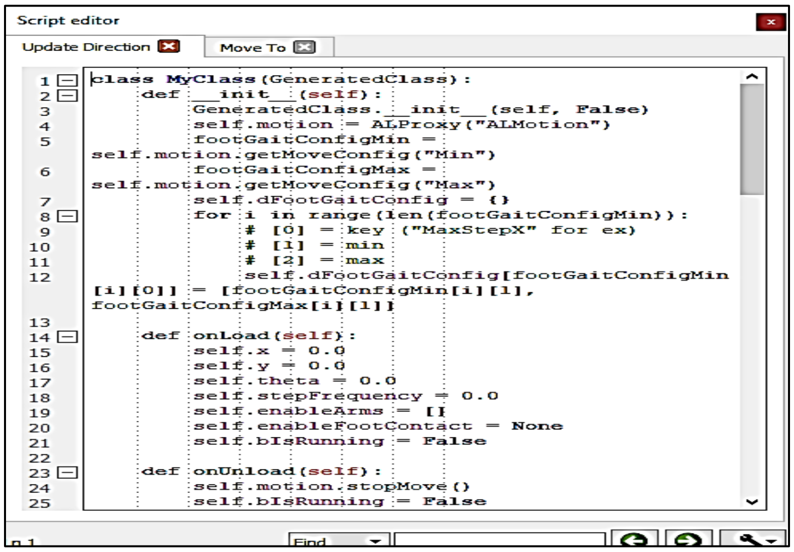


Figure II-28 box libraries in choregraph

- By clicking twice on the box u can modify or do change the program written in python



```python
class MyClass(GeneratedClass):
    def __init__(self):
        GeneratedClass.__init__(self, False)
        self.motion = ALProxy("ALMotion")
        footGaitConfigMin =
self.motion.getMoveConfig("Min")
        footGaitConfigMax =
self.motion.getMoveConfig("Max")
        self.dFootGaitConfig = {}
        for i in range(len(footGaitConfigMin)):
            # [0] = key ("MaxStepX" for ex)
            # [1] = min
            # [2] = max
            self.dFootGaitConfig[footGaitConfigMin
[i][0]] = [footGaitConfigMin[i][1],
footGaitConfigMax[i][1]]

    def onLoad(self):
        self.x = 0.0
        self.y = 0.0
        self.theta = 0.0
        self.stepFrequency = 0.0
        self.enableArms = []
        self.enableFootContact = None
        self.bIsRunning = False

    def onUnload(self):
        self.motion.stopMove()
        self.bIsRunning = False
```

Figure II-29 python code of the walking box

II.12.2 Programing Nao with python:

1. Import ALProxy
2. Create ALProxy to the module you want to use
3. Call the method and add the parameter

CHAPTER III: Development tools

In the last few years, humanoid robots have become a popular research tool To develop project with nao robot we utilize a couple of technologies and libraries in this chapters we will present them

IV. Technologies

IV.1 Neural network

Development of Neural Networks date back to the early 1940s. It experienced an upsurge in popularity in the late 1980s. This was a result of the discovery of new techniques and developments and general advances in computer hardware technology:

Inspired by the organic structure of human brain and its interconnections between neurons, artificial neural networks are linked in predefined directions of data diffusion, unlike the actual brain, where any neuron can bind to any other neuron. For example, you may take a picture, divide it into parts, and then enter it into the first layer of the neural network, which performs a certain task, and so on until we reach the last layer, the output layer, from which we obtain the final outputs [40].

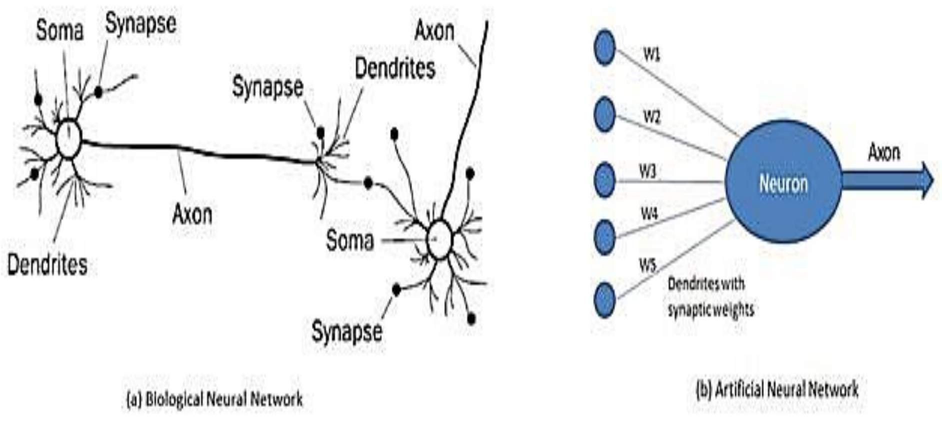

Soma
Synapse
Axon
Dendrites
Synapse
Axon
Dendrites
Soma
Synapse
(a) Biological Neural Network
W1
W2
W3
W4
W5
Neuron
Axon
Dendrites with synaptic weights
(b) Artificial Neural Network

Figure IV-1 biological neural network and artificial neural network [41]

Most NNs have some sort of “training" rule. In other words, NNs “learn" from examples (as children learn to recognize dogs from examples of dogs) and exhibit some capability for generalization beyond the training otherwise depending on the probability theory [40].

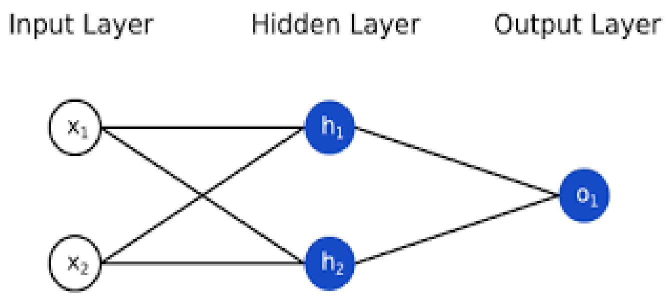


Figure IV-2 neural network simple model [40]

IV.1.1.1 Convolutional Neural Network (CNN)

CNN is mostly utilized in computer vision applications such as image classifications, face recognition, and object detection. For image classification, CNN, followed by neural layers and activation functions, analyzes an input picture as an array of pixels, processes it by looking for patterns in the image, and then classifies it into the problem statement's categories or classes [40].

IV.1.1.2 CNN Architecture

In deep learning, the CNN architecture will be composed of convolutional layers(filters), Pooling layers, fully connected layers and then output layers [40].

IV.1.1.2.1 Convolutional Layer

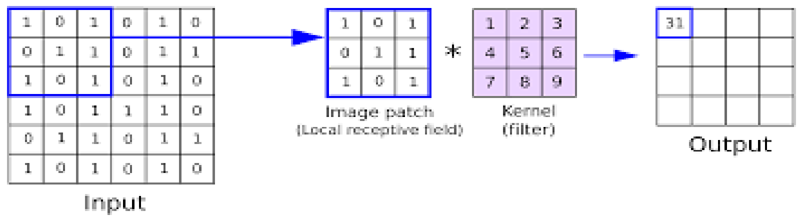


Figure IV-3 Convolutional Layer

The filter will be used to learn features from the picture by convolving it with the image, a mathematical operation, and then forming a new matrix with the learnt features such as edge detections, sharpening, or blurring the image.

IV.1.1.2.2 Pooling Layer

This layer's input would be used to execute a down sampling operation on the input picture, compressing it from a more detailed image to a less detailed image with more information. It can be of several kinds [42]:

- Max Pooling
- Average Pooling
- Sum Pooling

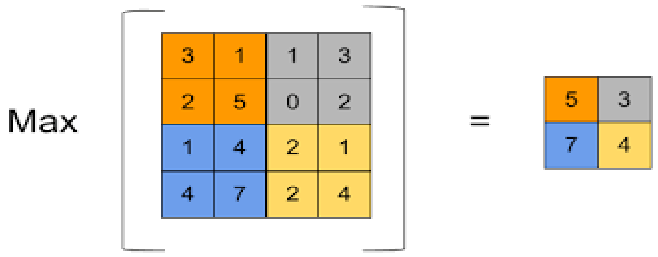


Figure IV-4 max pooling layer

IV.1.1.2.3 Fully Connected Layer

The matrix will be flattened before being sent to the Fully Connected Layer like a neural network, shortly after the CNN network and pooling. The output will then be classified using the softmax or sigmoid activation function, with the number of neurons in the final layer equaling the number of classes in the output [42].

IV.1.1.2.4 Activation Function

It is a nonlinear layer put at the end of or in between neural networks which then decide the output of that node. Available activation functions are: Sigmoid, ReLU, Leaky ReLU, Softmax etc [42].

IV.1.2 Training methods

IV.1.2.1 Supervised learning

Both the inputs and outputs are delivered during supervised training. After that, the network processes the inputs and compares the outputs to the desired outputs. The system then propagates the errors back through the system, causing the weights that regulate the network to be adjusted. As the weights are regularly adjusted, this process repeats itself. The "training set" is a collection of data that makes training possible. As the connection weights are improved during the training of a network[43].

IV.1.2.2 Unsupervised learning

In unsupervised training, the network is provided with inputs but not with desired outputs. The system must then pick which features will be used to organize the data input. Self-organization or adaptation are terms used to describe this process. Unsupervised learning is not well understood until our time [43].

IV.2 Deep learning technology:

Deep learning allows computational models with several processing layers to learn various degrees of abstraction for data representations. Deep learning uses the backpropagation technique to show how a machine should adjust its internal parameters that are used to calculate the representation in each layer from the representation in the previous layer, revealing detailed structure in enormous data sets. Deep convolutional nets have revolutionized image, video, voice, and audio processing [44].

an artificial neural network, signals travel between nodes and assign corresponding weights. A heavier weighted node will exert more effect on the next layer of nodes. The final layer compiles the weighted inputs to produce an output Because deep learning systems handle a vast quantity of data and perform multiple difficult mathematical computations, they demand powerful hardware. Even with such advanced hardware training computations can take a lot of time. Artificial neural networks categorize data based on responses to a series of binary true or false questions involving very complicated mathematical computations while processing the data, A facial recognition computer, for example, learns to identify and recognize faces' borders and lines, then more significant aspects of the faces, and eventually entire representations of faces. Over time, the program trains itself, and the probability of correct answers increases. In this case, the facial recognition program will accurately identify faces with time [45].

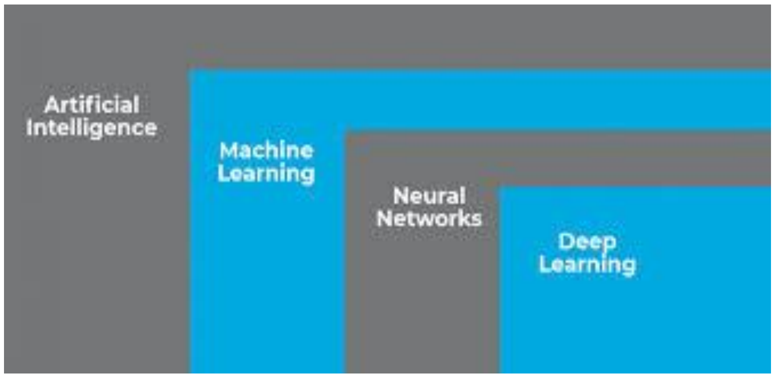


Figure IV-5 artificial intelligent -machine learning- neural network-deep learning

IV.3 Computer vision

IV.3.1 Definition:

The origins of computer vision go back to an MIT undergraduate summer project in 1966, it defined as a scientific field that process of extracting information from digital images. information obtained from a image from identification, navigational space measures, or augmented reality applications, or augmented reality applications. Which means understanding the content of images and use it for other applications. Computer vision brings together a large set of disciplines. Neuroscience help computer vision by first understanding human vision [46].

IV.3.2 Human vision and computer vision

through neurons specialized cells transmit the information coming from the eye that capture light coming through the iris to the brain (some specialized neurons fired only when the line was in a particular spot on the retina or if it had a certain orientation), A camera captures images in a similar way and transmit pixels to the computer. In this part, cameras farther away or with more precision. Second, the interpreting device has to process the information and extract meaning from it. The human brain solves this in multiple steps in different regions of the brain [46].

IV.3.3 Optical vision measuring machine (VMM)

it can measure linear size, such as point, line, distance, angle, circle, ellipse, rectangle, tangent and the geometrical size, such as straightness, flatness, circularity, cylindricity, perpendicularity, parallelism, angularity, position, concentricity, coaxiality, symmetry, line profile, surface profile [47].

IV.3.4 Visual semantic information:

Semantic vision seeks to understand what objects are present in an image an image has a lot of semantic information that leads the computer to recognize people, recognize actions, gestures, faces. Also, in medical filed to recognize cancer and other diseases…. [46].

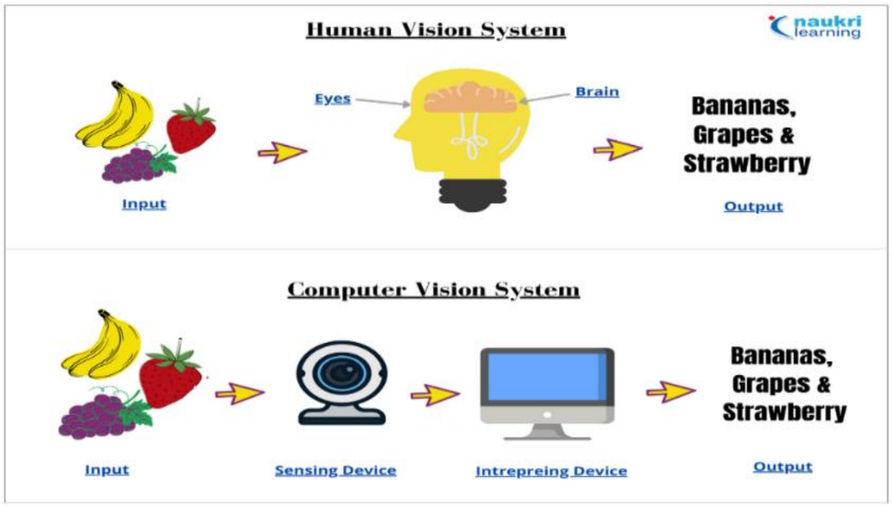


Figure IV-6 Human vision system [46]

IV.4 Color Representation

IV.4.1 RGB Color Mode:

The RGB space is the most widely used color model, in which colors are represented by their red, green, and blue components in an orthogonal Cartesian space. according to the human visual system There are three kinds of photoreceptors in the retina called cones, it’s difficult to tell what color is being represented by a certain red, green, or blue value because it is not uniform.

with given 2 RGB colors and applying the same variation to the 2 colors a different chromatic variation, Human vision is not the most comprehensive vision system in nature; other species have superior vision systems than humans in a variety of ways, but human vision has been the most researched and modeled for quite some time. With the goal of creating a color representation model that is as close to the human visual perception model as feasible [48].

IV.4.2 Thresholding HSV color space

HSV Color Scale: (which stands for Hue Saturation Value) Hue (H) refers to the perceived color, the saturation (S) component measures its dilution The third component is the intensity. The principal acquire device usually not use HSI model in the acquiring sensors. This means that a model conversion method must be computed to obtain an image in HSV color model. HSI model is an independent orthogonal model system in all cases excepts when intensity value is zero where the color black is defined. Black color is independent of hue and saturation values. Because it functions like a human visual perception system, the HSI model is a more intuitive color model system. The hue component defines the fundamental color, which is described as an angle that works from 0 to 360 degrees. Then the three complete color palettes (red or green or blue) can be defined with 120 degrees. Hue's other color is a lineal distribution that covers 360 degrees, and any color may be identified by an angle.

A normalized number from 0 to 1 is commonly used to denote the saturation component. When the saturation value is set to 0, the color is pure, and when the saturation value is set to 1, the color is saturated. The intensity value is the last model component. This component represents the quantity of light that falls on a color. In the HSI color model, this component is usually defined as a number between 0 and 1. Although some publications classify RGB and HSI models as not uniform color representation models [48].

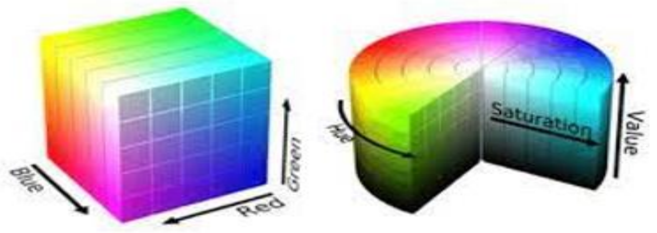


Figure IV-7 RGB and HSV color space [49]

IV.5 Image:

IV.5.1 Definition:

A three-dimensional array represents a standard digital image, such as a jpeg image. The x and y spatial axes correspond to the first two dimensions. Color information is stored in the third dimension in three chromatic channels: red, green, and blue., Color values typically range from 0 to 255 and are stored as unsigned integers (8 bits), which is a data type meant to retain exactly (255 = 28 − 1) For each There are 24 bits (3*8 bits) for each pixel, allowing for the storage of 16.7 million colors. Despite the abundance of color information, it's frequently easier to work with simply a gray-scale rendition of an image on a computer, lowering the information to 8 bits per pixel [49].

IV.5.2 Types of images:

IV.5.2.1 Binary Images

contain pixels that are either black (0) or white (1).

IV.5.2.2 Grayscale Images

have a wider range of intensity than black and white. Each pixel is a shade of gray with pixel values ranging between 0 (black) and 255 (white).

IV.5.2.3 Color Images

have multiple color channels; each color image can be represented in different color models (e.g., RGB, LAB, HSV). For example, an image in the RGB model consists of red, green, and blue channel. Each pixel in a channel has intensity values ranging from 0-255.

IV.6 Speech recognition:

Speech is a human communication process used to send messages and speech structured by several languages that contain letters to generate words and sentences, and can convert human voice to text file, and to enable the robot and a human being to interact on the basis of the same language, it is a trend to use speech recognition in artificial intelligence and machine learning filed.

IV.6.1 Principle of Speech recognition technology

the sound is some sort of waves compressed as MP3, WMV, and other popular formats To work with these types of files, they must first be converted to unpacked pure waveform files, such as Windows PCM files, or wav files. The bigger the number of points in each millisecond voice, the higher the sample rate. Voice can has multiple of channels. sound must usually be converted to a single channel before being processed voice recognition tasks.

Figure IV-8 sound wave

The waveform of the voice is separated into frames. Frames are stacked on top of one other.

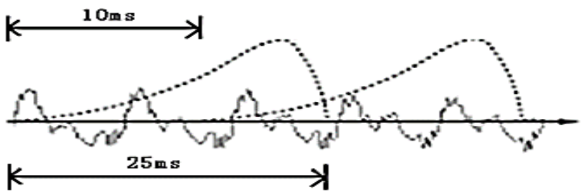


Figure IV-9 wave frame

There is an overlap between every two frames. As a result, this frame is said to be divided. However, because the waveform has absolutely no capacity to describe in the time domain, it must be transformed. With MFCC features Extracting method characteristics and converting each frame into a 12-dimensional vector, These 12 points were collected based on the physiological features of the human ear, implying that the frame's content information was incorporated. So far, the voice has been transformed into a 12-line, N-column matrix, where N is the total number of frames [50].

IV.6.2 Working of automatic speech recognition system (ASR):

IV.6.2.1 Data Collection

The first step is to collect the data that will be processed. The raw data (that is, the sentences that will be delivered by the speakers), The soundwaves of each sentence are then delivered by each speaker. These sound waves are then picked up by a microphone and turned into electrical signals, which are next translated into digital form using recording software and saved as a wave file [51].

IV.6.2.2 Pre-Processing:

The noise in the voice signal must be eliminated before passing it to the feature extraction block. This is accomplished by preprocessing. Based on the zero-crossing rate and energy, it eliminates the noise. The necessary information is contained in the output speech [51].

IV.6.2.3 Training:

Producing a pronunciation dictionary , contains words and mapping to their phonetic equivalents [51].

IV.6.2.4 Feature Extraction

extracts features from (.lab files) and represents them using the input signal's suitable data model. The suppression of unnecessary information from the recorded speech, such as information about the speaker (e.g. fundamental frequency) and information about the transmission channel (e.g. microphone characteristic), is a key property of feature extraction.

IV.6.2.5 Modelling

Modeling's goal is to create speaker-specific models from speaker-specific feature vectors. These models might be based on words or phonemes.

IV.6.2.6 Testing

The goal of testing is to discover the best match between the incoming acoustic model and the trained model. By integrating and maximizing the information communicated by the acoustic and linguistic models, a number of algorithms are available for matching and making real decisions .

IV.7 Hardware tools

IV.7.1 sonar in nao robot

Sonar: 2 emitters, 2 receivers.

Frequency: 40kHz.

Sensitivity: -86dB.

Resolution: 1cm.

Detection range: 0.25m - 2.55m.

Effective cone: 60°.

NAO is equipped with two ultrasonic sensors (or sonars) which allow it to estimate the distance to obstacles in its environment. The detection range goes from 25 cm to 255 cm, but under 25 cm there is no distance information, the robot only knows that an object is present. [24].

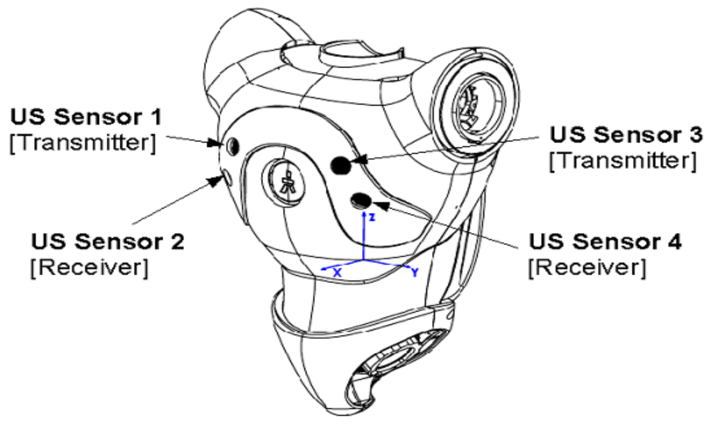


Figure IV-10 sonar position in nao robot [24]

IV.8 camera in nao robot:

IV.8.1 charge-coupled device (CCD)

A charge-coupled device (CCD) is a light-sensitive integrated circuit that captures images by converting photons to electrons. A CCD sensor breaks the image elements into pixels. Each pixel is converted into an electrical charge whose intensity is related to the intensity of light captured by that pixle.

IV.8.2 field of view (FOV)

FOV is the range of the observable world visible at any given time through the human eye, a camera viewfinder or on a display screen. It refers to the coverage of an entire area rather than a single, fixed focal point. FOV also describes the angle through which a person can see the visible world.

- The current Nao robot standard architecture contains two cameras centered at the head. the CCD model of two cameras are identical. These cameras are precisely positioned on the robot's face. Each camera has a 640x480 resolution at 30 frames per second, although none of them have any type of zoom. The diagonal angle of field of vision, commonly known as diagonal angle FOV, is the only intrinsic camera characteristic offered by the robot manufacturer [53].

