TecnoCampus
Escola Superior
Politècnica

Centre adscrit a la
upf. Universitat
Pompeu Fabra
Barcelona

Grau en Enginyeria Informàtica de Gestió i Sistemes d'Informació

QPIAA

(Quadruped Personal IA Assistant)

Memòria Final

Marc Ribas Gonzalez
TUTOR: Pere Barberan Agut
4t CURS ACADÈMIC

TecnoCampus
Mataró-Maresme

Abstract

The project consists of the integration of new software in a robotic kit for Raspberry Pi, develops three functionalities to integrate, facial recognition, speech recognition and natural language processing (text to speech) in order to study whether it is possible with accessible and free resources. Before integrating these features, the camera is replaced by a webcam that includes a microphone and speakers are added. During the development, both hardware and software unforeseen events have arisen, resulting in an unfinished project due to lack of resources.

Resum

El projecte consisteix en la integració de nou software en un kit robòtic per a Raspberry Pi, es desenvolupen tres funcionalitats a integrar, reconeixement facial, reconeixement de veu i processament de llenguatge natural (text a veu) amb l'objectiu d'estudiar si és possible amb els recursos accessibles i gratuïts. Abans d'integrar aquestes funcionalitats, se substitueix la càmera per una webcam que inclou micròfon i s'afegeix uns altaveus en el quadruped. Durant el desenvolupament han sorgit imprevistos tant a nivell hardware com a nivell software, resultant un projecte inacabat per manca de recursos.

Resumen

El proyecto consiste en la integración de nuevo software en un kit robótico para Raspberry Pi, se desarrollan tres funcionalidades a integrar, reconocimiento facial, reconocimiento de voz y procesamiento de lenguaje natural (texto a voz) con el objetivo de estudiar si es posible con los recursos accesibles y gratuitos. Antes de integrar estas funcionalidades, se sustituye la cámara por una webcam que incluye micrófono y se añaden unos altavoces en el cuadrupedo. Durante el desarrollo han surgido imprevistos tanto a nivel hardware como a nivel software, resultando un proyecto inacabado por carencia de recursos.

Índex

Índex d'il·lustracions

Índex de taules

1. Introducció

En els darrers anys, s'ha notat un increment quantitatiu de la presència de drons al mateix temps que de vehicles terrestres, aquàtics i aeris no tripulats.

Aquest fet, ha desencadenat que s'hagin popularitzat en tots els àmbits i que avui dia, l'ús de la intel·ligència artificial els hagi dotat de més autonomia i d'optimització en el seus propòsits.

Alhora també aquestes noves tecnologies que s'utilitzen per aquests autòmats són cada cop més accessibles al públic. És per això que existeixen una gran varietat de microcontroladors i ordinadors de placa simple que, aficionats del món tecnològic, amb les quals innoven i creen nous projectes afegint altres dispositius a aquests ordinadors petits.

És en aquest context que la motivació per aquest projecte és el de trobar els límits, impediments o problemes que poden sorgir, i la tecnologia disponible a les nostres mans per poder estudiar si és viable la creació d'autòmats intel·ligents.

L'objectiu d'aquest projecte és integrar nou software a un kit robòtic per a "nens" per replicar o imitar autòmats comercialitzats actuals.

Es desenvoluparà una primera idea de robot d'assistència personal intel·ligent, que l'ús d'aquest és dirigit per usuaris sense cap coneixement tecnològic i/o informàtic, mitjançant la veu.

Com a base d'un kit quadrúpede robot de Raspberry Pi per a nens o adults, es muntarà i s'adaptarà el software perquè pugui reconèixer facialment a l'usuari, executar ordres i respondre via veu.

Durant el desenvolupament es resoldran els problemes, s'analitzarà i implementarà nou software i hardware del proporcionat pel fabricant.

Finalment, es conclou si el hardware i el software gratuït que és accessible pel públic, podria assolir el mateix nivell de funcionalitats i prestacions que els actuals autòmats comercialitzats.

2. Marc teòric i anàlisis de referents

2.1. Context

Durant aquesta última dècada l'evolució tecnològica ha sigut cada cop més exponencial resultant en el desenvolupament de moltes eines hardware i software.