IV.8.3 Camera calibration:

using a camera calibration procedure, the cameras' intrinsic parameters must be established. camera calibration usually is determining geometric elements like the projection center and picture plane. Prior to acquiring intrinsic parameters from a camera, the first step is to select the camera model that will be used in the calibration procedure.

Note: Nao camera use pinhole model [53].

IV.8.4 Pinhole Model

This camera model is the most used in X-ray cameras, scanned

photographic and it principally model CCD like sensors. If we define the center of projection as the origin of the Euclidean coordinate system of the projection plane that generate the image. We can consider that the central projection of points in space falls onto a plane.

as shown in the figure bellow image generation where a 3D point P is represented in a 2D plane as p. Note that the origin of the camera coordinate frame is located in the lens plane. Len center position is known as camera center. The distance between the lens and the image plane is called the effective focal length [53].

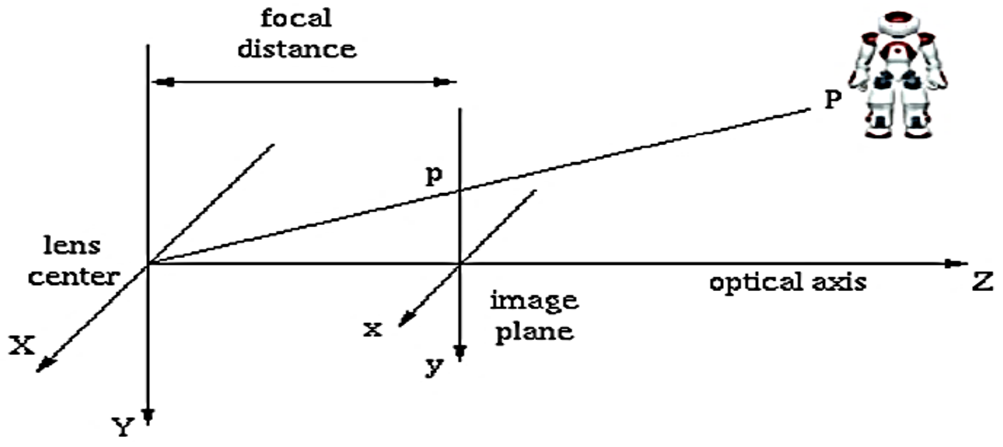


Figure IV-11 pinhole model in nao camera

Once pinhole camera model is selected as the mathematical model to characterize Nao's cameras. Camera parameters can be determined using 3 main techniques:

IV.8.5 Camera parameters

IV.8.5.1 Self-calibration

When you move a camera in a static scene, the rigidity of the scene supplies the camera's internal parameters. When the camera's internal characteristics can be changed, it's critical that these values remain consistent during the calibration process [53].

IV.8.5.2 Photogrammetric Calibration:

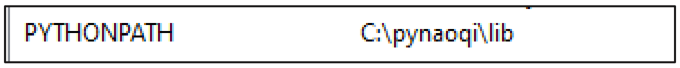
The camera views objects in three-dimensional space whose shape and position are known to a high degree of accuracy. Calibration objects are normally made up of two or three orthogonal planes, and setting up 3D space phase is challenging, and calibration instruments are often costly, When 3D space is clearly defined [53].

IV.9 Software tools

IV.9.1 pynaoqi:

(All about this library in chapter 2) is the naoqi python library and to integrate it in our work envairment we should folow this steps

1- Make sure to download Python 2.7
2- Install and extract the zip folder from the official web site https://developer.softbankrobotics.com/nao6/downloads/nao6-downloads-windows (extract it in your local C:/)
3- Add the PATH in your envairment variables as it shows in the figure



PYTHONPATH	C:\pynaoqi\lib

Figure IV-12 pynaoqi installation

IV.9.2 Naoqi module:

IV.9.2.1 The ALMotion module

provides methods which facilitate making the robot move.

It contains four major groups of methods for controlling the:

- joint stiffness (basically motor On-Off)

- joint position (interpolation, reactive control)
- walk (distance and velocity control, world position and so on)
- robot effector in the Cartesian space (inverse kinematics, whole body constraints)

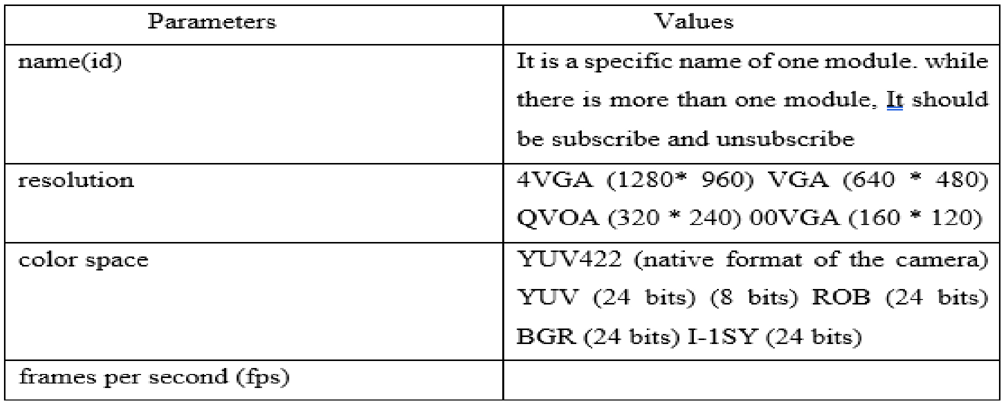
ALMotion runs at 50Hz (cycle of 20ms)

ALMotion provides methods that help make NAO move. It contains commands allowing you to manipulate joint stiffness, joint angles, and a higher level API allowing you to control the walk all this function in this link http://doc.aldebaran.com/1-14/naoqi/motion/almotion-api.html

IV.9.2.2 ALVideoDevice module

ALVideoDevice module is a portion of NAOqi vision API. The responsibility of ALVideoDevice is used to capture images from a video and supply images to be managed by other modules. The parameters of ALVideoDevice module are:

Table 1 CAMERA CALIBRATION

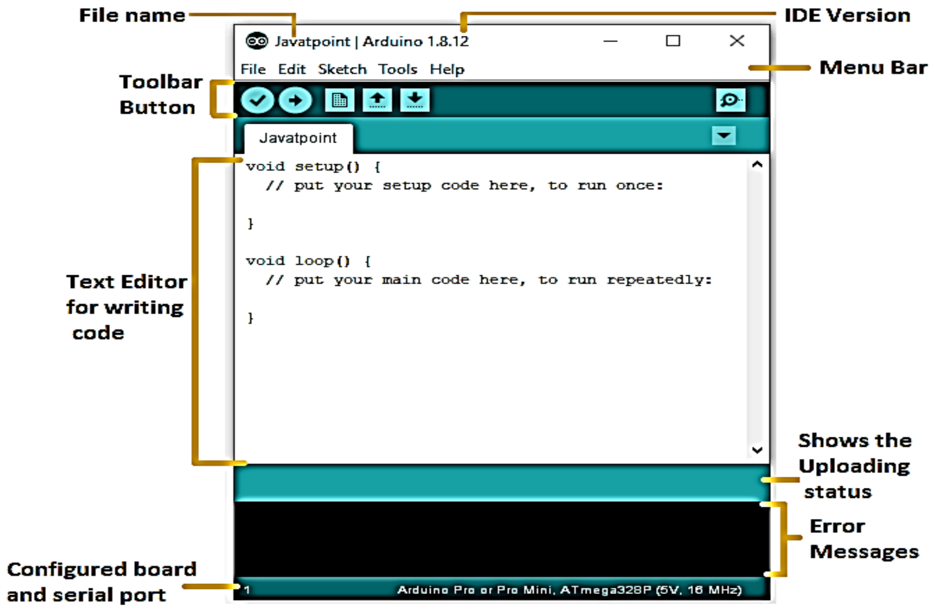


Parameters	Values
name(id)	It is a specific name of one module. while there is more than one module, It should be subscribe and unsubscribe
resolution	4VGA (1280* 960) VGA (640 * 480) QVOA (320 * 240) 00VGA (160 * 120)
color space	YUV422 (native format of the camera) YUV (24 bits) (8 bits) ROB (24 bits) BGR (24 bits) I-1SY (24 bits)
frames per second (fps)	

IV.9.3 Arduino IDE :

The Arduino IDE(IDE stands for Integrated Development Environment)is an open-source program for writing and uploading code to Arduino boards. The IDE program is compatible with a variety of operating systems, including Windows, Mac OS X, and Linux. C and C++ are supported programming languages.

Sketching refers to the process of writing a program or code in the Arduino IDE. To upload the sketch written in the Arduino IDE software, we must link the Genuino and Arduino boards to the IDE. The extension '.ino' is used to save the drawing [54].


File name
IDE Version
Javatpoint | Arduino 1.8.12
File Edit Sketch Tools Help
Menu Bar
Toolbar Button
Javatpoint
Text Editor for writing code
Shows the Uploading status
Error Messages
Configured board and serial port
Arduino Pro or Pro Mini, ATmega328P (5V, 16 MHz)

```
void setup() {
  // put your setup code here, to run once:

}

void loop() {
  // put your main code here, to run repeatedly:

}
```

Figure IV-13 Arduino IDE

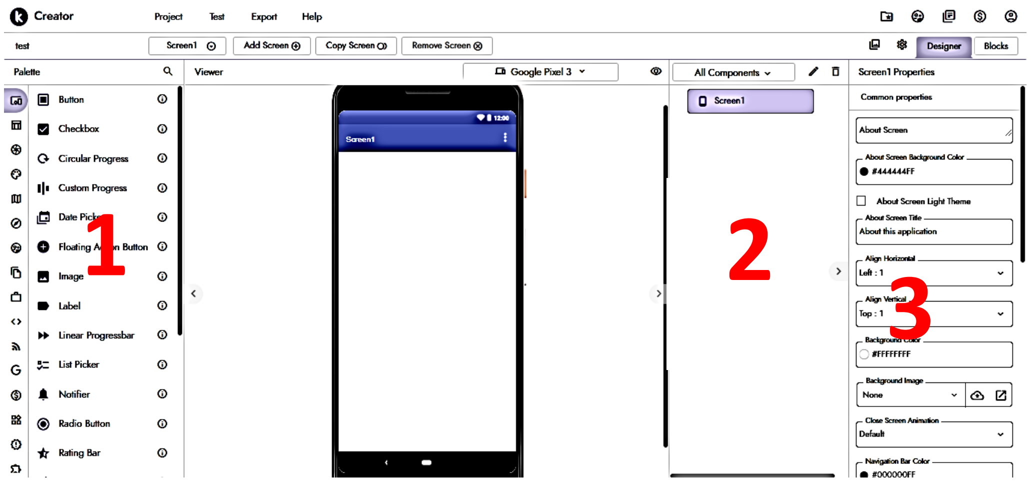
IV.9.4 Android Application Development Platforms

Platform as android studio require a knowlge of java script or kotlin programing languages also it take a lot of time to make a professional android application However, there are multiple platforms facilitate this process in this project we used kodular app creator platform .

IV.9.5 Kodular app creator:

Kodular is an online suite for mobile app development. It mainly provides a free drag-and-drop Android app creator without coding, based on MIT AppInventor. It brings lots of new features like new components and blocks. It also provides a free online app store to share and distribute apps and extensions IDE for advanced users. Furthermore, all of Kodular's components are free and able to create applications with modern layouts. This platform provides a direct app emulation on android phones, using their official application without needing to export and install the application of the created project.

IV.9.5.1 Kodular platform interface (designer part):



About Screen

About Screen Background Color: #444444FF

About Screen Light Theme

About Screen Title: About this application

Align Horizontal: Left : 1

Align Vertical: Top : 1

Background Color: #FFFFFFFF

Background Image: None

Close Screen Animation: Default

Navigation Bar Color: #000000FF

Figure IV-14 kodular interface

- Designer part

1. The pallet: where all the component needed for the project and contain 2sections

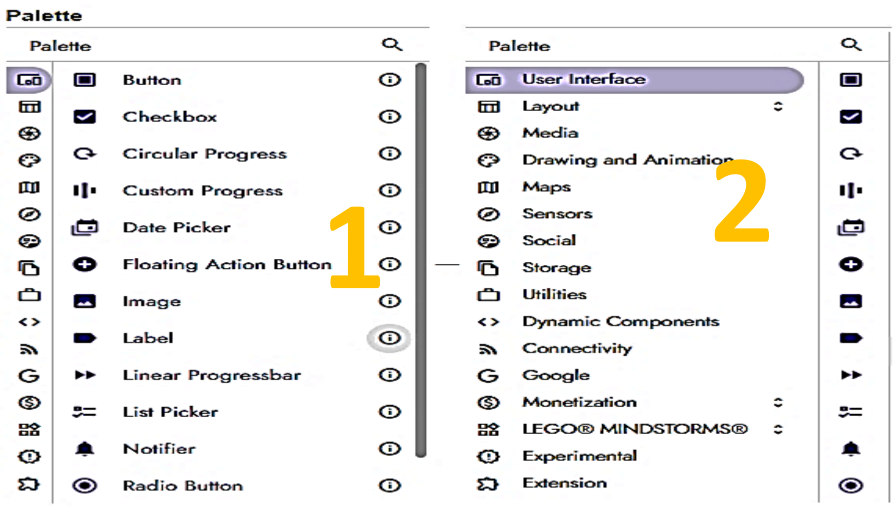


Figure IV-15the components in kodular

1- In this section Components are being categorized by type (for example, interface components, media components, connectivity…)
2- Here the available components for the specific type

Note: it uses drag and drop method

2. All components: here shows all the components used in the project and manage it (including extensions) Edit their names, Delete components…
3. In this section there is the component proprieties and external appearance of the it (edit the size, the color, add a picture…)

IV.9.5.2 Blocks (programing part)

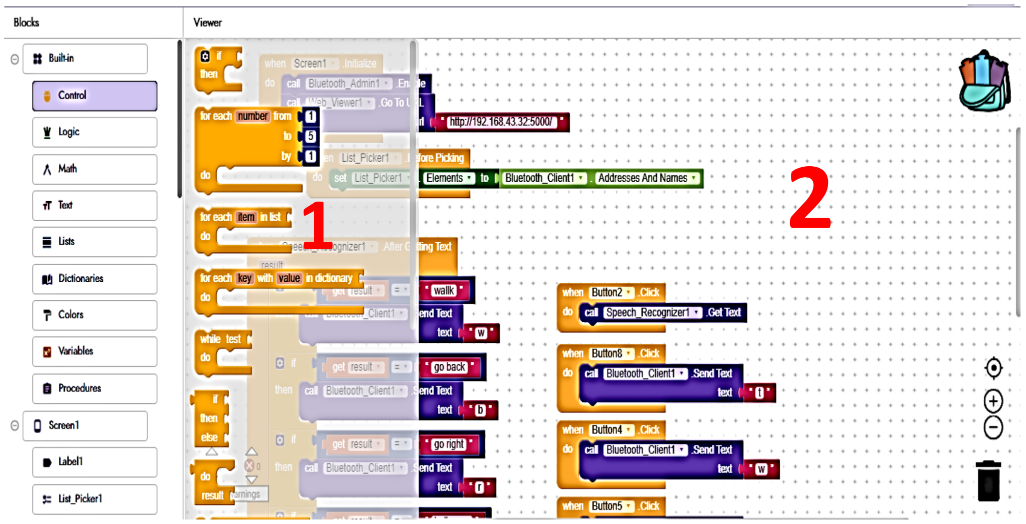


Figure IV-16 blocks in kodular

1- Blocks section contain the logic instruction and show all the methods for programing in addition to every component we choose in the design section have his own block of instruction and methods

- Blocks in orange color refers to evets
- Blocks in bleu are function
- Blocks in green are proprieties (There are 2 types, set and get propriety)
- Block pink are strings or texts

2- we can build our own application code using the selected blocks

- testing and export the project

1- testing:

The testing area allows us to test our app on our phone without having to install it. We may test it immediately by connecting the phone to our PC, or we can use the Kodular Companion app to scan the RQ code and view the built app right in front of us

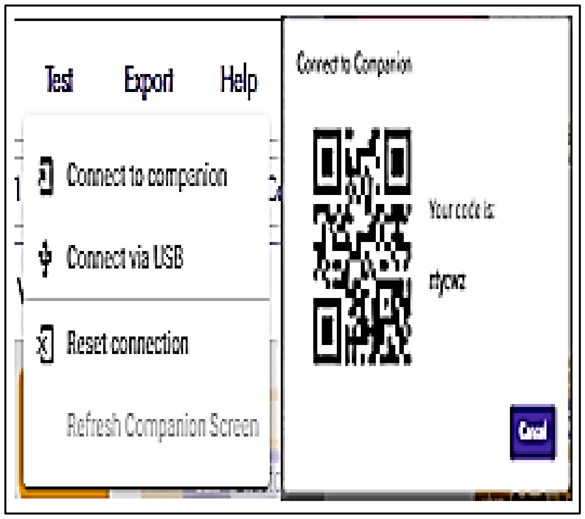


Figure IV-17 kodular testing the Project

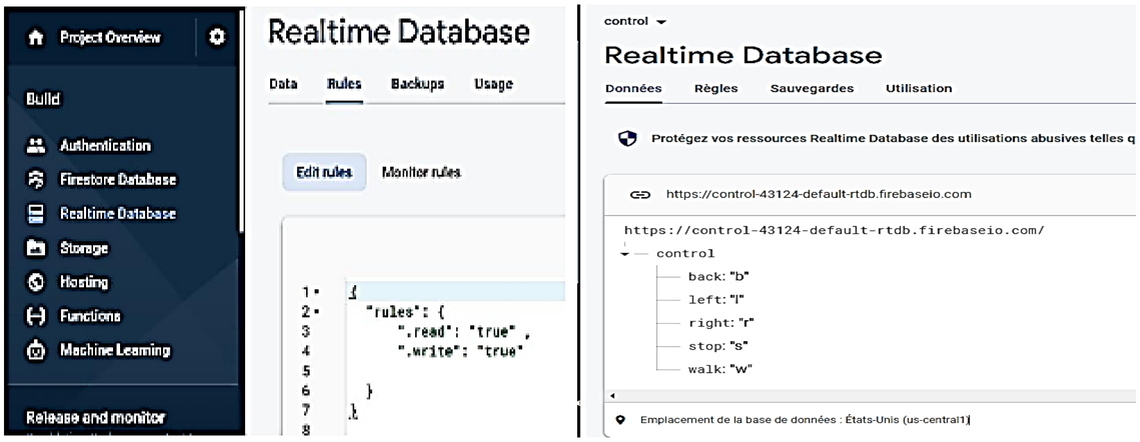
[illegible] nload th phone. We may scan the RQ Code, browse the provided URL, or click the Download APK

IV.9.6 Firebase Real-time Database

Firebase Google built the Firebase platform as a tool to support the creation of online applications. The platform offers nearly all of the web services necessary to build a fully working online application. The Firebase Real-Time Database is a service that is part of Firebase. It only allows us to store data there in JSON format, and we can only retrieve this data over the internet [55].

JavaScript Object Notation (JSON): "JSON is a text format that is entirely independent of programming language, although it employs standards that are well-known to programmers of the C-family of languages, including C, C++, C#, Java, JavaScript, Perl, Python, and many more. JSON is the best language for data exchange due of these characteristics [55].

Create a real time database: in order to begin Simply create a project in Firebase and select the Realtime Database option to use this service. We may modify the read/write criteria in the rules tab in order to interact with this database for security reasons. However, we may permit both read and write without any restrictions while testing.



```
{
  "rules": {
      ".read": "true" ,
      ".write": "true"

  }
}
```

Protégez vos ressources Realtime Database des utilisations abusives telles q

https://control-43124-default-rtdb.firebaseio.com

```
https://control-43124-default-rtdb.firebaseio.com/
  control
      back: "b"
      left: "l"
      right: "r"
      stop: "s"
      walk: "w"
```

Emplacement de la base de données : États-Unis (us-central1)

Figure IV-18 realtime firebase database

IV.9.7 Python programing language

Python is a general-purpose interpreted, interactive, object-oriented, and high-level programming language. It was created by Guido van Rossum during 1985- 1990

Python is a programming language that may be used in a variety of ways. Many of its features enable functional programming and aspect-oriented programming, as well as object-oriented programming and structured programming (including metaprogramming and metaobjects (magic methods) Many more paradigms, such as design by contract[64][65] and logic programming, are supported , Python manages memory through dynamic typing and a combine of reference counting and a cycle-detecting garbage collector [56].

- **Note:** Nao use only python2.7.x version

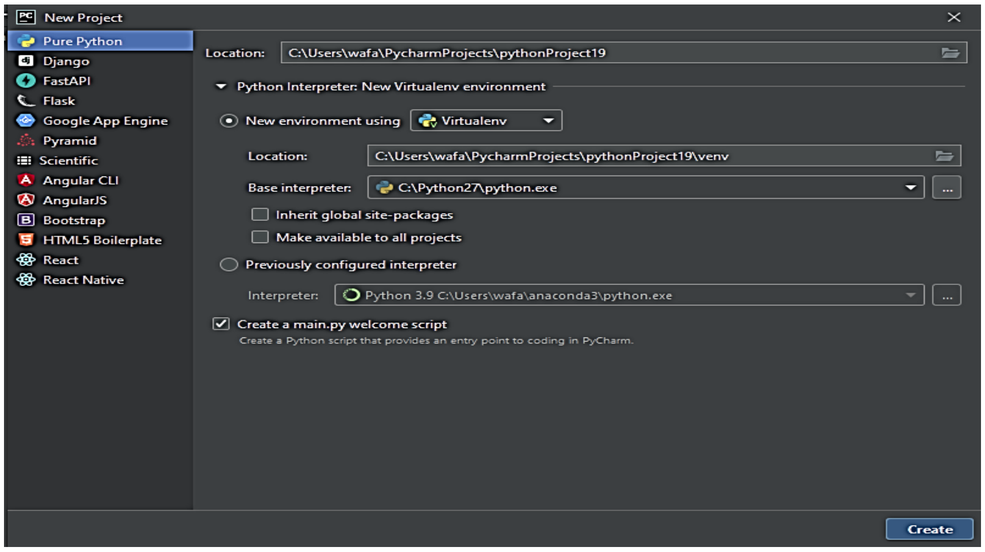
IV.9.8 pycharm IDE

PyCharm is an integrated development environment used in computer programming, specifically for the Python programming language. It is developed by the Czech company JetBrains. With PyCharm, you can access the command line, connect to a database, create a virtual environment, and manage your version control system.

IV.9.8.1 Writing a code in pycharm

After instaling pycharm and python

Pycharm can find the path for python automatcly



Location: C:\Users\wafa\PycharmProjects\pythonProject19

Python Interpreter: New Virtualenv environment

(●) New environment using Virtualenv

Location: C:\Users\wafa\PycharmProjects\pythonProject19\venv

Base interpreter: C:\Python27\python.exe

☐ Inherit global site-packages

☐ Make available to all projects

( ) Previously configured interpreter

Interpreter: Python 3.9 C:\Users\wafa\anaconda3\python.exe

☑ Create a main.py welcome script

Create a Python script that provides an entry point to coding in PyCharm.

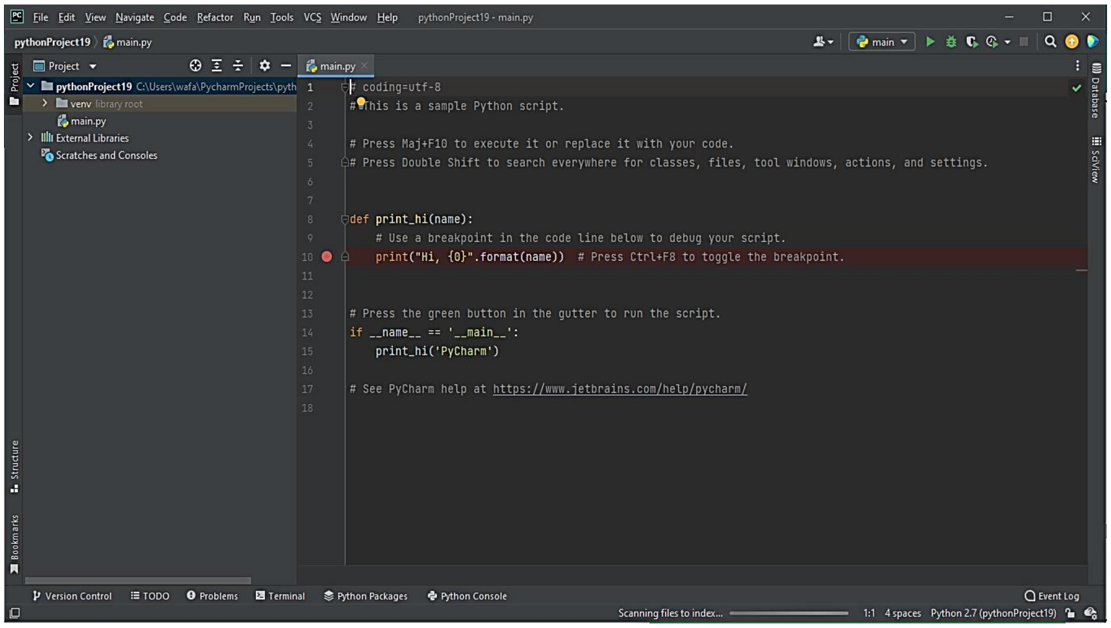
Figure IV-19 project configuration in pycharm

Specify the project location and expand the Project Interpreter drop down. Here, you have options to create a new project interpreter or reuse an existing one. Choose New environment using. Right next to it, you have a drop down list to select one of Virtualenv, Pipenv, or Conda, which are the tools that help to keep dependencies required by different projects separate by creating isolated Python environments for them. you can specify the environment location and choose the base interpreter from the list, which is a list of Python interpreters (such as Python2.7 and Python3.6) installed on your system. Then you have to select boxes to inherit global site-packages to your new environment and make it available to all other projects. Leave them unselected.

Click Create on the bottom right and you will see the new project created.

IV.9.8.2 Start coding in pycharm

PyCharm provides Intelligent Coding Assistance with code completion, code inspections, on-the-fly error highlighting, and quick-fix suggestions. In particular, , PyCharm auto-completed the instrction and coloring it.



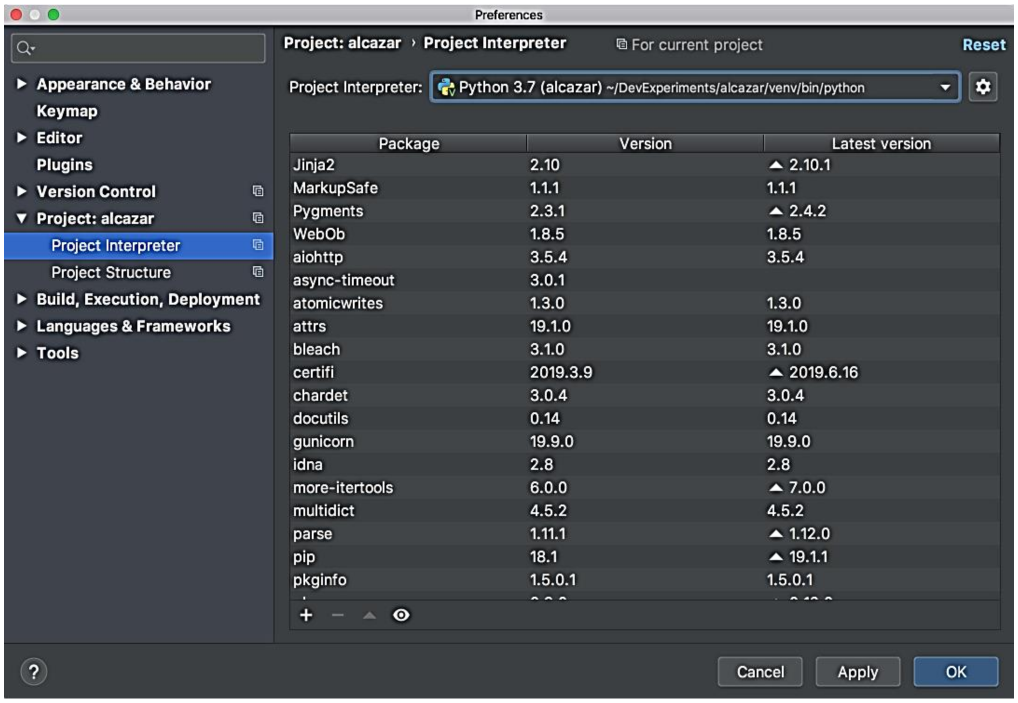
```
# coding=utf-8
#This is a sample Python script.

# Press Maj+F10 to execute it or replace it with your code.
# Press Double Shift to search everywhere for classes, files, tool windows, actions, and settings.


def print_hi(name):
    # Use a breakpoint in the code line below to debug your script.
    print("Hi, {0}".format(name))  # Press Ctrl+F8 to toggle the breakpoint.


# Press the green button in the gutter to run the script.
if __name__ == '__main__':
    print_hi('PyCharm')

# See PyCharm help at https://www.jetbrains.com/help/pycharm/
```

File Edit View Navigate Code Refactor Run Tools VCS Window Help pythonProject19 - main.py
pythonProject19 main.py
main
Project
pythonProject19 C:\Users\wafa\PycharmProjects\pyth
venv library root
main.py
External Libraries
Scratches and Consoles
Structure
Bookmarks
Database
SciView
Version Control
TODO
Problems
Terminal
Python Packages
Python Console
Event Log
Scanning files to index...
1:1 4 spaces Python 2.7 (pythonProject19)

Figure IV-20 pycharm interface

IV.9.8.3 Editing the project interpter

In file – setting -- project – project interpter u can see all the libraries that you use and version of python also you can install the libraries by clicking the add button



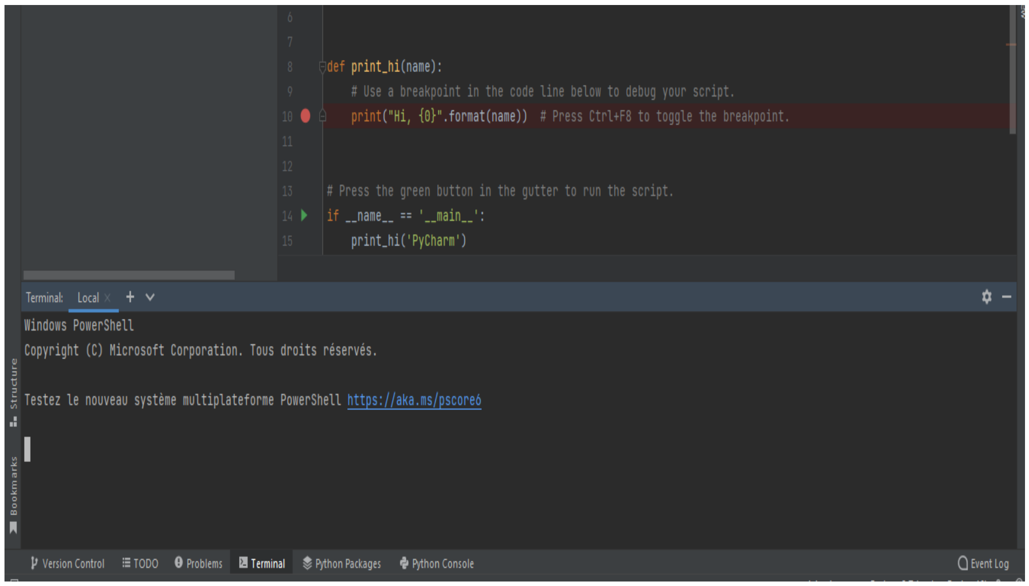
Project: alcazar › Project Interpreter

Project Interpreter: Python 3.7 (alcazar) ~/DevExperiments/alcazar/venv/bin/python

Package	Version	Latest version
Jinja2	2.10	▲ 2.10.1
MarkupSafe	1.1.1	1.1.1
Pygments	2.3.1	▲ 2.4.2
WebOb	1.8.5	1.8.5
aiohttp	3.5.4	3.5.4
async-timeout	3.0.1	
atomicwrites	1.3.0	1.3.0
attrs	19.1.0	19.1.0
bleach	3.1.0	3.1.0
certifi	2019.3.9	▲ 2019.6.16
chardet	3.0.4	3.0.4
docutils	0.14	0.14
gunicorn	19.9.0	19.9.0
idna	2.8	2.8
more-itertools	6.0.0	▲ 7.0.0
multidict	4.5.2	4.5.2
parse	1.11.1	▲ 1.12.0
pip	18.1	▲ 19.1.1
pkginfo	1.5.0.1	1.5.0.1

Figure IV-21 Project interrupter in pycharm

IV.9.8.4 Terminal in pycharm



```
def print_hi(name):
    # Use a breakpoint in the code line below to debug your script.
    print("Hi, {0}".format(name))  # Press Ctrl+F8 to toggle the breakpoint.


# Press the green button in the gutter to run the script.
if __name__ == '__main__':
    print_hi('PyCharm')
```

```
Windows PowerShell
Copyright (C) Microsoft Corporation. Tous droits réservés.

Testez le nouveau système multiplateforme PowerShell https://aka.ms/pscore6
```

Figure IV-22 terminal in pycharm

PyCharm includes an embedded terminal emulator for working with command-line shell from inside the IDE. set file permissions, and perform other command-line tasks without switching to a dedicated terminal application.,the terminal emulator runs with the default system shell, but it supports many other shells, such as Windows PowerShell, Command Prompt cmd. The terminal has the path of your project you can install libraries for the project and also run a python file in that project and so many thing…

IV.10 Python libraries:

IV.10.1 Opencv library

OpenCV OpenCV is open source computer vision library, based on BSD protocol and designed for computational efficiency and with a strong focus on real-time applications. It is utilized for doing some research and business. OpenCV has theprogramming languages interfaces, which allows users to connect with C++, C, Python and Java,

OpenCV supports Linux, Windows, Mac OS, IOS and Android systems it takes full advantages of multi-processor and OpenSL interface,There are over 500 algorithms and about ten times as many functions that compose or support those algorithms. It is usually implemented on machine learning and network maps.

The library of OpenCV has more than 2500 optimized algorithms, which includes a comprehensive set of both classic and state-of-the-art computer vision and machine learning algorithms. These algorithms can be used to detect and recognize faces, identify objects, classify human actions in videos, track camera movements, track moving objects, extract 3D models of objects, produce 3D point clouds from stereo cameras, stitch images together to produce a high resolution image of an entire scene, find similar images from an image database, remove red eyes from images taken using flash, follow eye movements, recognize scenery and establish
markers to overlay it with augmented reality, etc.

- To install it go to pycharm terminal and make sure to install this exact version (figure) because it's the only version is suitable with python2.7 and have all the missing dll files

```
pip install opencv-
```

- Python code of opencv read and show image

```
# import the cv2 library
import cv2
# The function cv2.imread() is used to read an image.
img_grayscale = cv2.imread('test.jpg',0)
# The function cv2.imshow() is used to display an image in
a window.
cv2.imshow('graycsale image',img_grayscale)
# waitKey() waits for a key press to close the window and 0
specifies indefinite loop
cv2.waitKey(0)
# cv2.destroyAllWindows() simply destroys all the windows we
created.
cv2.destroyAllWindows()
```

IV.10.1.1 Python example convert to HSV color

```
import cv2
img = cv2.imread("D:/ball.png")
img = cv2.cvtColor(img,
cv2.COLOR_BGR2HSV)
cv2.imshow('hsv', img)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

- The result:

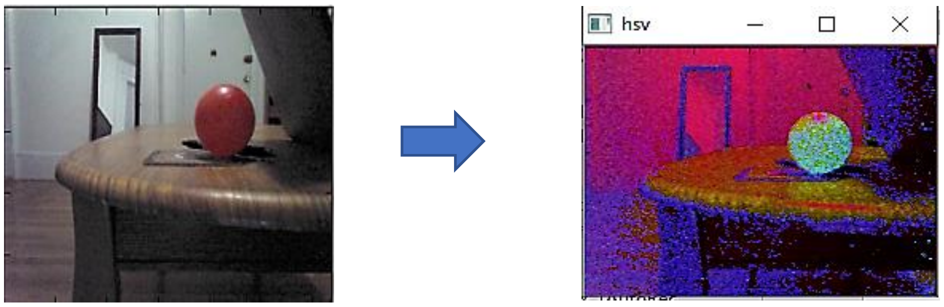
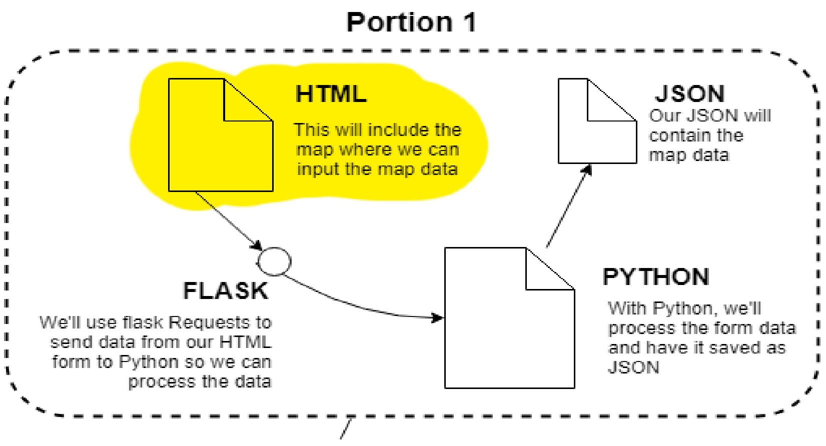


Figure IV-23 example HSV color

IV.10.2 Flask library

Flask is a web framework, it's a Python module that lets you develop web applications. It does have many features like URL routing, template engine and suitable for projects where requirements change dynamically.

- **Templates** are files that include both static and dynamic data placeholders. To create a finished document, a template is displayed with precise data. The Jinja


Portion 1
HTML
This will include the map where we can input the map data
JSON
Our JSON will contain the map data
FLASK
We'll use flask Requests to send data from our HTML form to Python so we can process the data
PYTHON
With Python, we'll process the form data and have it saved as JSON

template; Templates will be used in your application to render HTML that will be shown in the user's browser [57]

Figure IV-24 flask Diagram [58]

- To create a flask project: in pycharm go to file > new project > choose flask
- Add html file in the template folder

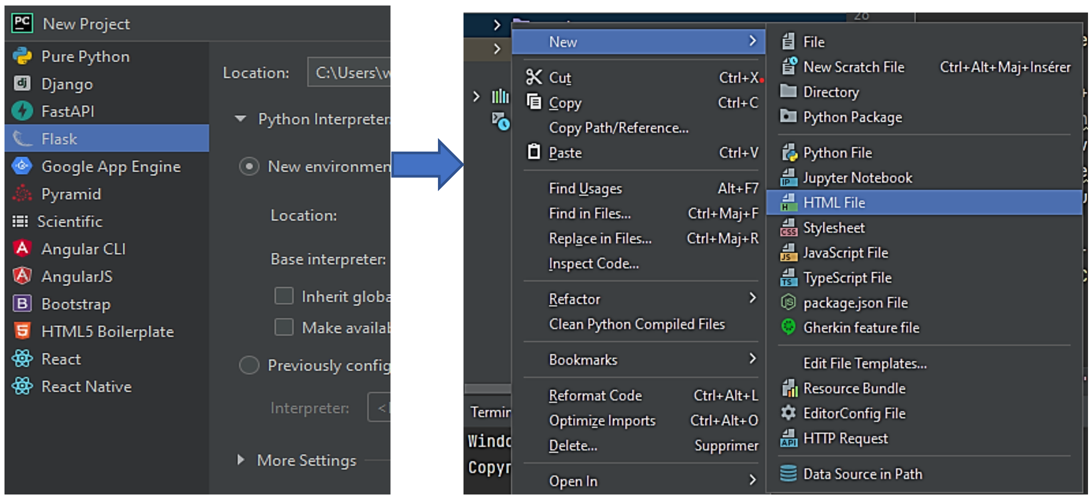


Figure IV-25 creating flask project in pycharm

IV.10.3 Serial library

This module encapsulates the access for the serial port. It provides backends for Python

- In pycham terminal pip install pyserial
- Make sure to choose the right port and baud rate as same as the Arduino serial port

```
ser = serial.Serial('COM5', 9600, timeout=1)
```

IV.10.4 pillow library

The Python Imaging Library adds image processing capabilities to Python interpreter. This library provides extensive file format support, an efficient internal representation, and fairly powerful image processing capabilities. The core image library is designed for fast access to data stored in a few basic pixel formats.

IV.10.5 matplotlib library

Matplotlib is a Python toolkit for data visualization and graphical charting that provides an open-source alternative to MATLAB. Developers can also use matplotlib's APIs to embed plots in GUI applications [59]

IV.10.5.1 General Concepts for matplotlib library

- Figure: Figure is the canvas on which different plots are created. The Matplotlib Figure is a canvas that contains one or more than one axes/plots.
- Axis: Axis in a Matplotlib figure is used for generating the graph limit and it is basically like a number line. There can be an X axis, Y axis and Z axis for plots up to 3 dimensional.
- Axes: Axes is generally considered as a plot. In the case of 3D, there are three-axis objects but there are many axes in a figure.
- Artist: Everything which can be seen on a figure is an Artist. Most of the artists are tied with the axes. These are like text objects, collection objects and Line2D objects, and many more.

IV.10.6 TensorFlow library

It was originally released in late 2015, and the first stable version in 2017. It's free and open source library from google . It's a symbolic math toolkit that employs dataflow and differentiable programming to handle a variety of tasks related to deep neural network training and inference, By receiving inputs as a multi-dimensional array called Tensor, to specify how data goes through a graph. It enables you to create a flowchart of operations that may be done on these inputs, which goes on one end and returns as output on the other.. It enables programmers to design machine learning applications utilizing a variety of tools, frameworks, and open-source resources.

Google's TensorFlow is now the most well-known deep learning package on the planet. used its products to enhance search, translation, picture captioning, and recommendations, TensorFlow is Set of Functions' that determines the output based on previous learning or prior knowledge. For example in recognition we will have dataset of images We'll use a subset of the images for training and the rest to test the neural model, The dataset consists of a zipped file including all of the images in the dataset [60] [61]

IV.10.7 Keras library:

Keras Keras is a Python interface for artificial neural networks it is an open-source software package. Keras is the TensorFlow library's interface

IV.10.8 TensorFlow with Keras training model:

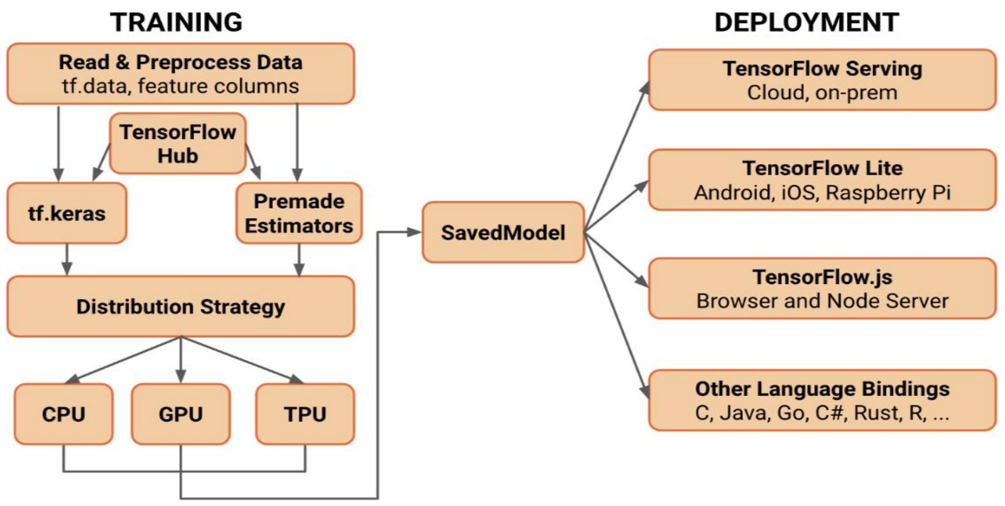


Figure IV-26 training module with Keras [62]

1- Load your data using tf.data. Training data is read using input pipelines which are created using tf.data.
2- Build, train and validate your model with tf.keras, or Premade Estimators. Keras integrates tightly with the rest of TensorFlow so you can access TensorFlow’s features, you can utilize TensorFlow Hub modules to train a Keras or Estimator model using transfer learning if you don't want to train a model from scratch.

3- Utilize eager execution to run and debug, then use tf.function for graph advantages. This function annotation converts your Python routines into TensorFlow graphs invisibly...
4- SavedModel best way is to save it in pickle file

IV.10.9 Socket library

Sockets and the socket API are used to send messages across a network. They provide a form of inter-process communication (IPC) it uses TCP/IP protocol

CHAPTER IV: APPLICATIONS AND IMPLEMENTATION OF THE ROBOT

V. The work we applied on nao robot

Nao robot have many abilities and sensor and make it a perfect platform to do a lot of research and application so in this part we tried to apply a couple of ideas and project and algorithm in order to make it help people with special needs and make their daily life much easier.

V.1 Nao robot Android application

It is important for the robot to have a control device that transmits and sends information, commands and various instructions from a distance to facilitate and simplify dealing with it.

In this part we going to make an android application to control the robot

- The application:
  In this part we use kodular creator to develop the application
  We use Bluetooth HC-06 as server
  Kodular app:
  The used component:
  Button: visible component with the ability to detect clicks. Many aspects of its appearance can be changed, as well as whether it is clickable (Enabled), can be changed in the Designer or in the Blocks Editor
  List picker: A button that, when clicked on, displays a list of texts for the user to choose among. The texts can be specified through the Designer or Blocks Editor
  Bluetooth client: A non-visible component that acts as a Bluetooth client.
  Speech Recognizer: A non-visible component that converts spoken words to text using voice recognition (kodular use extension of google speech recognition)
  Fire database: Fire database component
  Web viewer: Component for viewing Web pages. The Home URL can be specified in the Designer or in the Blocks Editor. The view can be set to follow links when they are tapped, and users can fill in Web forms. Warning: This is not a full browser. For example, pressing the phone's hardware Back key will exit the app, rather than move back in the browser history

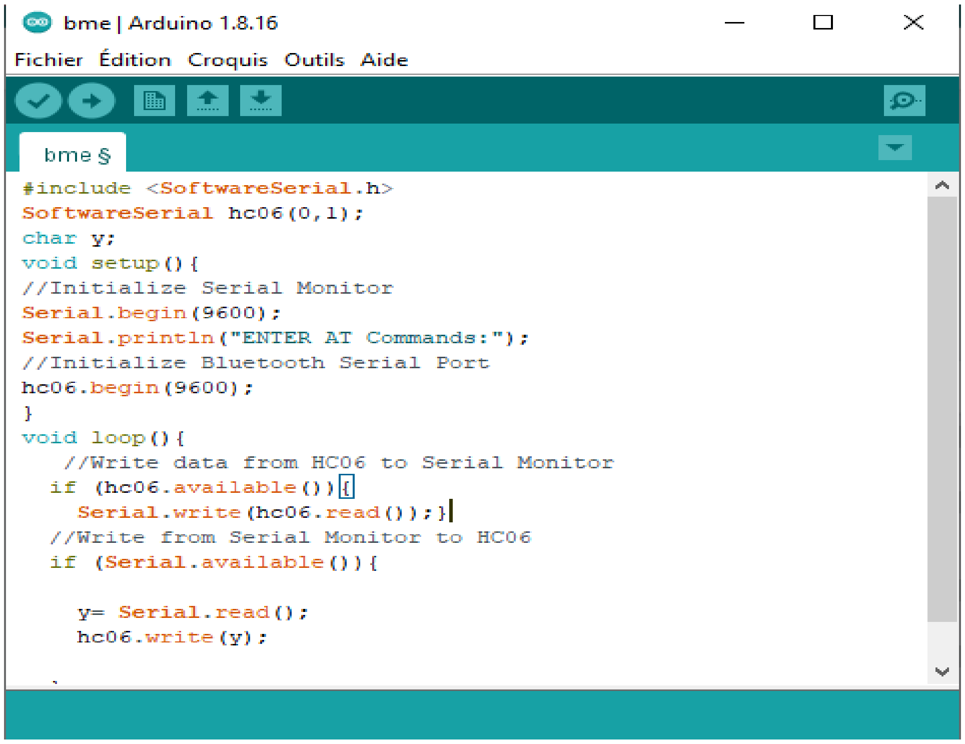
V.1.1 Connection between nao robot and the phone

V.1.2 Hardware solution

To make a connection between the phone and nao robot we use a hc-06 Bluetooth connected to an Arduino work as server for the data from the and send it back to nao robot to do that follow this steps

1- Connecting the phone to HC-06 Arduino Bluetooth

- In the Arduino:



```
#include <SoftwareSerial.h>
SoftwareSerial hc06(0,1);
char y;
void setup(){
//Initialize Serial Monitor
Serial.begin(9600);
Serial.println("ENTER AT Commands:");
//Initialize Bluetooth Serial Port
hc06.begin(9600);
}
void loop(){
   //Write data from HC06 to Serial Monitor
  if (hc06.available()){
    Serial.write(hc06.read());}
  //Write from Serial Monitor to HC06
  if (Serial.available()){

    y= Serial.read();
    hc06.write(y);
```

Figure V-1 Bluetooth Arduino code

- Receive the data in python environment using pyserial library

```
ser = serial.Serial('COM5', 9600, timeout=1)
```

- Than In kodular app

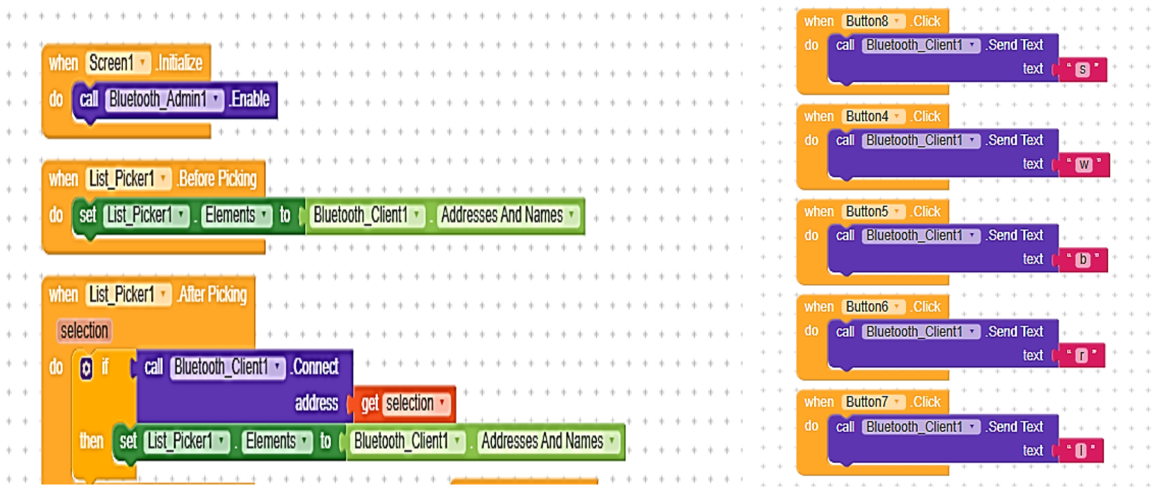
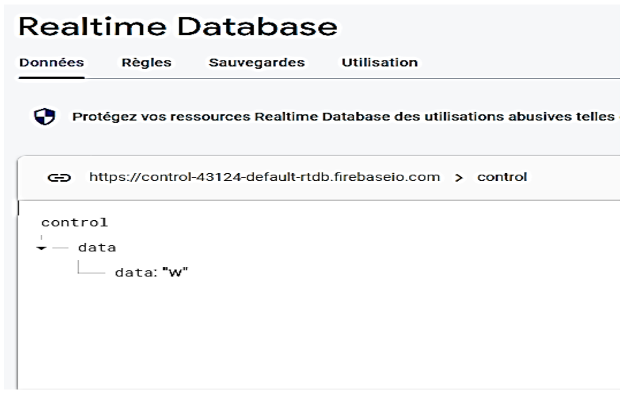

when Screen1 .Initialize
do call Bluetooth_Admin1 .Enable
when List_Picker1 .Before Picking
do set List_Picker1 . Elements to Bluetooth_Client1 . Addresses And Names
when List_Picker1 .After Picking
selection
do if call Bluetooth_Client1 .Connect
address get selection
then set List_Picker1 . Elements to Bluetooth_Client1 . Addresses And Names
when Button8 .Click
do call Bluetooth_Client1 .Send Text
text "s"
when Button4 .Click
do call Bluetooth_Client1 .Send Text
text "w"
when Button5 .Click
do call Bluetooth_Client1 .Send Text
text "b"
when Button6 .Click
do call Bluetooth_Client1 .Send Text
text "r"
when Button7 .Click
do call Bluetooth_Client1 .Send Text
text "l"

Figure V-2 code block for Bluetooth connection in kodular the control button

V.1.2.1 Software solution

Using the firebase Realtime database to store the data then read it from python and reverse it to the nao robot the data will change every time we will enter a new command this example when we click on walk button

Note: you must have internet connection


Realtime Database

Données Règles Sauvegardes Utilisation

Protégez vos ressources Realtime Database des utilisations abusives telles

https://control-43124-default-rtdb.firebaseio.com > control

```
control
└─ data
   └── data: "w"
```

Figure V-3 realtime database

- Python reding from fire base code:

```python
from firebase import firebase
firebase = firebase.FirebaseApplication('URL of database', None)
result = firebase.get('control', '')
```

```
4 print(result)
```

V.1.3 control button

Adding the button in kodular calling the Bluetooth to send data to the Arduino

Which in turn send this data to the robot

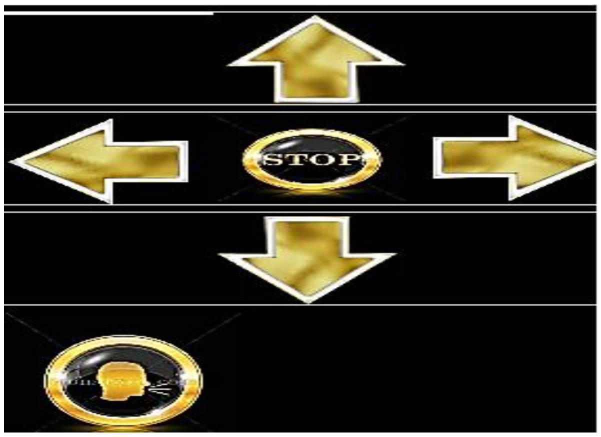


Figure V-4 control button

- python code to control nao with app button

```
if data == 'd':
    motion.post.moveTo(1, 0.0, 0.0)
if line == 't':
    motion.stopMove()
if data == 'o':
    motion.post.moveTo(0, 1, 0)
if data == 'f':
    motion.post.moveTo(-1, 0, 0)
if data == 'b':
    motion.post.moveTo(0, -1, 0)
```

V.1.4 adding speech recognition feature

adding speech recognition ability to the application give more facility to the user to control the robot specially for disable people who can’t move their hand or finding difficulty to use the phone properly

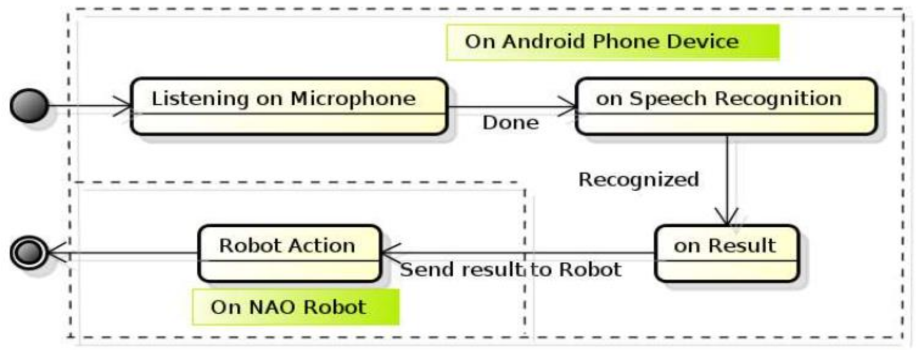
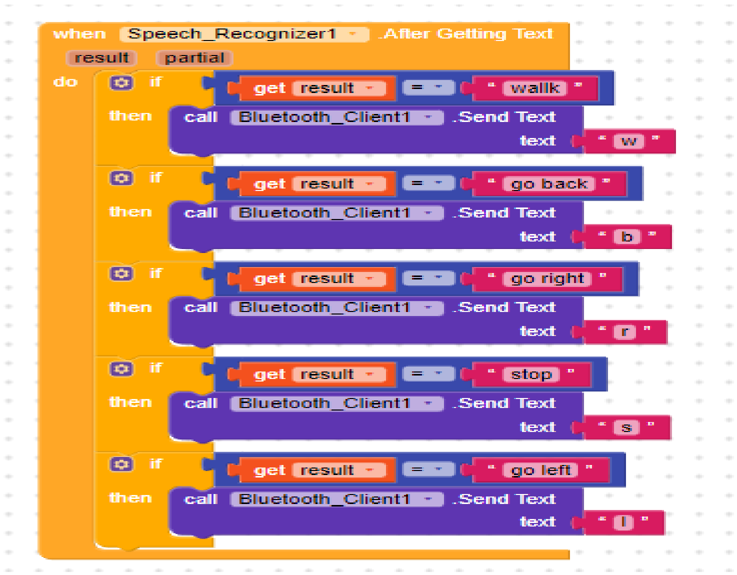


Figure V-5 speech recognition to control nao



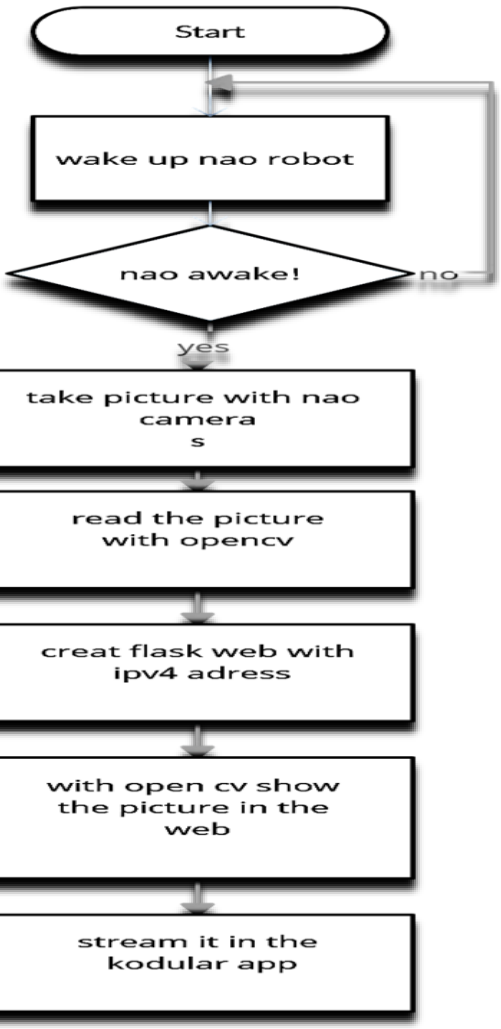
```
when Speech_Recognizer1 .After Getting Text
  result  partial
do  if   get result = "wallk"
    then call Bluetooth_Client1 .Send Text
                               text "w"
    if   get result = "go back"
    then call Bluetooth_Client1 .Send Text
                               text "b"
    if   get result = "go right"
    then call Bluetooth_Client1 .Send Text
                               text "r"
    if   get result = "stop"
    then call Bluetooth_Client1 .Send Text
                               text "s"
    if   get result = "go left"
    then call Bluetooth_Client1 .Send Text
                               text "l"
```

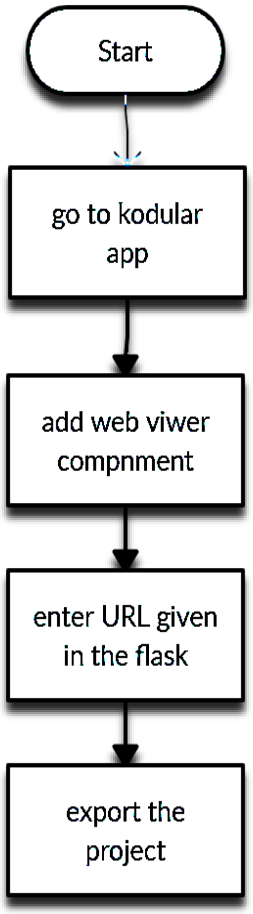
Figure V-6 speech recognition code block

V.1.5 Nao stream his vision in the app

It’s important for the user to see the road and vision of the robot in case it was far from the person vision or when he want to control it from far so steaming it vision on the phone give the access to it camera and give a better control to the robot

V.1.5.1 The plan:


Start
wake up nao robot
nao awake!
no
yes
take picture with nao camera s
read the picture with opencv
creat flask web with ipv4 adress
with open cv show the picture in the web
stream it in the kodular app



V.1.5.2 take picture from nao robot:

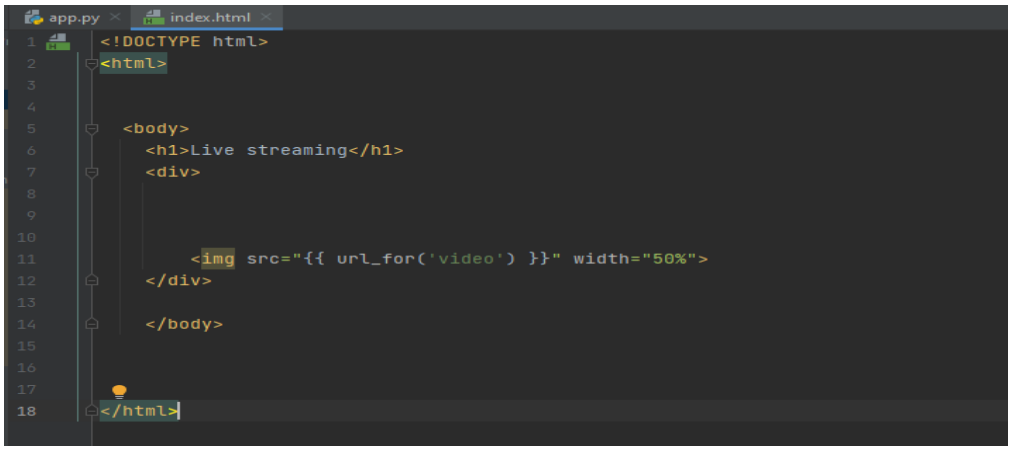
using ALVideoDevice module here is the python code to take picture with upper camera and save it in direction

```python
from naoqi import ALProxy
IP="desktop-6363djg.local."
from PIL import Image
PORT=9559
path = 'D:/' +  '/'
cam = ALProxy("ALVideoDevice", IP, PORT)
resolution = 2 # VGA (640*480px)
```

```
colorSpace = 11 # RGB

def pic(_name, CameraIndex):
    videoClient = cam.subscribeCamera("python_client", CameraIndex,
resolution, colorSpace, 5)
    naoImage = cam.getImageRemote(videoClient)
    cam.unsubscribe(videoClient)
    # Get the image size and pixel array.
    imageWidth = naoImage[0]
    imageHeight = naoImage[1]
    array = naoImage[6]
    im = Image.frombytes("RGB", (imageWidth, imageHeight),
str(array))
    im.save(_name, "PNG")
if __name__ == "__main__":
    pic(path + 'f3.png',0)
```

V.1.5.3 Create a web server to show the streaming

- In flask project go to template then add html file
- Create html web



```
<!DOCTYPE html>
<html>


  <body>
    <h1>Live streaming</h1>
    <div>



        <img src="{{ url_for('video') }}" width="50%">
    </div>

    </body>


</html>
```

Figure V-7 create web page with html

- Write your flask project Stream the picture given from nao robot in the web

V.1.5.4 Full source code for app streaming

```
#!/usr/bin/python3
""" Demonstrating APScheduler feature for small Flask App. """
```

```
from flask import Flask, render_template, Response
import cv2
#flask run -h 192.168.1.33

app = Flask(__name__)
import serial
import time
from naoqi import ALProxy
IP="169.254.232.44"
motion = ALProxy("ALMotion", IP,9559)
tts = ALProxy("ALTextToSpeech", IP,9559)
# make sure the 'COM#' is set according the Windows Device Manager
ser = serial.Serial('COM4', 9600, timeout=1)
time.sleep(2)
from naoqi import ALProxy
from PIL import Image
PORT=9559
path = 'D:/' +  '/'
camera = ALProxy("ALVideoDevice", IP, PORT)
resolution = 2 # VGA (640*480px)
colorSpace = 11 # RGB
from apscheduler.schedulers.background import BackgroundScheduler
def sensor():
    line = ser.readline()  # read a byte
    print(line)
    if line == 'w':
            motion.post.moveTo(1, 0.0, 0.0)
    if line == 's':
            motion.stopMove()
    if line == 'r':
            motion.post.moveTo(0, 1, 0)
    if line == 'b':
            motion.post.moveTo(-1, 0, 0)
    if line == 'l':
            motion.post.moveTo(0, -1, 0)

sched = BackgroundScheduler(daemon=True)
```

```
sched.add_job(sensor,'interval',seconds=1)
sched.start()
def pic(_name, CameraIndex):
    videoClient        =        camera.subscribeCamera("python_client",
CameraIndex, resolution, colorSpace, 5)
    naoImage = camera.getImageRemote(videoClient)
    camera.unsubscribe(videoClient)
    # Get the image size and pixel array.
    imageWidth = naoImage[0]
    imageHeight = naoImage[1]
    array = naoImage[6]
    im    =    Image.frombytes("RGB",    (imageWidth,    imageHeight),
str(array))
    im.save(_name, "PNG")


def generate_frames():
    while True:
        pic(path + 'f3.png', 0)
        ## read the camera frame
        frame = cv2.imread('D:/f3.png')
        ret, buffer = cv2.imencode('.jpg', frame)
        frame = buffer.tobytes()

        yield (b'--frame\r\n'
               b'Content-Type: image/jpeg\r\n\r\n' + frame + b'\r\n')


@app.route('/')
def index():
    return render_template('index.html')


@app.route('/video')
def video():
    return  Response(generate_frames(),  mimetype='multipart/x-mixed-
replace; boundary=frame')
```

```


if __name__ == '__main__':
        app.run(host='0.0.0.0', port=5000, debug=True)
```

- In terminal of the pc write ipconfig
- Look for ipv4 address (it the Ip address of Ur network)
- In pycharm terminal write flask run -h ipv4 address
- The given link will take u directly to the web page where u find the streaming

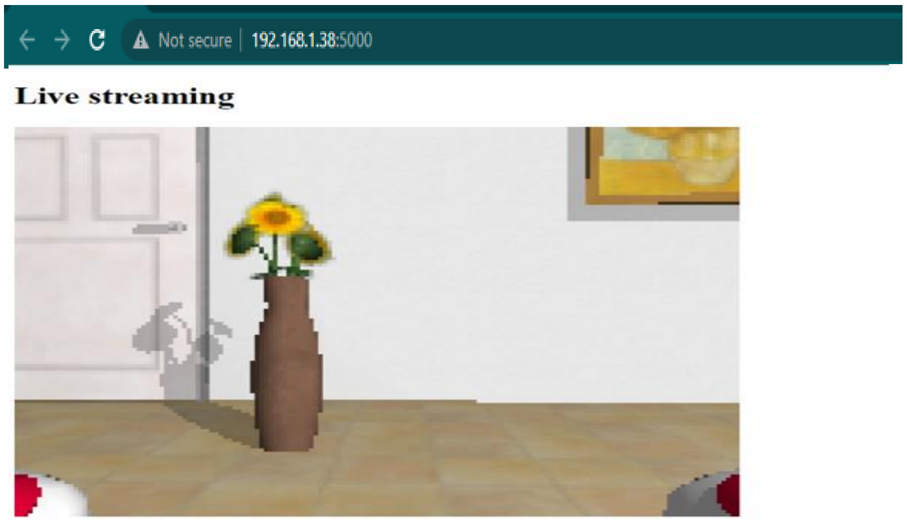


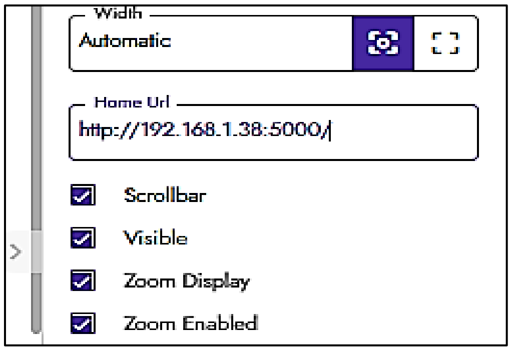
Figure V-8 the streaming in the web page

Note: this streaming was tested on simulated and real robot and the figure show the simulated robot vision

V.1.5.5 Streaming in Kodular app:

web viewer component: Component for viewing Web pages. The view can be set to follow links when they are tapped, and users can fill in Web forms. Pay attention that is not a full browser. For example, pressing the phone's hardware Back key will exit the app, rather than move back in the browser history.

- In pallet go to layout click viewer drag and drop web viewer
- Write the given link in flask project in home URL


Width
Automatic

Home Url
http://192.168.1.38:5000/

☑ Scrollbar
☑ Visible
☑ Zoom Display
☑ Zoom Enabled

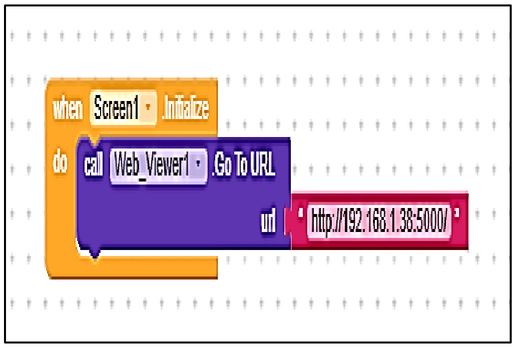