Aquesta evolució de hardware i software constant no tan sols ha beneficiat en la vida quotidiana, en el treball, la medicina, etc. sinó que ha facilitat molt més l'aprenentatge i la creativitat a moltes persones.

Un exemple de dispositiu que permet a qualsevol usuari aprendre i explotar la seva creativitat tecnològica són els SBC (Single Board Computer), ordinadors de placa única.

La Raspberry Pi[1] és un exemple de SBC, i la més popular actualment, desenvolupat al Regne Unit per la fundació Raspberry Pi Foundation amb l'objectiu de que qualsevol persona pugui accedir a la informàtica i la creació digital.

El model original buscava la promoció de l'ensenyament d'informàtica a les escoles, aquest va acabar sent més popular del que s'esperava, fins i tot venent-se fora del mercat objectiu per a usos com la robòtica.

És un producte amb propietat registrada, mantenint el control de la plataforma, però permetent el seu ús lliure tant a nivell educatiu com a particular. En canvi, el software sí que és de codi obert, sent el seu sistema operatiu oficial una versió adaptada de Debian, anomenada Raspberry Pi OS, encara que permet usar altres sistemes operatius.

L'organització darrere de la Raspberry Pi està formada per dos fundacions. Els primers models van ser desenvolupats per la Raspberry Pi Foundation. Després que la Raspberry Pi 1 Model B fos llançat, la fundació va crear Raspberry Pi Trading, amb Eben Upton com a CEO, per desenvolupar el tercer model, el Raspberry Pi Model 1 B+.

Raspberry Pi Trading és responsable de desenvolupar la tecnologia, mentre que la fundació és una organització sense ànim de lucre educativa que té com a objectiu promocionar l'ensenyament d'informàtica a escoles i països en desenvolupament.

Els primers models de Raspberry Pi van ser dissenyats en 2006, basats en microcontroladors de Atmel (ATmegas644). En maig de 2009, la fundació Raspberry Pi va ser fundada a Caldecote,

South Cambridgeshire, Regne Unit com a una associació caritativa que és regulada per la comissió de caritat de Inglaterra i Gales.

La primera fabricació de les plaques Raspberry Pi de prova va ser durant el 2011 i finalment el 29 de febrer de 2012 van començar les vendes del primer model.

La Raspberry Pi te moltes possibilitats d'ús, el més comú i el que al grau d'Enginyeria Informàtica del TecnoCampus forma, és la possibilitat didàctica que permet de manera fàcil programar amb aquesta eina sensors, leds, servo motors i tota mena de components.

Però no tant sols serveix pel sector de l'ensenyament , sinó que es pot aplicar a molt mes usos[2], com per exemple com a un servidor d'impressió inalàmbric que permet noves funcionalitats a les impressores antigues o sense capacitat de connexió. També permet convertir una televisió antiga en una Smart TV.

Les avantatges principals que fan de la Raspberry Pi tan exitosa són el seu baix preu, nombroses comunitats de suport, moltes idees de projectes Raspberry Pi i conjunts d'accessoris.

Des de llocs web com Raspberry Pi Foundation fins a Raspberry Pi subreddit, hi ha una gran quantitat de recursos creats per les comunitats de suport. Això inclou tutorials, revisions i més, cosa que simplifica enormement la resolució de problemes.

No hi ha competència quan es tracta d'accessoris Raspberry Pi disponibles. Es troba tot, des d'opcions de la càmera a dispositius de pantalla tàctil, i un munt de casos. Com per exemple components més específics com el Pi Hat que fan de la Raspberry Pi un dispositiu adaptable a més dispositius o components.

La Raspberry Pi presenta moltes opcions de compra, des de la placa independent fins a kits complets. Els seus pins GPIO fan del Pi un dispositiu utilitari.[24]

A causa de les seves opcions de hardware i software, hi ha molts usos. Des d'un servidor a l'automatització de la llar , Pi és immensament versàtil.