```
when Screen1 .Initialize
do call Web_Viewer1 .GoToURL
        url " http://192.168.1.38:5000/ "
```

Figure V-9 web viewer bloc code

- **Note** : since in flask project the only function run is the flask app we did time scheduler library it used when you want to run in the background inside your application
- So the streaming and control take turns every second

V.1.6 Final result:

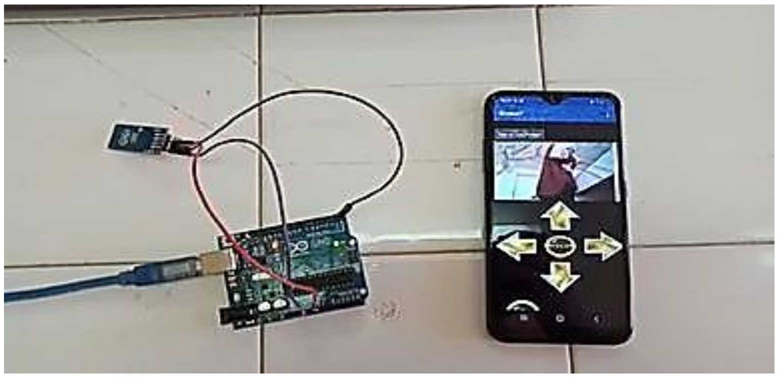


Figure V-10 the application final result

V.1.7 Android application Full project plan

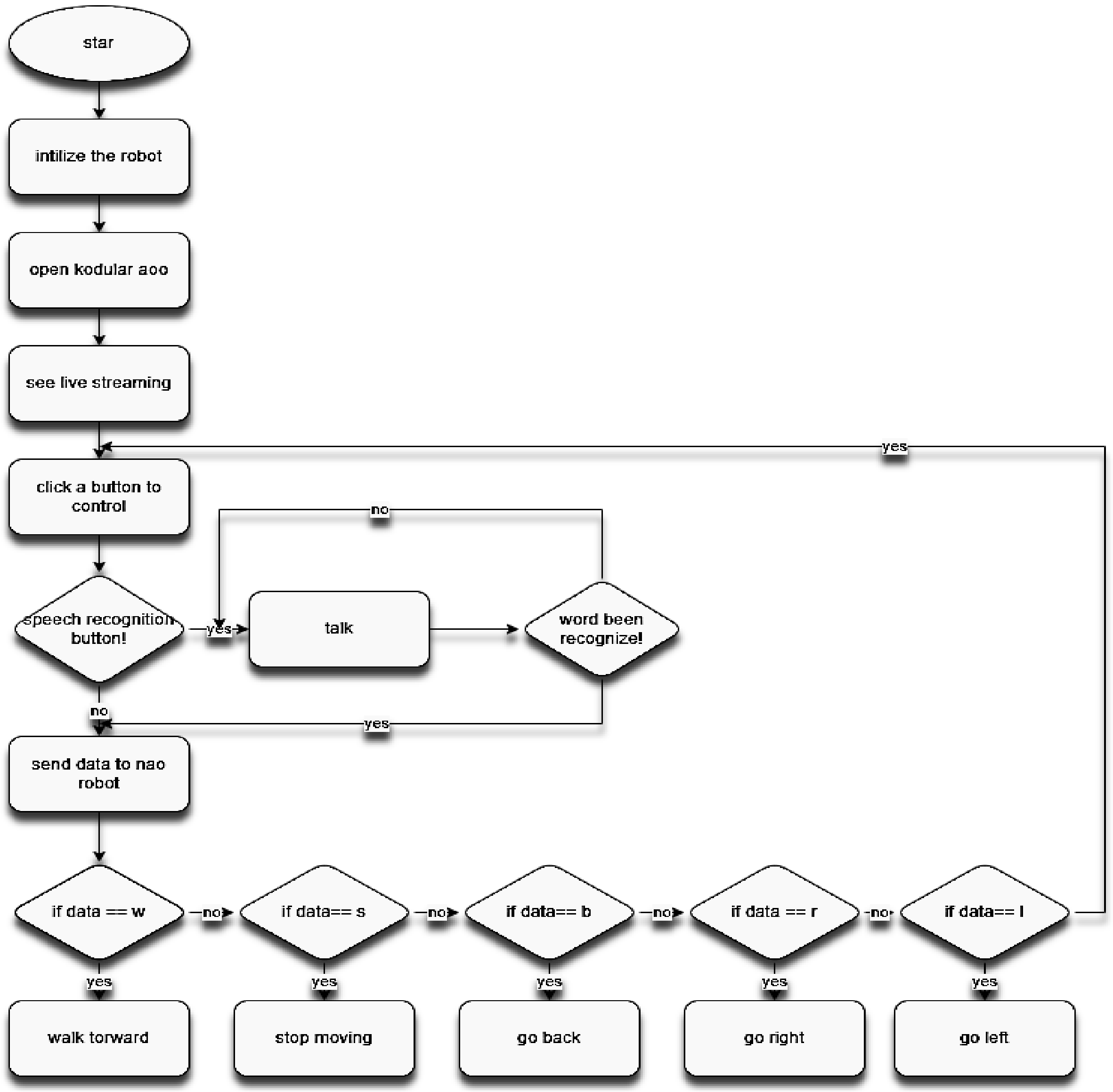


Figure V-11 Android application Full project plan

V.2 Autonomous navigation and tracking

To help people with special needs in their daily life robot could facilitate that with bringing object and peining helpful to them.

Autonomous navigation and tracking require appropriate modeling of environment. Over the years, incredible progress and a many algorithms has been made for this filed For this project we tried to develop an algorithm for nao ; NAO wears two cameras on his head, giving him a wider field of vision and allowing him to see in front of him and at his feet. Because of his vast capabilities and easy-to-use SDK, he's a good candidate for combining computer vision and motion dynamics into a single project.Our approach to make this project will be broken into 4 main parts: Initial scan of the room to find the ball, positioning the robot facing the ball, walking to the ball while updating the position every certain distance based on the visual information and finally grip the ball.

V.2.1 Autonomous navigation and tracking plan

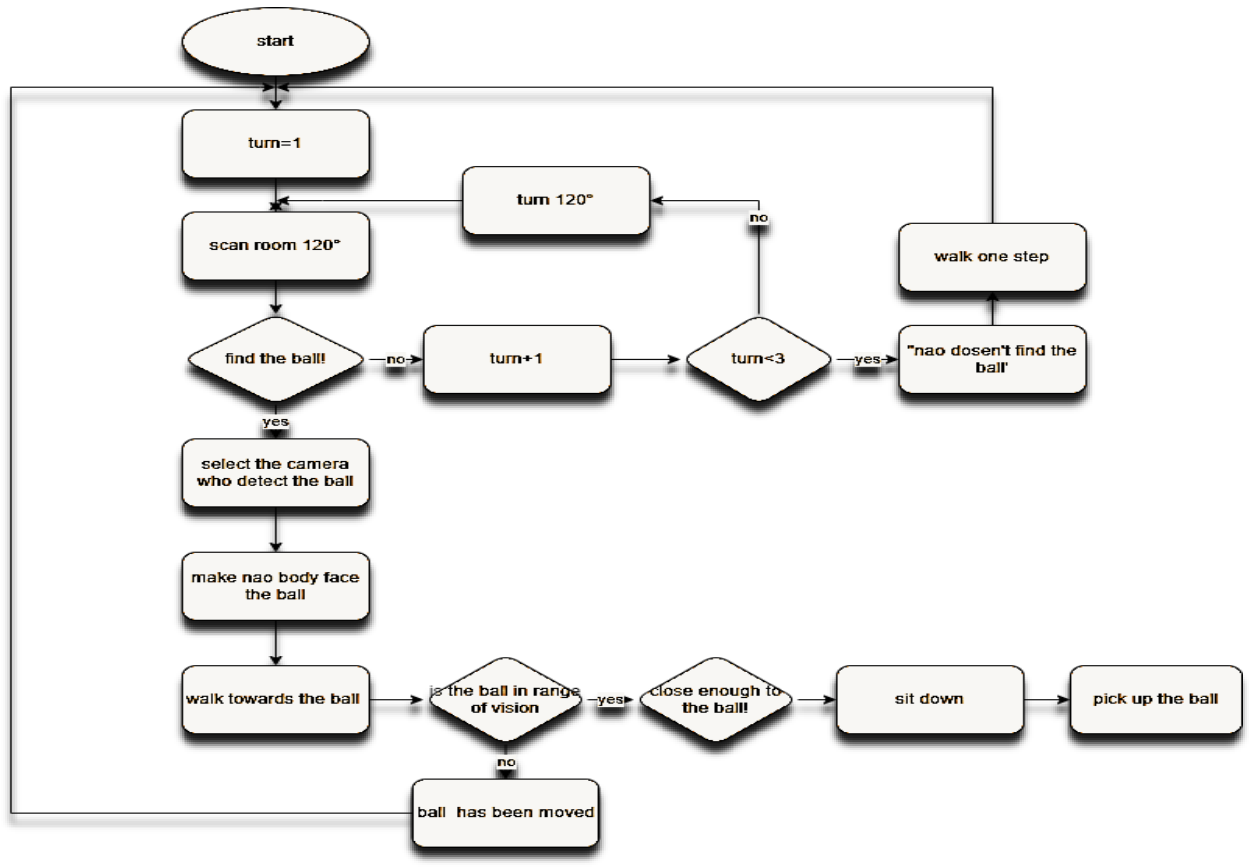


Figure V-12 Autonomous navigation and tracking plan

V.2.2 Scan the place to find the ball:

Nao robot head joint angle according to this table

In this part nao robot will scan the place take a few pictures while his moving his head process those images to find the ball, nao robot move his head 120° each time

V.2.3 Finding the ball:

We will be using nao camera to take picture and Opencv process it and find the ball To find the ball in image follow the following steps

1- Threshold image in HSV color space

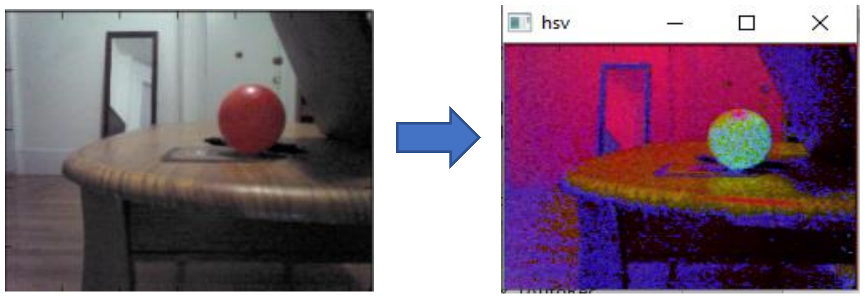


Figure V-13 the result of converting to HSV color space

2- Calculate the center of mass with this equation

The first and simplest method used is a simple center of mass calculation

$$C_m = [x_{C_m} y_{C_m}]$$

$$x_{C_m} = \frac{\bar{m}_x . \bar{x}^T}{M} \qquad y_{C_m} = \frac{\bar{m}_y . \bar{y}^T}{M}$$

This was done because ball pixels are clumped together in the masks formed by thresholding in HSV color space, and given a perfect threshold, this will yield the precise center of the ball. Furthermore, because the weight of every coordinate is proportional to the number of pixels in that coordinate, single noise pixels should have no effect on the center of mass computation. This is because the ball has a big number of pixels in one location compared to a small number of noise pixels scattered over the image. This is a relatively straightforward procedure:

1- Find contours around groups of identified pixels

2- Find a minimum enclosing circle around each contour

3- Find the contour with the smallest error metric that is within the range of expected radius values

4- Reject contour if error metric is too large

The code of finding and draw a circle on the ball

V.2.3.1 python code of finding the center of mass of the ball and draw a circle on it

```python
import sys
import time
import math
import cv2
import numpy as np
# Python Image Library
from PIL import Image
from naoqi import ALProxy
import vision_definitions
import time
def CenterOfMass(image):
    hsv = cv2.cvtColor(image, cv2.COLOR_BGR2HSV)
    # 100 lower range of saturation for daveed's place, day.
165 night 95 sat; 10 hue bottom
    # 195 lower range of saturation for travis' place
    lowera = np.array([160, 95, 0])
    uppera = np.array([180, 250, 255])
    lowerb = np.array([0, 95, 0])
    upperb = np.array([10, 250, 255])
    mask1 = cv2.inRange(hsv, lowera, uppera)
    mask2 = cv2.inRange(hsv, lowerb, upperb)
    mask = cv2.add(mask1, mask2)
    kernel = np.ones((5, 5), np.uint8)
    mask = cv2.morphologyEx(mask, cv2.MORPH_OPEN, kernel)
    mask = cv2.morphologyEx(mask, cv2.MORPH_CLOSE, kernel)
    cont, hier = cv2.findContours(mask, cv2.RETR_TREE,
cv2.CHAIN_APPROX_NONE)[-2:]
    icenter = []
    if len(cont) >= 1:
        maxim = 0
        for i in range(len(cont)):
            if len(cont[i]) > len(cont[maxim]):
                maxim = i
        contour = cont[maxim]
        center, radius = cv2.minEnclosingCircle(contour)
        icenter.append(int(center[0]))
        icenter.append(int(center[1]))
        radius = int(radius)
        print 'Center', center, '. Radius', radius
        if radius >= 8 and radius < 60:
            i = icenter[1]
            j = icenter[0]
        else:
            i = 0
            j = 0
    else:
        i = 0
        j = 0
        icenter = [1, 1]
        contour = cont
        radius = 1
    CM = [i, j]
    return CM, contour, icenter, radius
if __name__ == '__main__':
    image = 'D:/ball_downfront1.png'
    img = cv2.imread('D:/ball_downfront1.png')
    CM, cont, icenter, radius = CenterOfMass(img)
    cv2.drawContours(img, cont, -1, (0, 190, 255), 2)
    cv2.circle(img, (icenter[0], icenter[1]), radius, (255,
0, 0),
    cv2.circle(img, (CM[1], CM[0]), 2, (0, 255, 0), 3)
    cv2.imshow('detected ball', img)
    cv2.waitKey(0)
    cv2.destroyAllWindows()
    cv2.imwrite("CircTest0.png", img)
```

- The result:

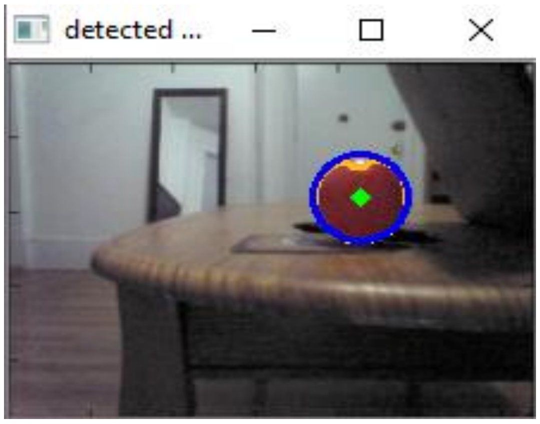


Figure V-14 ball detection result

V.2.4 Centering the robot facing the ball

Once we have the location of the center of the ball, we will make use of pixel information of the image to rotate the robot and face it directly. By using a 640x480 pixel image, we need to have the ball centered at 320. We need at least 2 images where the ball has been detected, and we also make use of the rotation angles where every picture has been taken. Fusing this two different information's we can calculate the closest rotation angle, which gives us a good centering of the ball.

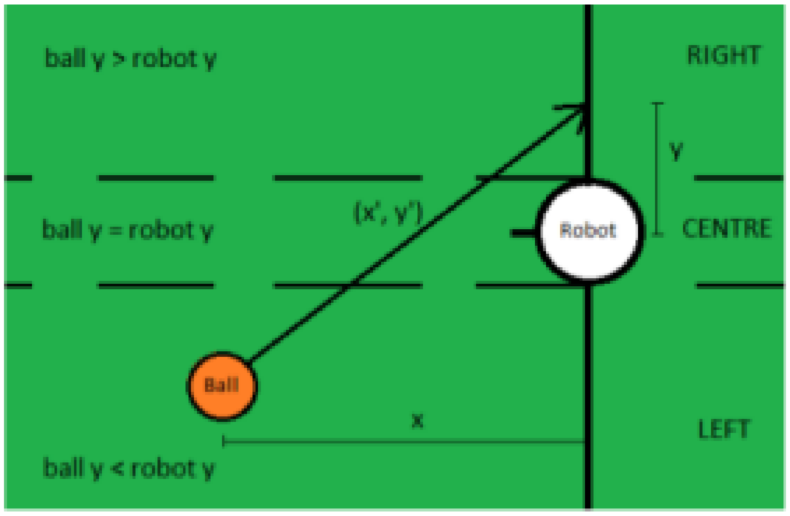


Figure V-15 center the robot face the ball

V.2.5 Walking to the ball

Once it has aligned its body facing the ball, NAO will start walking towards it. Because NAO doesn't have feedback for its odometry, information about real data can't be extracted.

When NAO is more than 10cm away from the ball, it will do a 20cm displacement and correct the drifting using visual information from his cameras (upper or lower, depending on where it sees the ball). As we are trying to have NAO stand right in front of the ball, it needs to walk in a more precise manner once it gets close enough (10cm), so we rely on the visual information from its lower camera again.

V.2.6 Conclusion

The result of this project was disparate the scanning and finding the ball gives pretty good result but walking to the ball because of stability of the robot when walking not so good sometimes it miss the ball but

V.3 Robot communication with humans

This project presented face detection to achieve human-following function. The NAO robot could avoid obstacles and move to human beings. This project included Google speech recognition, which converted audio files to text files and enabled the NAO robot to carry out a simplified communication with human beings, using OpenCV trained cascade, which recognized human’s face and the corresponding box. and drew face frames and face’s confidence level was used. Also get used with the robot sensor such as tactile and sonar sensors Finally, the work utilized NAOqi modules to control the whole-body joint of the NAO robot and employ the NAO robot to follow human

V.3.1 the plan:

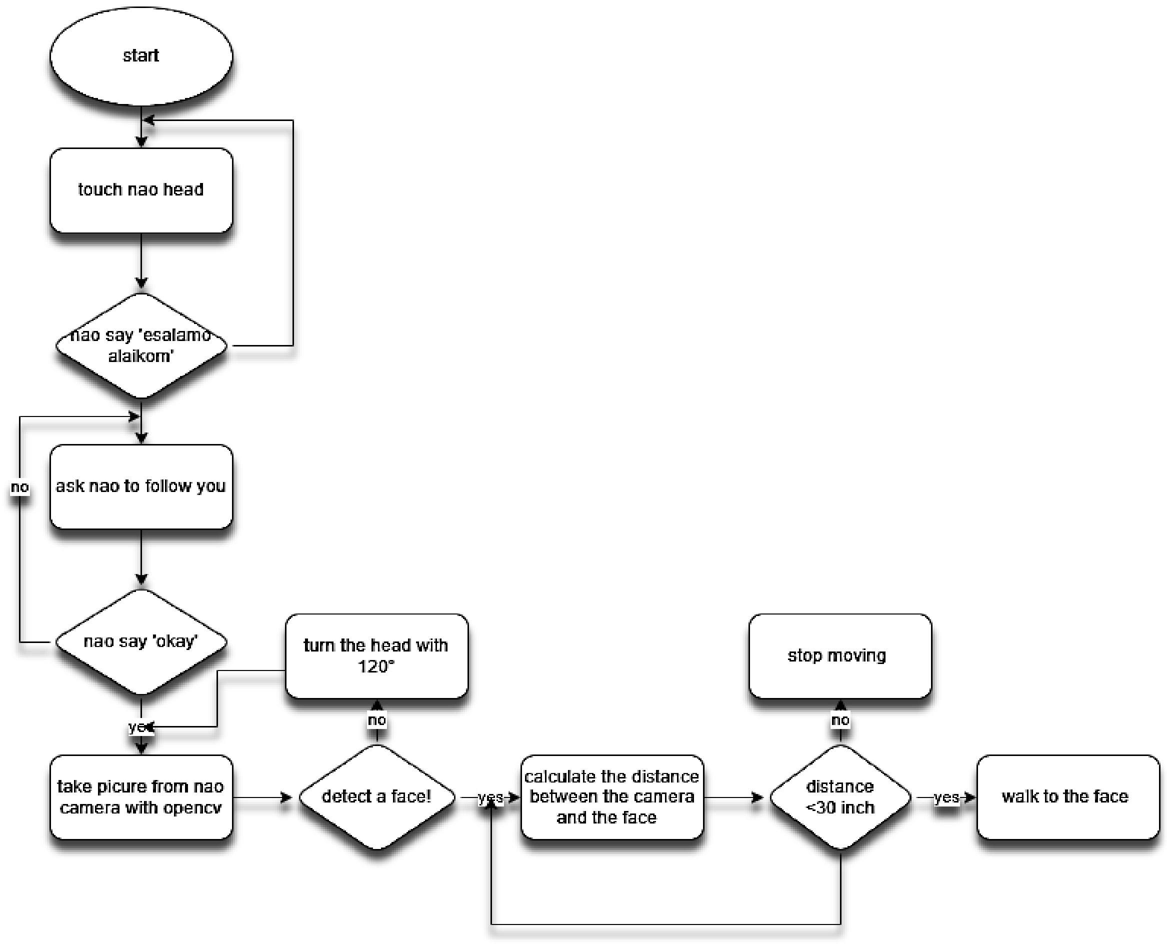


Figure V-16 the plan of human following project

V.3.2 ALTouch module

1- Subscribe to event

2- Print the touched sensor

3- Say “esalomo alaikom”

```python
import argparse
from naoqi import ALProxy
import time
def main(robot_IP, robot_PORT=9559):
        touch = ALProxy("ALTouch", robot_IP, robot_PORT)
                status = touch.getStatus()
        counter = 0
        for e in status:
                if e[1] == True:
                        tts.say("esalamo alaykom")
                        break
                time.sleep(0.2)
```

V.3.2.1 Tactile sensor in nao robot list:

```python
        print touch.getSensorList()

#       output:
#       ['Head/Touch/Front', 'Head/Touch/Middle', 'Head/Touch/Rear',
#       'LHand/Touch/Back', 'LHand/Touch/Left', 'LHand/Touch/Right',
#       'RHand/Touch/Back', 'RHand/Touch/Left', 'RHand/Touch/Right',
#       'LFoot/Bumper/Left', 'LFoot/Bumper/Right',
#       'RFoot/Bumper/Left', 'RFoot/Bumper/Right']
```

V.3.2.2 The status of the sensor in nao robot

```python

        print touch.getStatus()

#         output:
#       [
#               ['Head', False, []],
#                ['LArm', False, []],
#                ['LLeg', False, []], ['RLeg', False, []],
#                ['RArm', False, []],
#                ['LHand', False, []], ['RHand', False, []],
#                ['Head/Touch/Front', False, []],
['Head/Touch/Middle', False, []], ['Head/Touch/Rear', False, []],
#                ['LFoot/Bumper/Left', False, []],
['LFoot/Bumper/Right', False, []],
#                ['RFoot/Bumper/Left', False, []],
['RFoot/Bumper/Right', False, []],
#                ['LHand/Touch/Left', False, []], ['LHand/Touch/Back',
False, []], ['LHand/Touch/Right', False, []],
#                ['RHand/Touch/Left', False, []], ['RHand/Touch/Back',
False, []], ['RHand/Touch/Right', False, []]
#        ]
```

V.3.3 Speech recognition module in nao robot

To access to that module do the following steps

1- Import the module
2- Set the language
3- Set the vocabulary
4- Subscribe to the asr event
5- Subscribe to the memory
6- Get the data from the memory
7- If the data == vocabulary do the action

```
asr = ALProxy("ALSpeechRecognition",robot_IP, robot_PORT)
asr.setLanguage("English")
vocabulary = ['come to me']
asr.setVocabulary(vocabulary, False)
asr.subscribe("TEST_ASR")
data = (None, 0)
while not data[0]:
    data = memory.getData("WordRecognized")
#stops listening after he hears yes or no
```

```
asr.unsubscribe("TEST_ASR")

if data[0]!="follow me":
        tts.say('am coming ')
```

V.3.4 Face detection:

OpenCV already contains many pre-trained classifiers for face, eyes, smile, etc. Those XML files are stored in a folder. We will use the face detection model

```
face_cascade = cv2.CascadeClassifier('face_detector.xml')
```

- Read the image pictured from nao with opencv
- Thanks to people contributing to OpenCV. Here is the code that detects faces in an image, OpenCV train cascade is one of deep learning methods

```
#Detect faces
faces = face_cascade.detectMultiScale(img, 1.1, 4)
```

- After the faces are detected, we will draw rectangles around them so that we know what the machine sees

```
for (x, y, w, h) in faces:
  cv2.rectangle(img, (x, y), (x+w, y+h), (255, 0, 0), 2))
```

- Last, we show the image with opencv
- The result

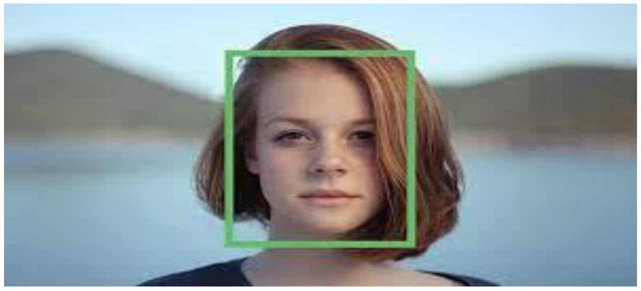


Figure V-17 face detection resut

V.3.5 Calculate the distance between human and NAO

The camera's image-forming principle is shown in the figure below

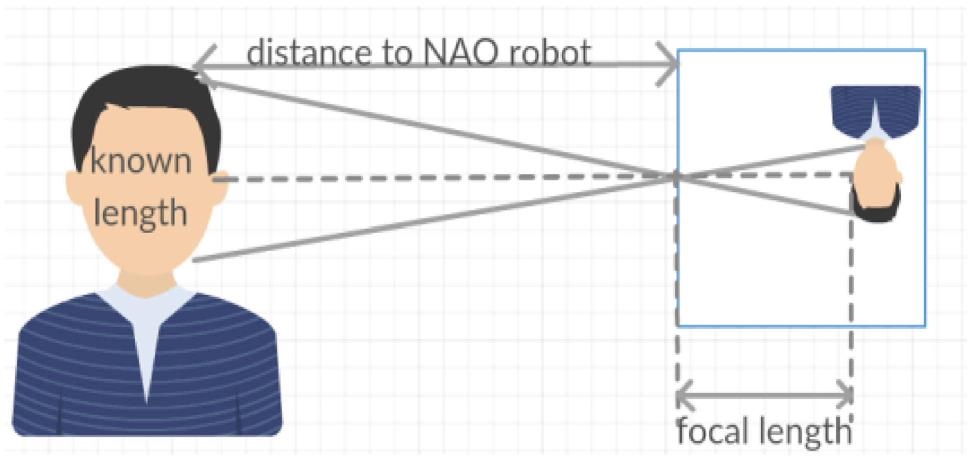


Figure V-18 The camera's image-forming principle

OpenCV will get NAO robot camera's apparent width in pixels. According to the triangle similarity formula for object to camera distance is:

$$F = \frac{(P \times D)}{W}$$

Camera's apparent width in pixels is multiplied by Distance from the camera to the human and then divided by object with known width. It will get the focal length of the NAO robot camera. As the result, this triangle similarity formula already has three fixed values: focal length, known width and camera's pixels. In the following, each images distance can be calculated

$$D' = \frac{(W \times F)}{P}$$

At the beginning of a human-following the NAO robot project, the first distance between the human to the NAO robot is initialized 30 inches, when the known face length is 8 inches

1- Get the focal length

```
KNOWN_DISTANCE = 30.0
KNOWN_PERSON_HEIGHT = 8
img = cv2.imread('/home/nao/WORKSPACE/src/cameras/image_0.png')
mask = self.find_face(img, face_cascade)
for (x1, y1, x2, y2) in mask:
    # cv2.rectangle(img, (x1, y1), (x2, y2), (255, 255, 0), 2)
    y1_max = y1
    y2_max = y2
pix_height = y2_max - y1_max
real_focal_length = (pix_height * KNOWN_DISTANCE) / KNOWN_PERSON_HEIGHT
```

2- calculate the distance between the camera and the face

```
def distance_to_camera(self, known_width, focal_length, per_width):
 return (known_width * focal_length) / per_width
```

V.3.6 Conclusion

We did this project so the robot would communicate with the person and it gives pretty good result

V.4 Hand gesture recognizing

We also think of people with speech disabilities so they would communicate with Nao

A hand gesture recognition system provides a natural, innovative and modern way of non-verbal communication. It has a wide area of application in human computer interaction and sign language and since our thesis deal with disable people we provide a deferent way to control nao robot for this project we train a hand recognition module and try to apply it nao robot

V.4.1 The plan for hand recognition project

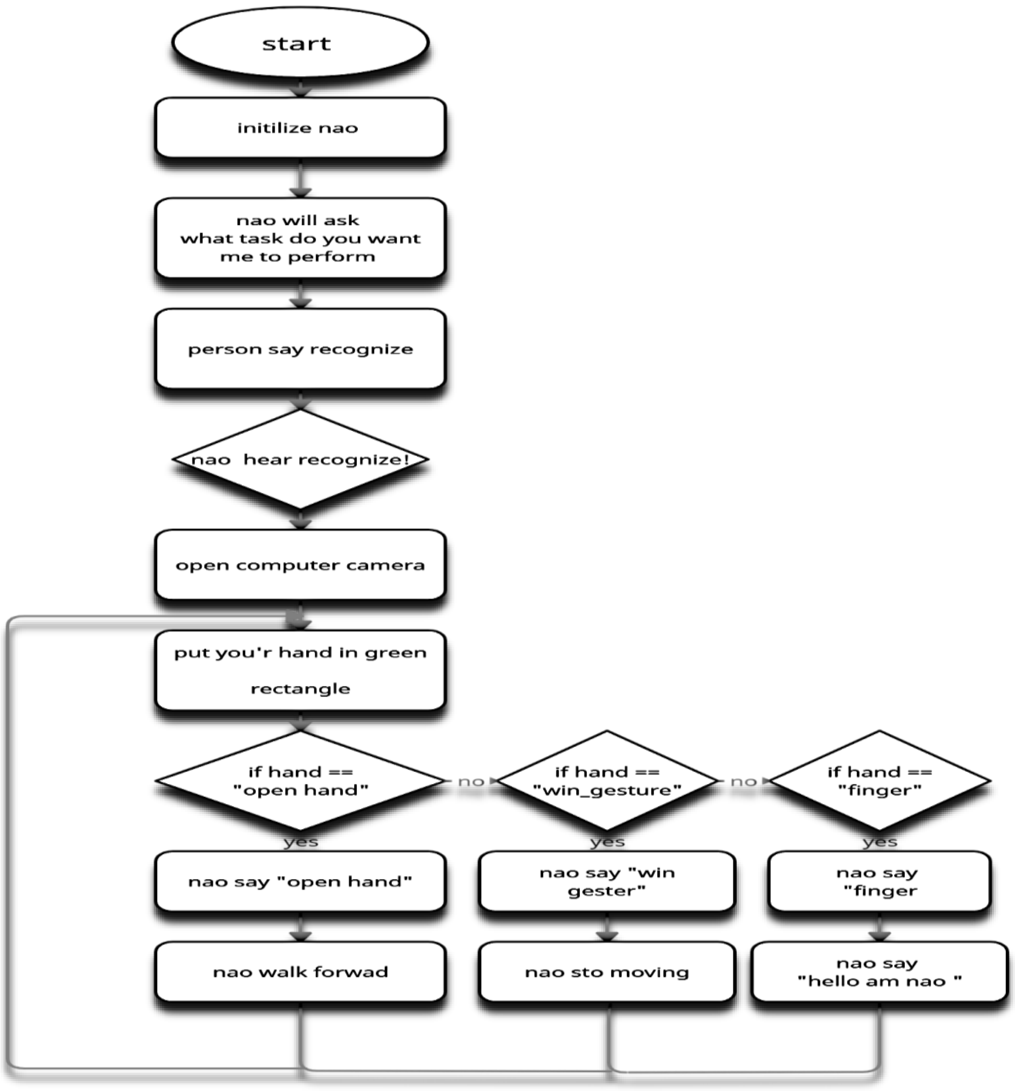


Figure V-19 the pan for hand recognition project

V.4.2 Training hand recognition model

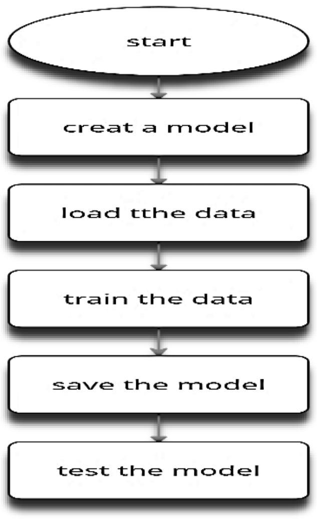


Figure V-20 training model with TensorFlow stage

V.4.2.1 To create a dataset

take a lot of binary pictures of each gesture using opencv

- Result:

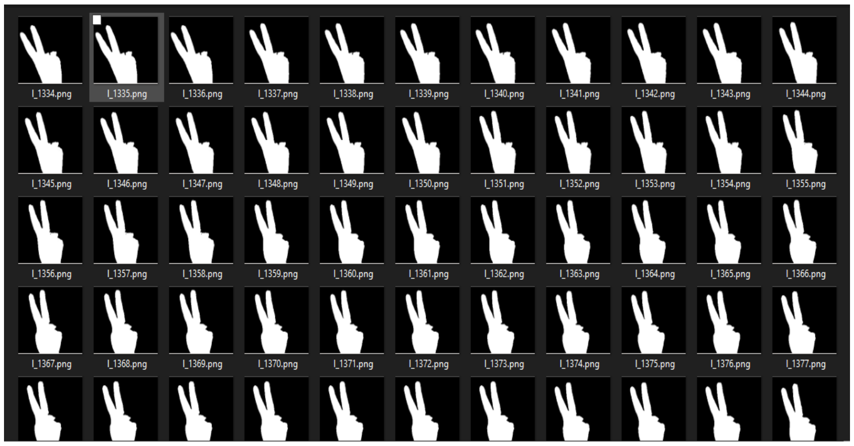


Figure V-21 win_gesture data

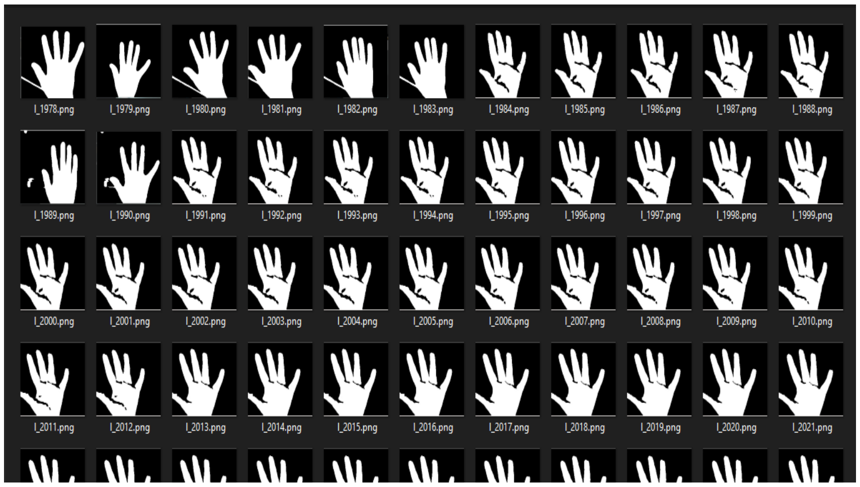

l_1978.png l_1979.png l_1980.png l_1981.png l_1982.png l_1983.png l_1984.png l_1985.png l_1986.png l_1987.png l_1988.png
l_1989.png l_1990.png l_1991.png l_1992.png l_1993.png l_1994.png l_1995.png l_1996.png l_1997.png l_1998.png l_1999.png
l_2000.png l_2001.png l_2002.png l_2003.png l_2004.png l_2005.png l_2006.png l_2007.png l_2008.png l_2009.png l_2010.png
l_2011.png l_2012.png l_2013.png l_2014.png l_2015.png l_2016.png l_2017.png l_2018.png l_2019.png l_2020.png l_2021.png

Figure V-22 open hand data

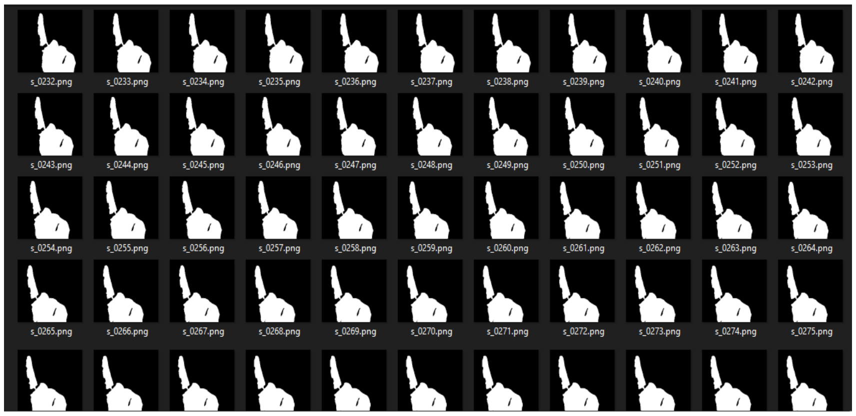
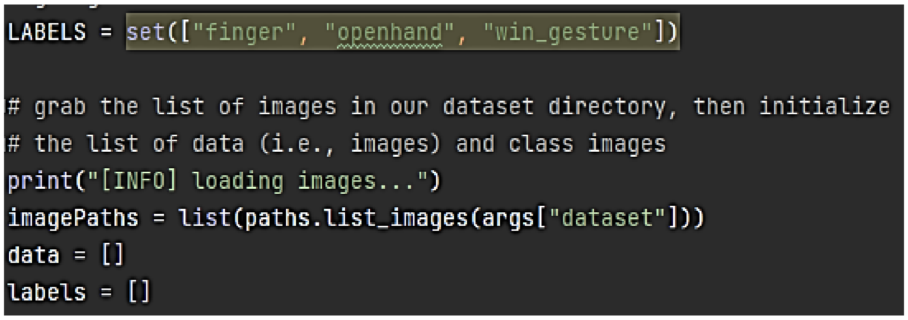

s_0232.png s_0233.png s_0234.png s_0235.png s_0236.png s_0237.png s_0238.png s_0239.png s_0240.png s_0241.png s_0242.png
s_0243.png s_0244.png s_0245.png s_0246.png s_0247.png s_0248.png s_0249.png s_0250.png s_0251.png s_0252.png s_0253.png
s_0254.png s_0255.png s_0256.png s_0257.png s_0258.png s_0259.png s_0260.png s_0261.png s_0262.png s_0263.png s_0264.png
s_0265.png s_0266.png s_0267.png s_0268.png s_0269.png s_0270.png s_0271.png s_0272.png s_0273.png s_0274.png s_0275.png

Figure V-23 finger data

V.4.2.2 Train the data

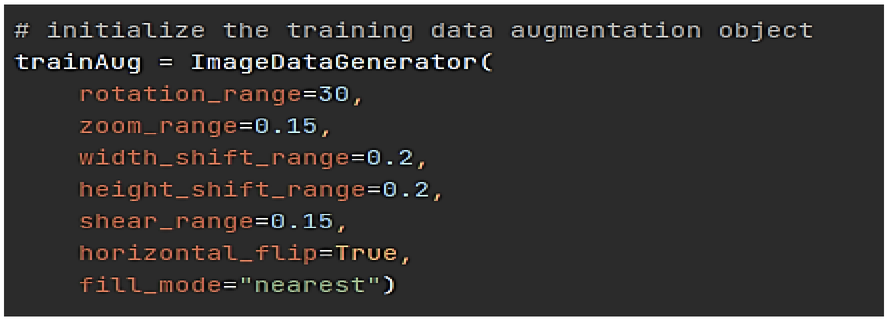
Using Keras with TensorFlow backend and Deep Convolution Neural Network technique

1- Define the path to the data



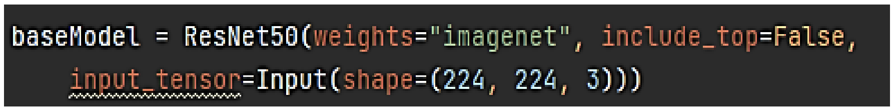
```
LABELS = set(["finger", "openhand", "win_gesture"])

# grab the list of images in our dataset directory, then initialize
# the list of data (i.e., images) and class images
print("[INFO] loading images...")
imagePaths = list(paths.list_images(args["dataset"]))
data = []
labels = []
```

2- initialize the training data augmentation object



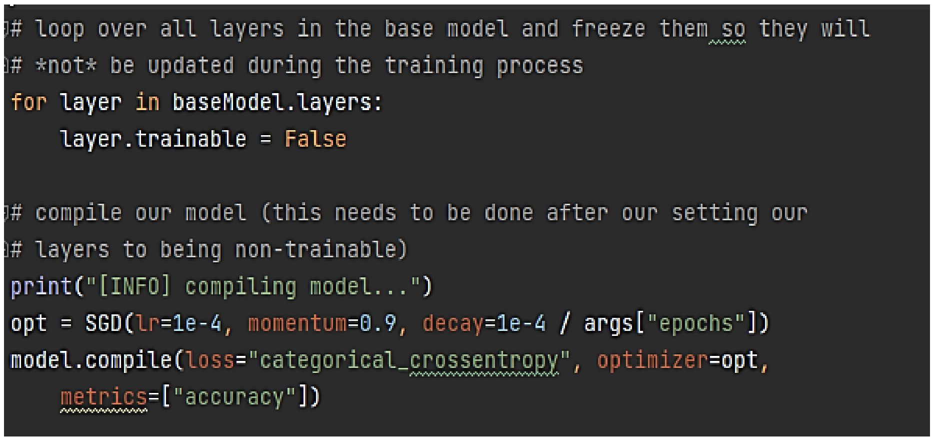
```
# initialize the training data augmentation object
trainAug = ImageDataGenerator(
    rotation_range=30,
    zoom_range=0.15,
    width_shift_range=0.2,
    height_shift_range=0.2,
    shear_range=0.15,
    horizontal_flip=True,
    fill_mode="nearest")
```

3- load the ResNet-50 network (ResNet-50 is a convolutional neural network that is 50 layers deep)



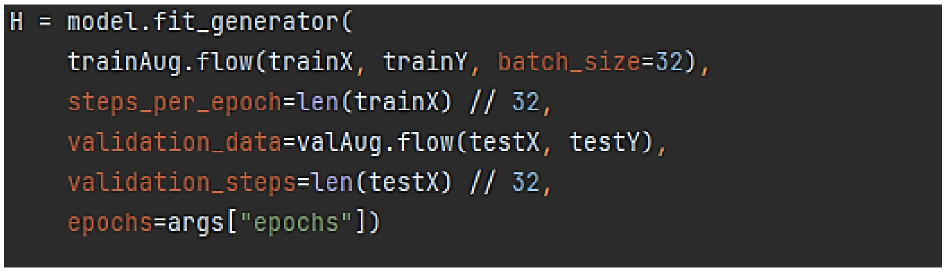
```
baseModel = ResNet50(weights="imagenet", include_top=False,
    input_tensor=Input(shape=(224, 224, 3)))
```

4- loop over all layers in the base model and freeze them so they will not*be updated during the training process and then compile the model



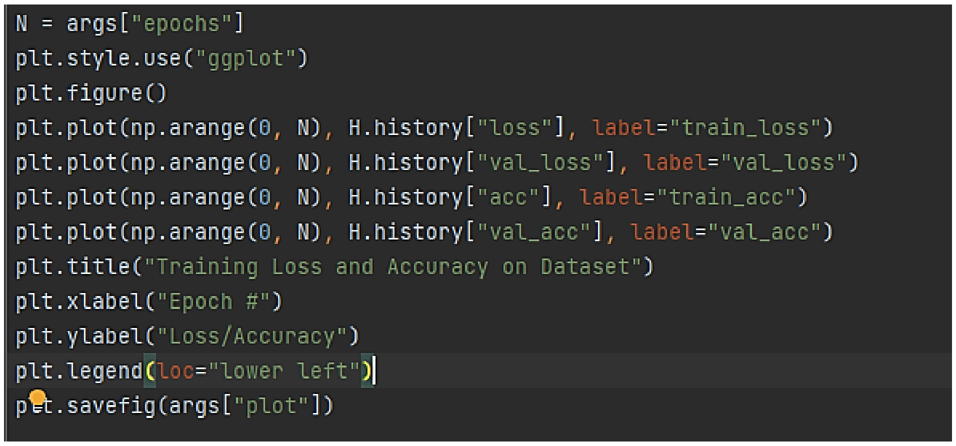
```
# loop over all layers in the base model and freeze them so they will
# *not* be updated during the training process
for layer in baseModel.layers:
    layer.trainable = False