Per començar, hi ha molts projectes de Raspberry Pi per a principiants que són pràctics i divertits, cosa que no es pot dir exactament d'altres competidors.

Un exemple de projecte és construir una màquina de arcade emulant les consoles retro com la GameBoy gràcies a sistemes operatius dedicats exclusivament a l'emulació d'antigues videoconsoles.[25]

Però el més interessant i que interessa més per aquest projecte, es el seu ús com a controlador per a robòtica.

La versatilitat i el gran suport que hi ha darrere de les aplicacions per a programació de robòtica en Raspberry Pi converteixen aquesta plataforma, juntament amb Arduino, en una de les preferides pels aficionats a la programació d'autòmats.

L'ús d'aquests petits ordinadors s'ha estès tant que es poden trobar kits complets de robots controlats mitjançant una Raspberry Pi.

Existeixen tot tipus de kits robòtics, com cotxes robots, robots quadrúpedes o hexàpodes, braços robòtics o fins hi tot drons i la majoria amb sensors, càmeres i altres components que exploten tot el seu potencial.

2.2. Antecedents

Com a alternatives d'ordinadors de placa simple podem trobar varis[3], com el Tinker Board S de ASUS, es una molt bona alternativa a Raspberry Pi, amb connector de 40 pins, un processador de 4 nuclis i 2GB de memòria RAM. Admet els sistemes operatius Retropie, qualsevol distribució Linux i Android.

Un altra alternativa igual de bona és la placa LePotato, amb processador de 4 nuclis i amb 1GB o 2GB de memòria RAM. Es compatible amb Android com a sistema operatiu i també amb qualsevol versió Linux. Com a punts en contra no compte amb tecnologia WiFi ni tecnologia Bluetooth.

Jetson Nano és una alternativa per a Raspberry Pi una mica especial, ja que es pot utilitzar per a diversos tipus de tasques, però té un mòdul de memòria on pot incorporar ports d'expansió. Pel que fa al processador tenim un ARM Tall A57 de 4 nuclis juntament amb una memòria RAM de 4 GB DDR4. La part més positiva és la incorporació d'una gràfica NvidiaMaxwell que fa que tinguem un apartat gràfic molt bo on gaudirem d'un nivell de visualització molt més gran del que pot oferir la Raspberry Pi en tot tipus de situacions i en tot tipus de paràmetres. Un punt negatiu és que no té cap mòdul WiFi i tan sols disposem de connexió via Ethernet i el software

que ja ve incorporat es un sistema operatiu Linux amb una sèrie d'aplicacions de la mateixa manera que la Raspberry.

La majoria dels kits robòtics que es comercialitzen són dissenyats per funcionar amb Arduino o Raspberry Pi. Aquests kits es venen complets amb tots els components que es necessiten menys el controlador Raspberry o Arduino i amb un software de codi obert desenvolupat perquè un cop muntat el robot ofereixi unes funcionalitats bàsiques i es pugui modificar.

Per aquells que s'inicien en el món tecnològic o de programació hi han empreses que inclouen guies i documents perquè puguin aprendre més fàcilment.

Actualment, en venda hi han molts tipus de kits diferents i comercialitzats per diferents empreses, com a exemple el RaspClaws Hexápode**[4]** de l'empresa Adeept, es un kit aranya amb sensors per estabilitzar-se i càmera.

També hi són disponibles tancs o cotxes robot, com el cotxe robot Mecanum Wheels Robot Car**[5]** dissenyat per l'empresa OSOYOO on aquest cas s'utilitza l'Arduino com a controlador o el tanc robot RaspTank**[6]** de Adeept amb un braç robòtic i com a controlador una Raspberry Pi.

No és realment necessari comprar un d'aquests kits per a poder construir el teu propi robot sinó que un mateix se'l pot fer comprant per separat els components necessaris i amb una impressora 3D crear les peces necessàries per l'estructura.

Es el cas d'un Robot assistent personal que segueix a la persona i troba objectes creat i dissenyat per Saral Tayal**[7]**, de fet aquest projecte personal que ha compartit sense ànim de lucre és la inspiració d'aquest projecte. També basat amb el controlador Raspberry Pi aquest projecte permet interactuar per veu i està dotat de sensors i una càmera pel reconeixement d'objectes i facial.