# compile our model (this needs to be done after our setting our
# layers to being non-trainable)
print("[INFO] compiling model...")
opt = SGD(lr=1e-4, momentum=0.9, decay=1e-4 / args["epochs"])
model.compile(loss="categorical_crossentropy", optimizer=opt,
    metrics=["accuracy"])
```

5- train the head of the network for a few epochs (all other layers are frozen) -- this will allow the new CNN layers to start to become initialized with actual "learned" values



```
H = model.fit_generator(
    trainAug.flow(trainX, trainY, batch_size=32),
    steps_per_epoch=len(trainX) // 32,
    validation_data=valAug.flow(testX, testY),
    validation_steps=len(testX) // 32,
    epochs=args["epochs"])
```

6- plot the training loss and accuracy



```
N = args["epochs"]
plt.style.use("ggplot")
plt.figure()
plt.plot(np.arange(0, N), H.history["loss"], label="train_loss")
plt.plot(np.arange(0, N), H.history["val_loss"], label="val_loss")
plt.plot(np.arange(0, N), H.history["acc"], label="train_acc")
plt.plot(np.arange(0, N), H.history["val_acc"], label="val_acc")
plt.title("Training Loss and Accuracy on Dataset")
plt.xlabel("Epoch #")
plt.ylabel("Loss/Accuracy")
plt.legend(loc="lower left")
plt.savefig(args["plot"])
```

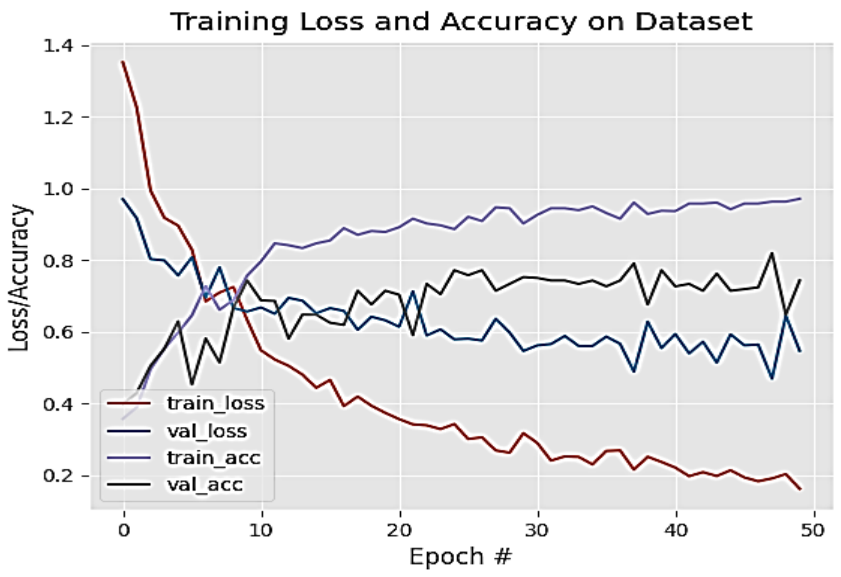
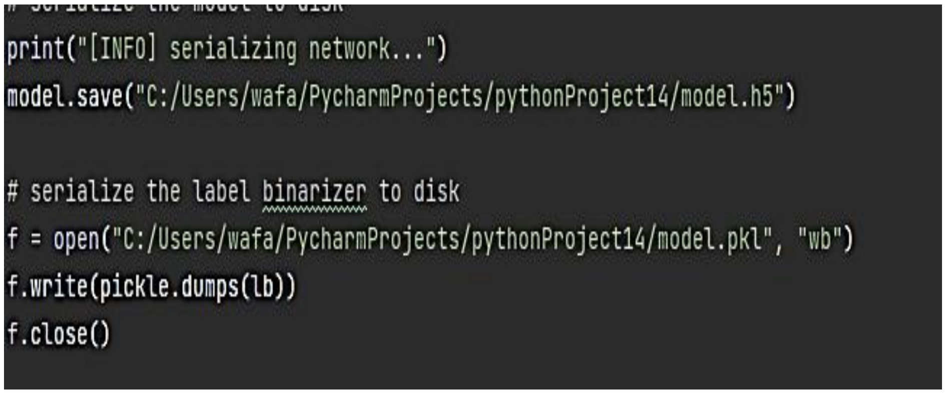


Figure V-24 plot of training loss and accuracy on dataset

7- Save the training in pickles (pickling is the process of converting a Python object into a byte stream to store it in a file/database, maintain program state across sessions, or transport data over the network)then save the model as .h5 file and the pickle as .pkl file



```
print("[INFO] serializing network...")
model.save("C:/Users/wafa/PycharmProjects/pythonProject14/model.h5")

# serialize the label binarizer to disk
f = open("C:/Users/wafa/PycharmProjects/pythonProject14/model.pkl", "wb")
f.write(pickle.dumps(lb))
f.close()
```

- Test the model:

  Since TensorFlow don't support python 2.7 and nao robot only work with that version we use socket library so basically, we create a server to export the result of the code to nao robot make sure to run test file and the server file in the same machine but not the same environment ,The server file has the result of the hand gesture extract from the computer camera, to do so follow this step:

  1- initialize the background
  2- compute weighted average, accumulate it and update the background
  3- segment the region of hand in the image
     - find the absolute difference between background and current frame
     - threshold the diff image so that we get the foreground
     - get the contours in the thresholder image
     - based on contour area, get the maximum contour which is the hand
  4- compare the result with the training model
  5- keep looping, until interrupted

V.4.3 result:

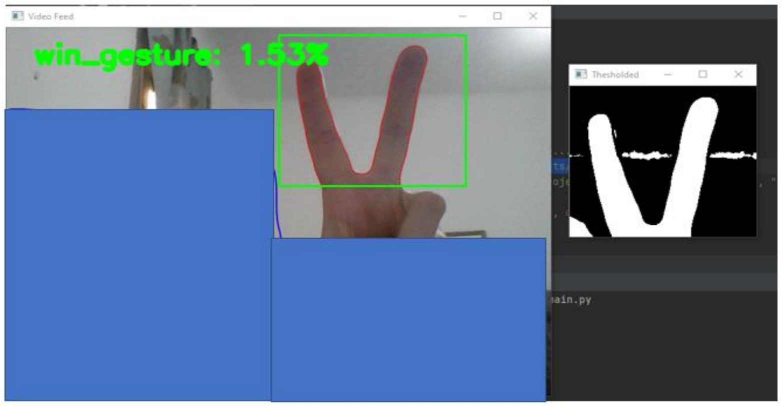


Figure V-25 win_gesture

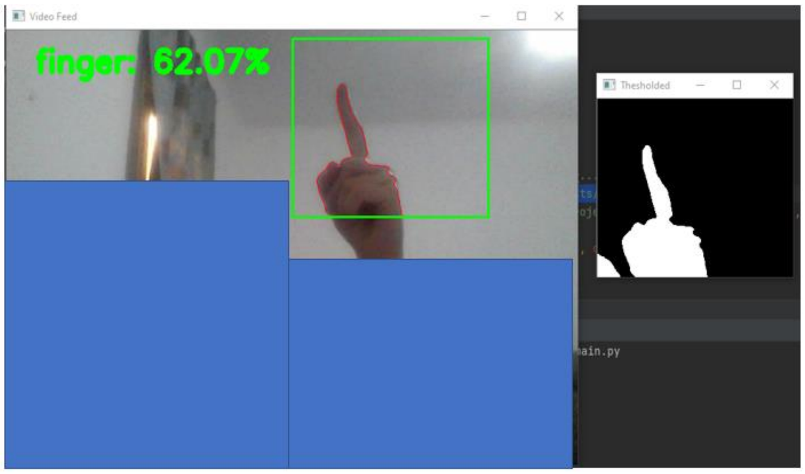


Figure V-26 finger gesture

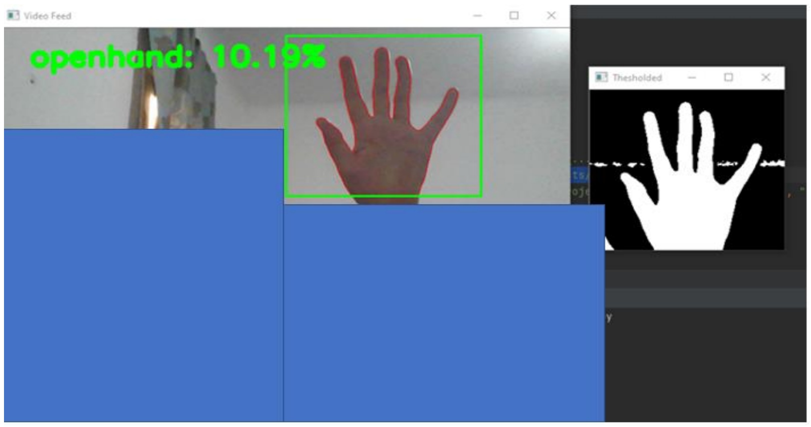


Figure V-27 openhand

V.5 Localization and path planning

the goal from robot localization (robot know and update it location) and path planning is Reach goal destination. Avoid obstacles in the way. Navigation encompasses the ability of the robot to act based on its knowledge and sensor values so as to reach its goal positions as efficiently and reliably

V.5.1 Localization:

Robot localization is the process of determining where a robot is located with respect to its environment. Localization is fundamental competencies required in autonomous robot,knowledge of robot's location is an essential precursor to making decisions about future actions

V.5.1.1 The plan:

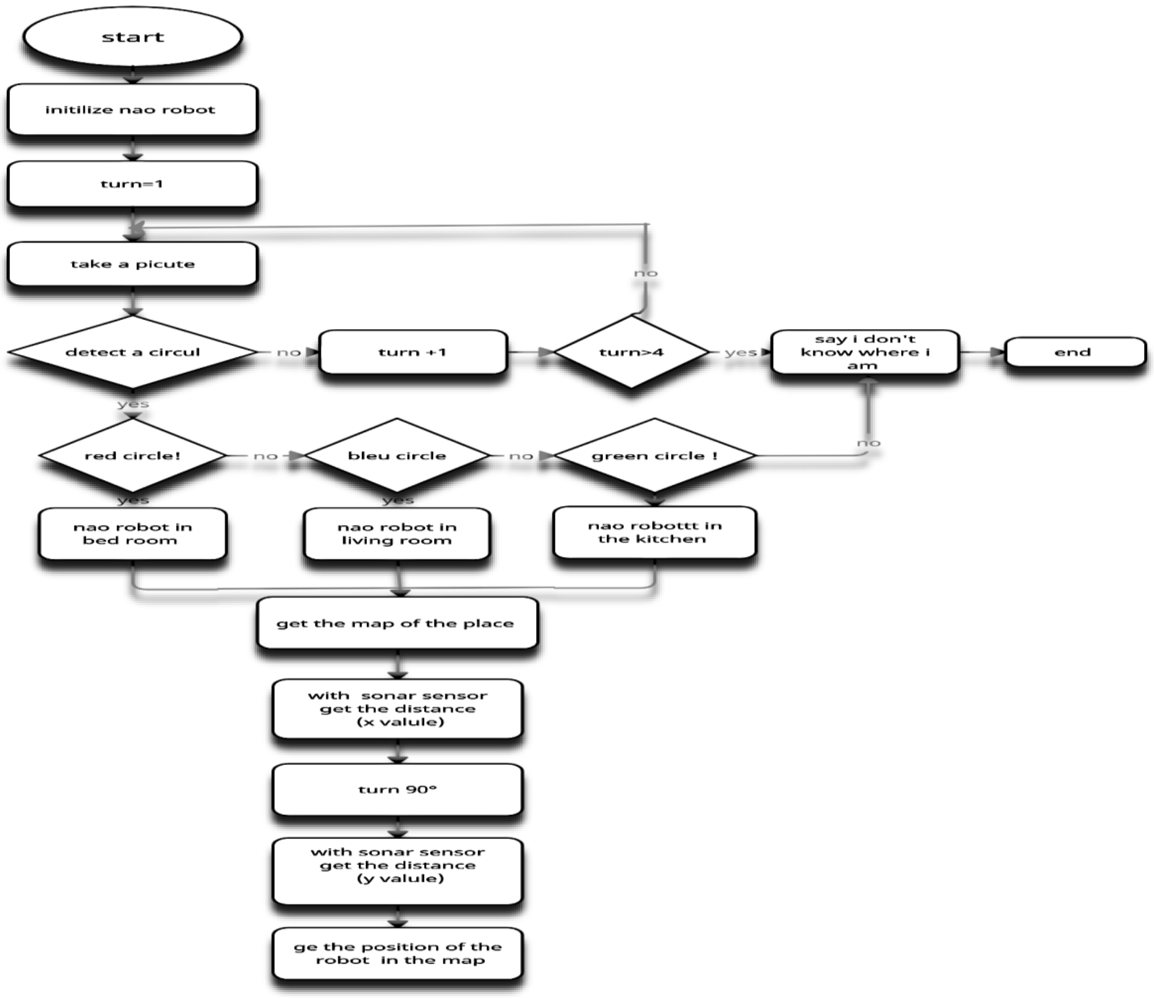


Figure V-28 localization organigram

The first think when nao wake up is to find where is it

We put in every room a landmark in the top on the wall so that it does not mix with other signs in the room. first step when nao wake up is to raises his head, scan the place looking for that landmark

These landmarks are in the shape of a circle in various colors every color define a place

- Red circle represents the bed room
- Green circle represents the living room
- Blue circle represents the kitchen

So basically, what we are trying to do is essentially filter out a color. By default, the images are represented in three channels Blue, Green, and Red. But, using this mode of representation, you cannot filter colors easily as the values are split into three channels. That's where the HSV (Hue, saturation, value) mode of representation comes to play, The line HSV = cv2.cvtColor(img, cv2.COLOR_BGR2HSV) converts the BGR format image to HSV format representation. And just add +-delta value to H channel and you can filter the color accordingly.For example, if you want to filter green color The BGR representation of green color will be (0,255,0). First, we need to find the equivalent color representation in HSV that is (60,255,255). We can add a [H-10, 100,100] and [H+10, 255, 255] as upper and lower values accordingly.

Result:

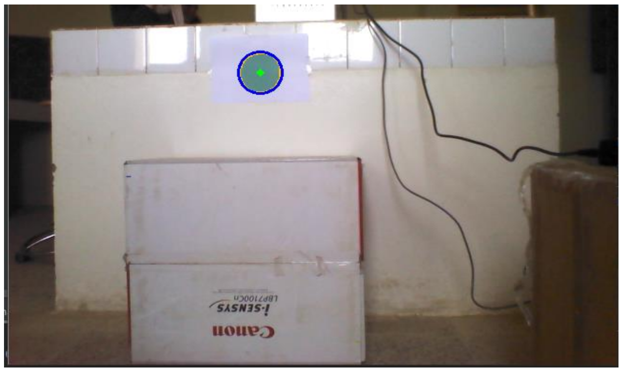


Figure V-29 detect the green circle

In our study, we did a 2D map has the fixed obstacle in each room

Note: this is virtual environment

If NAO detected that it is in the living room by finding the green circle, the living room map will pop up

The boundaries of the map represent the boundaries of the room

And coordinates with red color represent fixed obstacle and other obstacle will be detected in real time

note

The living room map:

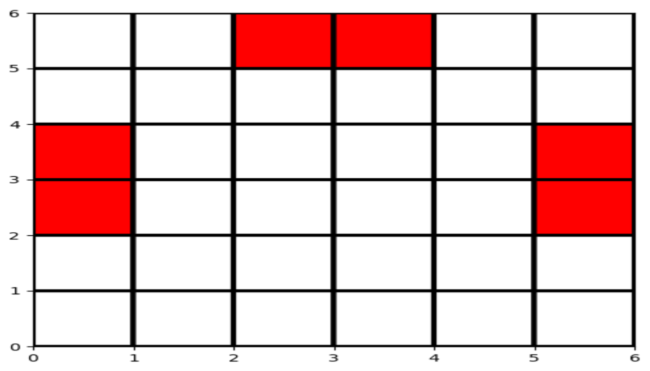


Figure V-30 map for living room

Since the green circle It is located in a certain place for this example located in (0,3) coordinate that means nao find his direction in the room and left only two coordinates so it would find it exact location (x, y coordinate) in the map

Every coordinate in the map has certain values within a specified range (x and y)

To find this coordinate Nao will activate his sonar calculate the distance to get x value and the turn 90° and activate his sonar and calculate the distance to get y value

Consider that the room surroundings are (the width is 2.5 cm and the length also 2.5cm) because the maximum range of nao sonar is 2.5cm

And also taken into account the fixed obstacle

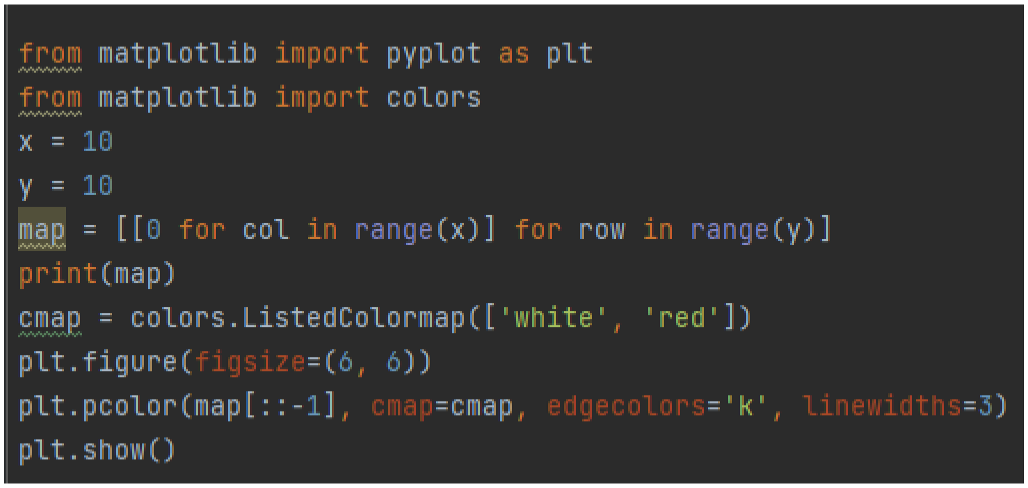
Table 2 coordinate with their sonar range

X=0= 0.1<sonar<0.2999 Y=0=0.1<sonar<0.2999	X=0=0.1<sonar<0.2999 Y=1=0.3<sonar<0.4999	X=0=0.1<sonar<0.2999 Y=2=0.5<sonar<0.6999	X=0=0.1<sonar<0.2999 Y=3=0.7<sonar<0.8999	X=0=0.1<sonar<0.2999 Y=4=0.9<sonar<1.5999	X=0=0.1<sonar<0.2999 Y=5=1.6<sonar<2.5999
X=1=0.3<sonar<0.4999 Y=0=(0.1<sonar<0.2999	X=1=0.3<sonar<0.4999 Y=1=0.3<sonar<0.4999	X=1=0.3<sonar<0.4999 Y=2=0.5<sonar<0.6999	X=1=0.3<sonar<0.4999 Y=3=0.7<sonar<0.8999	X=1=0.3<sonar<0.4999 Y=4=0.9<sonar<1.5999	X=1=0.3<sonar<0.4999 Y=5=1.6<sonar<2.5999
X=2=0.5<sonar<0.6999 Y=0=0.1<sonar<0.2999	X=2=0.5<sonar<0.6999 Y=1=0.3<sonar<0.4999	X=2=0.5<sonar<0.6999 Y=2=0.5<sonar<0.6999	X=2=0.5<sonar<0.6999 Y=3=0.7<sonar<0.8999	X=2=0.5<sonar<0.6999 Y=4=0.9<sonar<1.5999	X=2=0.5<sonar<0.6999 Y=5=1.6<sonar<2.5999
X=3=0.7<sonar<0.8999 Y=0=0.1<sonar<0.2999	X=3=0.7<sonar<0.8999 Y=1=0.3<sonar<0.4999	X=3=0.7<sonar<0.8999 Y=2=0.5<sonar<0.6999	X=3=0.7<sonar<0.8999 Y=3=0.7<sonar<0.8999	X=3=0.7<sonar<0.8999 Y=4=0.9<sonar<1.5999	X=3=0.7<sonar<0.8999 Y=5=1.6<sonar<2.5999
X=4=0.9<sonar<1.5999 Y=0=0.1<sonar<0.2999	X=4=0.9<sonar<1.5999 Y=1=0.3<sonar<0.4999	X=4=0.9<sonar<1.5999 Y=2=0.5<sonar<0.6999	X=4=0.9<sonar<1.5999 Y=3=0.7<sonar<0.8999	X=4=0.9<sonar<1.5999 Y=4=0.9<sonar<1.5999	X=4=0.9<sonar<1.5999 Y=5=1.6<sonar<2.5999
X=5=1.6<sonar<2.5999 Y=0=(0.1<sonar<0.2999)	X=5=1.6<sonar<2.5999 Y=1=0.3<sonar<0.4999	X=5=1.6<sonar<2.5999 Y=2=0.5<sonar<0.6999	X=5=1.6<sonar<2.5999 Y=3=0.7<sonar<0.8999	X=5=1.6<sonar<2.5999 Y=4=0.9<sonar<1.5999	X=5=1.6<sonar<2.5999 Y=5=1.6<sonar<2.5999

V.5.2 Python codes:

- Drawing the map using matplotlib library

First, we create an empty map (full of zeros)

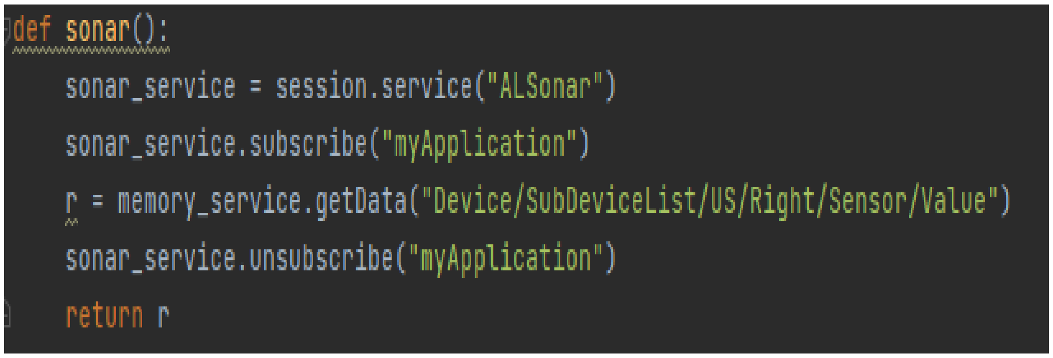


```python
from matplotlib import pyplot as plt
from matplotlib import colors
x = 10
y = 10
map = [[0 for col in range(x)] for row in range(y)]
print(map)
cmap = colors.ListedColormap(['white', 'red'])
plt.figure(figsize=(6, 6))
plt.pcolor(map[::-1], cmap=cmap, edgecolors='k', linewidths=3)
plt.show()
```

Figure V-31 create a map with python

And when map[x][y] =1 it means there is obstacle and colored with red color

- Activate the sonar and get the distance



```
def sonar():
    sonar_service = session.service("ALSonar")
    sonar_service.subscribe("myApplication")
    r = memory_service.getData("Device/SubDeviceList/US/Right/Sensor/Value")
    sonar_service.unsubscribe("myApplication")
    return r
```

Figure V-32 activate the sonar and get the distance python code

After getting x value turn 90° to the left and the get y value

V.5.3 Result

Nao in the last corner of the room in the point (5,5) (the blue point)

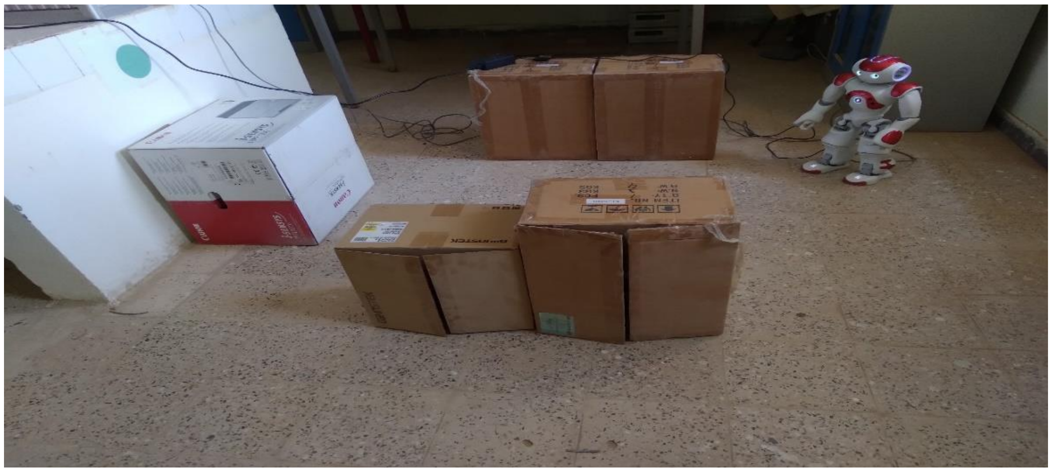


Figure V-33 nao robot localisation

V.5.4 Path planning

Path planning is a robotics field on its own. Its solution gives free path for going from one place to another. Humans do path planning without thinking how it is done. In this part we are going to learn nao robot how it move from place to place and avoid obstacle in his way

V.5.4.1 The plan:

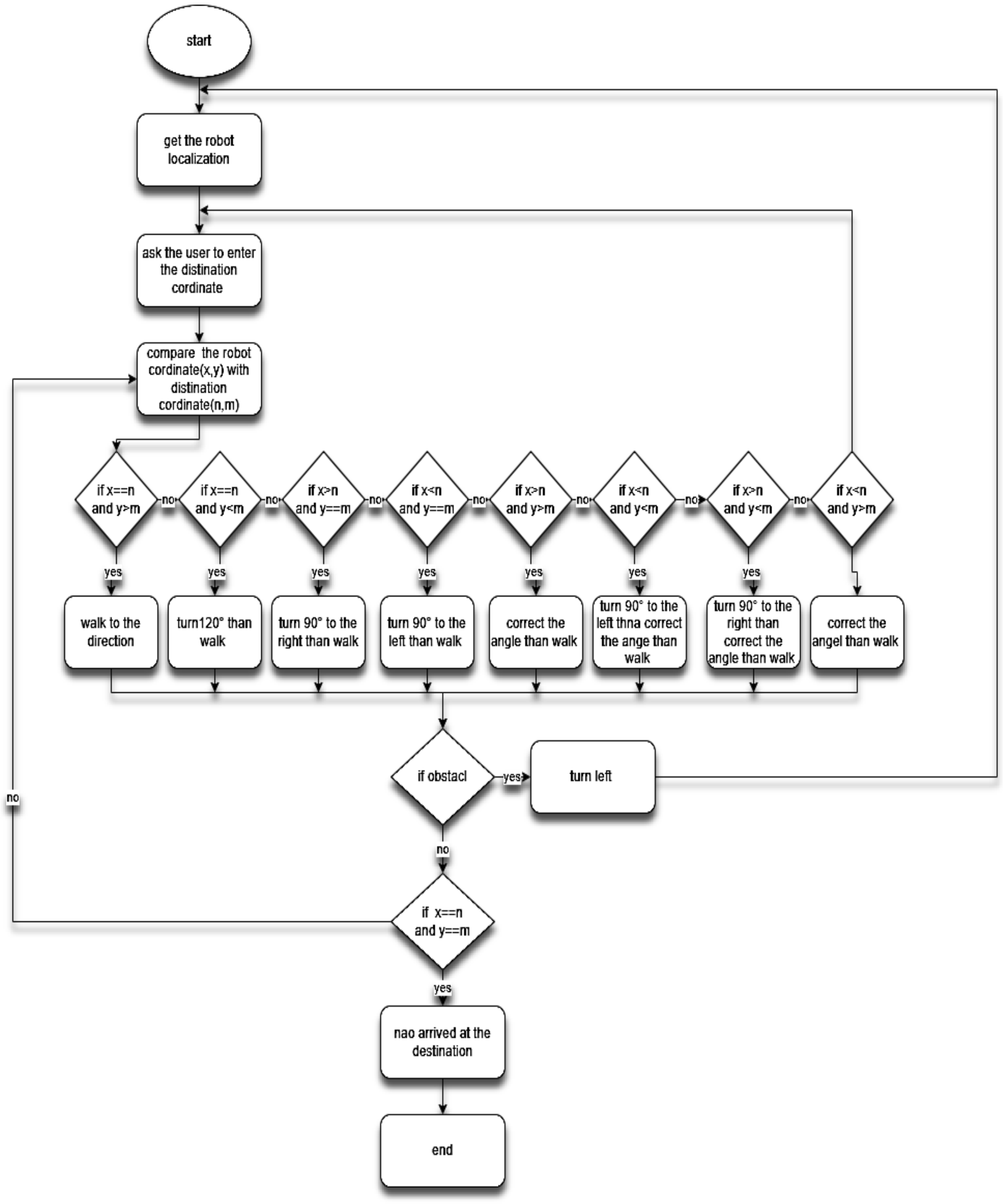

start
get the robot localization
ask the user to enter the distination cordinate
compare the robot cordinate(x,y) with distination cordinate(n,m)
if x==n and y>m
if x==n and y<m
if x>n and y==m
if x<n and y==m
if x>n and y>m
if x<n and y<m
if x>n and y<m
if x<n and y>m
no
yes
walk to the direction
turn120° than walk
turn 90° to the right than walk
turn 90° to the left than walk
correct the angle than walk
turn 90° to the left thna correct the ange than walk
turn 90° to the right than correct the angle than walk
correct the angel than walk
if obstacl
turn left
if x==n and y==m
nao arrived at the destination
end

Figure V-34 path planning organigram

Example: giving an example is the best way to explain the process

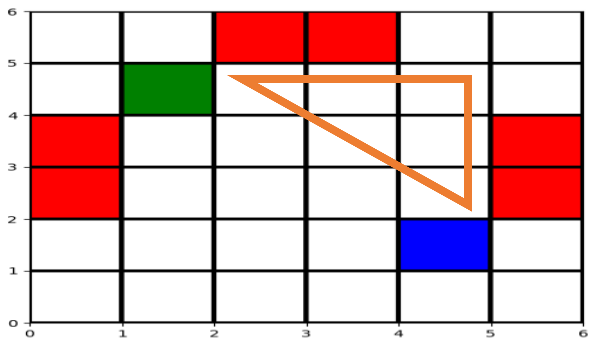


Figure V-35 path planning for the robot

In the figure 99 the robot is in coordinate with blue color and the destination in the coordinate with green color so in order to find the best way to go to that point we notice when dropping the robot coordinate on the x and y parameter and then connect it with destination coordinate an right triangle formed ,Now based on the relationship between the robot point and the destination point will fix the robot direction without forgetting that the robot in the beginning is faced to the left with 90° from the previous part what left is correcting the angle calculate the steps , with using phytagels formular $A^2 + B^2 = C^2$ to find hypotenuse value witch represent the number of steps and calculate the angle of it $\tan\emptyset = tan^{-1}\frac{opposite\ side}{adjacent\ side}$ and we get the turning angle for the robot , now after getting the angle and the number od steps nao will walk to the destination point

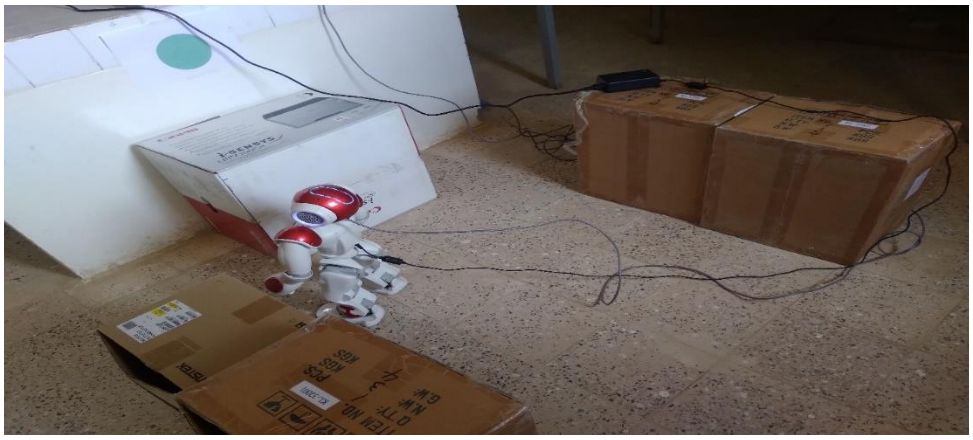


Figure V-36 nao get to the destination coordinate

V.5.5 Conclusion

The result of this project was quite good the only problem the t robot don't do the right angles sometime and has an error in his steps that lead to wrong direction and as future vision for it we would like to add obstacle avoidance and path planning from previous information on the localization and updating his localization every time it walk

Conclusion

Conclusion:

Robotics nowadays is emerging and becoming one the core technologies trends in the 21st century due to the remarkable impact that robots have on humans lives todays. Many scientists, researcher and engineers are trying to improve, expand and develop new technics and methods to make the robots more reliable and efficient.

Nao robot knows a special interests from researches in field of studies This is due to several reasons , his stability and balance , represent a human sense , it has two camera with good properties to develop computer vision, deep learning , machine learning and artificial intelligent algorithm also packed with a deferent sensor from two ultrasonic tree tactile sensors in the head and two in the hand to represent Sense of touch and microphone around his head and speaker for hearing and talking and that help develop many project from speech recognition API , sound tracking , doing dialogue and so on … also sensor used to evaluate the current status to know robot posture so it can manage falling without forgetting his 25 degrees of freedom divided across his body make his movement so smooth and also noticing that the robot company develop ready-made programs for the robot to make it easier to work with it

In this thesis We discussed several of topics to have a bigger look at the field of robotics and we tried to develop are own solution with considering the use of this project for the help of disable people

The first part we did is to get to know this robot and have an overview on his sensors and capabilities and know how to work with it, learn his choregraph platform and how to program it with python

Conclusion

The first project we did is to control the robot from a distance so we thought of doing an android application, we use kodular platform to facilitate creating the app we had an issue to how to send the command to the robot we find a simple solution is adding an Arduino with Bluetooth work as server for the data and send it back to the robot with python program we want to go further so we add speech recognition property in the app to control the robot with voice commands and also we add live streaming from the robot camera so basically we take picture with camera stream it in web server show that page in the app do it in while loop to look like a video

The second project is robot navigation we think that is an important propriety for the robot to find stuff go to it and grasp it, in this project we learn about object detection and using opencv we detect a red ball and face the robot to it , this part was quite challenging to learn about image pixels and how to compare a two image pixels to find the right position for the ball and then walk to the ball and always updating the position of the ball in case it move when the robot Reaches to the ball will grasp it and because of our robot have a broken fingers we end the project when it reach to the ball

For the third project we try to detect the face of a person an go to him so the robot will keep company with the human we get the access to his tactile sensors and his speech recognition API to communicate with the person and detect his face and follow him

People with speech disabilities had a portion of this thesis in the fourth project we train a hand gesture recognition module to control nao with hand sign the only problem we had in this part was the duration of the training because it take over a 7 hours to detect an error and repeat the training all over again but the final result was fine we train three sign open hand and win gesture and single finger

The last project of our thesis was localization and path planning , it consider the most important thing for the robot to be connected to his environment know his location and find

Conclusion

a way to go to certain point , in known environment we use a define shape in room robot will scan the place looking for that shape to know witch room in the house is he the will utilize his sonar to specify his location in that room then the user define the destination coordinate , the robot will find a way to go to it with considering to the obstacle while his moving , this hole concept was quite challenging from finding a simple ideas and using Things available in the robot

References

[1] J. J. Craig, "Introduction to Robotics Mechanics and Control," Pearson Education International, pp. 1-2.

[2] J. Cicolani, " Beginning Robotics with Raspberry Pi and Arduino Using Python andOpenCV," Texas, USA, Pflugerville, 2018.

[3] "journals.sagepub," [Online]. Available: https://journals.sagepub.com/doi/full/10.1177/1729881419839596. [Accessed 11 6 2022].

[4] "softbankrobotics," [Online]. Available: https://developer.softbankrobotics.com/nao6/naoqi-developer-guide/naoqi-apis/naoqi-core/almemory/almemory-tutorial. [Accessed 11 3 2022].

[5] "sciencefocus," [Online]. Available: https://www.sciencefocus.com/buyers-guides/best-drones/. [Accessed 10 6 2022].

[6] "made-in-china," [Online]. Available: https://www.made-in-china.com/showroom/123nexus/product-detailcSznskeCGvVw/China-4WD-Mecanum-Wheel-Mobile-Robot-Kit-10011.html. [Accessed 10 6 2022].

[7] Asst.prof.dr.ilhan.Konukseven, "trends in robot business and robotic research in ME dept," turkey, 2004.

[8] "researchgate," [Online]. Available: https://www.researchgate.net/figure/The-classification-of-robots-and-robotic-devices-2_fig4_235417999.. [Accessed 10 3 2022].

[9] ""The Ever Expanding World of Robots,"," [Online]. Available: http://theeverexpandingworldofrobots.yolasite.com/the-5-ds-of-. [Accessed 2022].

[10] "galileo," [Online]. Available: https://www.galileo.org/robotics. [Accessed 11 3 2022].

[11] A. LAHDEN, "THE BASICS OF ROBOTICS," in *THE BASICS OF ROBOTICS*.

[12] "yourmechanic," [Online]. Available: https://www.yourmechanic.com/article/a-mechanic-s-guide-to-the-parts-of-a-car-by-maddy-martin. [Accessed 2 3 2022].

[13] "tdk," [Online]. Available: https://www.tdk.com/en/featured_stories/entry_037.html. [Accessed 11 3 2022].

[14] "springer," [Online]. Available: https://link.springer.com/referenceworkentry/10.1007/978-3-540-30301-5_9. [Accessed 10 3 2022].

[15] "wikipedia," [Online]. Available: https://en.wikipedia.org/wiki/Humanoid_robot. [Accessed 10 3 2022].

[16] G. P. Vadakkepat, "Humanoid Robotics: A Reference ambarish," Goswami Prahlad Vadakkepat Editors, pp. 148-149.

[17] "wikichip," [Online]. Available: https://en.wikichip.org/wiki/intel/atom/z530#:~:text=Z530%20is%20an%20ultra%2Dlow,on%20a%2045%20nm%20process.. [Accessed 3 4 2022].

[18] Datasheet NAO Next Gen - H21/H25 Model - English version.

[19] "softbankrobotics," [Online]. Available: https://www.softbankrobotics.com/emea/en/nao. [Accessed 27 2 2022].

[20] "robosavvy," [Online]. Available: https://forum.robosavvy.com/viewtopic.php?t=7670. [Accessed 22 05 2022].

[21] D. G.-V. H.-P. Blazevic, "Mechatronic design of NAO humanoid," pp. 770-771.

[22] "cs.cmu.edu," [Online]. Available: http://www.cs.cmu.edu/~cga/nao/doc/reference-documentation/nao/hardware/fsr.html. [Accessed 11 3 2022].

[23] D. P. &. T. K. Haggerty, "Interface System for NAO Robots," pp. 10-11, 2015.

[24] "aldebran," aldebaran, [Online]. Available: http://doc.aldebaran.com/2-1/index.html. [Accessed 2 3 2022].

[25] M. R. I.-M. R.-M. S. Miah, "NAO robot for cooperative rehabilitation training," pp. 3-4.

[26] "cmu.edu," [Online]. Available: http://www.cs.cmu.edu/~cga/nao/doc/reference-documentation/nao/hardware/kinematics/nao-joints-40.html. [Accessed 10 05 2022].

[27] "microchip," [Online]. Available: https://www.microchip.com/design-centers/16-bit/. [Accessed 24 5 2022].

[28] "wikipedia," [Online]. Available: https://en.wikipedia.org/wiki/ARM_architecture_family. [Accessed 10 5 2022].

[29] "wikipedia," wikipedia, [Online]. Available: https://en.wikipedia.org/wiki/Zero_moment_point. [Accessed 6 5 2022].

[30] "scaron," scaron, [Online]. Available: https://scaron.info/robot-locomotion/linear-inverted-pendulum-model.html. [Accessed 6 5 2022].

[31] J. S. W. Jason Kulk, "A low power walk for the NAO robot," *reaserchgate,* 2014.

[32] E. Pot, J. Monceaux, R. Gelin and B. Maisonnier, Graphical Tool for Humanoid Robot Programming, ieee.

[33] "aldebaran," [Online]. Available: http://doc.aldebaran.com/2-5/index_dev_guide.html. [Accessed 28 2 2022].

[34] "udelft," [Online]. Available: https://ii.tudelft.nl/naodoc/site_en/reddoc/framework/broker.. [Accessed 11 3 2022].

[35] "aldebaran," [Online]. Available: http://doc.aldebaran.com/1-14/dev/naoqi/index.html. [Accessed 11 3 2022].

[36] "Webots," cyberbotics, [Online]. Available: https://cyberbotics.com/. [Accessed 16 5 2022].

[37] "Webots for nao," 10 2012.

[38] "github," [Online]. Available: https://github.com/cyberbotics/naoqisim. [Accessed 16 5 2022].

[39] "webots," cyberbotics, [Online]. Available: https://cyberbotics.com/doc/guide/nao#nao-models. [Accessed 16 5 2022].

[40] "MASTER INTELLIGENCE ECONOMIQUE ET STRATÉGIES COMPÉTITIVES," [Online]. Available: https://master-iesc-angers.com/artificial-intelligence-machine-learning-and-deep-learning-same-context-different-concepts/. [Accessed 15 4 2022].

[41] "medium," [Online]. Available: https://medium.com/@sakshisingh_43965/biological-artificial-neural-network-471722148217. [Accessed 14 4 2022].

[42] "analyticssteps," [Online]. Available: https://www.analyticssteps.com/blogs/hand-gesture-classification-using-deep-learning-keras. [Accessed 31 4 2022].

[43] "introduction to newor," [Online]. Available: http://web.pdx.edu › week7b-neuralnetwork. [Accessed 26 5 2022].

[44] Y. B. Yann LeCun, "deep learning," p. 1, 2015.

[45] "simplilearn," [Online]. Available: https://www.simplilearn.com/tutorials/deep-learning-tutorial/what-is-deep-learning#how_does_deep_learning_work. [Accessed 22 5 2022].

[46] R. Krishna, C O M P U T E R V I S I O N F O U N D AT I O N S A N D, stanford university, 2017.

[47] "worldoftest," [Online]. Available: https://www.worldoftest.com/vision-measuring-machine#:~:text=Optical%20vision%20measuring%20machine%20(Abbreviation,%2C%20angle%2C%20circle%2C%20ellipse%2C. [Accessed 26 5 2022].

[48] T. G. S´anchez, "Artificial Vision," pp. 17-18.

[49] C. Rasche, "Computer Vision An Overview For Enthusiasts," 2022.

[50] Z. Zhou, "THE NAO ROBOT AS A PERSONAL," pp. 16-18, 2016.

[51] N. Arora, "Automatic Speech Recognition System: A Review," pp. 25-26.

[52] [Online]. Available: http://doc.aldebaran.com/1-14/family/robots/sonar_robot.html. [Accessed 13 3 2022].

[53] T. G. S´anchez, "Artificial Vision in the Nao Humanoid Robot," pp. 38-49, 2009.

[54] "javapint," [Online]. Available: https://www.javatpoint.com/arduino-ide. [Accessed 9 3 2022].

[55] "json," [Online]. Available: https://www.json.org/json-en.html. [Accessed 1 6 2022].

[56] "wikipedia," [Online]. Available: https://en.wikipedia.org/wiki/Python_(programming_language)#:~:text=Python%20was%20conceived%20in%20the,implementation%20began%20in%20December%201989.. [Accessed 10 3 2022].

[57] "flask," [Online]. Available: https://flask.palletsprojects.com/en/2.1.x/tutorial/templates/#:~:text=Templates%20are%20files%20that%20contain,display%20in%20the%20user's%20browser.. [Accessed 9 3 2022].

[58] "medium.," [Online]. Available: https://medium.com/hackernoon/python-flask-data-visualization-interactive-maps-30bc4e872a14. [Accessed 10 3 2022].

[59] "activestate," [Online]. Available: https://www.activestate.com/resources/quick-reads/what-is-matplotlib-in-python-how-to-use-it-for-plotting/. [Accessed 16 5 2022].

[60] "towardsdatascience," [Online]. Available: https://towardsdatascience.com/tensorflow-the-core-concepts-1776ea1732fa. [Accessed 14 5 2022].

[61] "xenonstack," [Online]. Available: https://www.xenonstack.com/blog/tensorflow. [Accessed 14 5 2022].

[62] "tensorflow," [Online]. Available: https://blog.tensorflow.org/2019/01/whats-coming-in-tensorflow-2-0.html. [Accessed 15 5 2022].