2.3. Necessitats d’informació

Per al desenvolupament del projecte, primer de tot calen els coneixements necessaris per muntar el kit robòtic que hem escollit. Es poden obtenir els coneixements necessaris gràcies a la guia i documents que el fabricant ens facilita en la seva pàgina web**[8]**.

Altres alternatives de fonts d'informació són el canal de Youtube SSTec Tutorials**[10]** o Heap Art Coding**[11]**.

La duració de l'adquisició d'aquests coneixements son d'una hora aproximadament i sense cap coneixement previ de la matèria.

Previament necessitarem coneixements de programació en Python, aquests es poden obtenir cursant l'assignatura de Tècniques d'Intel·ligència Artificial o Màrqueting del grau en Enginyeria Informàtica del TecnoCampus de Mataró.

Una altre font seria els cursos Python que l'empresa fabricant del kit robòtic facilita amb la documentació.

La duració per adquirir els coneixements de programació en Python són de sis mesos i com a requisit previ és necessari coneixements en programació en qualsevol llenguatge, preferiblement en Arduino.

Seguidament per a poder treballar i desenvolupar amb el sistema operatiu de Raspberry Pi serien necessaris coneixements en el funcionament del sistema Linux. Es pot obtenir amb el curs certificat LPI Linux Essentials amb l'instructor Antonio Sánchez Corbalán - LPI Training Partner**[9]** i els seus cursos amb temari oficial.

El temps a invertir amb la formació és aproximadament d'un mes i no es necessiten coneixements previs.

Finalment, serà necessari coneixements per manipular i/o modificar el hardware base en què partim, en aquest cas el kit robòtic.

Per part de l'usuari, els coneixements necessaris per l'ús del producte es poden obtenir amb la guia subministrada a l'usuari amb la compra del robot i un cop interactuï amb el producte. El temps emprat per adquirir aquests coneixements són d'uns deu minuts aproximadament.

A continuació es mostrarà de forma esquematitza les tasques i les seves necessitats:

Desenvolupament del robot assistent personal.

Coneixements de muntatge del kit robòtic

Forma d'obtenció	Es pot obtenir dels documents i guies que la pròpia empresa del kit facilita. També es pot obtenir coneixement d'altres fonts, com el canal de Youtube de SSTec Tutorials o Heap Art Coding.
Duració	Una hora aproximadament.
Enfocament	Específic.
Coneixement previ del tema	Cap coneixement previ.

Taula 2.3.1 - Coneixements de muntatge del kit robòtic. *Font: Elaboració pròpia*

Coneixements en desenvolupament del software

Forma d'obtenció	Es pot obtenir cursant l'assignatura de Tècniques d'Intel·ligència Artificial o Màrqueting on s'aprèn a programar en Python. Una altre font seria els cursos Python que l'empresa del kit robòtic facilita amb la documentació que facilita.
Duració	6 mesos
Enfocament	General
Coneixement previ del tema	Coneixement en programació en qualsevol llenguatge. Preferiblement en Arduino.

Taula 2.3.2 - Coneixements en desenvolupament del software. *Font: Elaboració pròpia*

Coneixements en el funcionament del sistema Linux

Forma d'obtenció	Es pot obtenir amb el curs certificat LPI Linux Essentials amb el instructor Antonio Sánchez Corbalán - LPI Training Partner i els seus cursos amb temari oficial.
Duració	Un mes aproximadament
Enfocament	General
Coneixement previ del tema	Cap coneixement previ.

Taula 2.3.3 - Coneixements en el funcionament del sistema Linux. Font: Elaboració pròpia

Usuari del robot.

Coneixements de l'ús i funcionament del producte final

Forma d'obtenció	Es poden obtenir amb la guia suministrada al usuari amb la compra del robot i un cop interactui amb el producte.
Duració	Deu minuts aproximadament
Enfocament	General
Coneixement previ del tema	Cap coneixement previ.

Taula 2.3.4 - Coneixements de l'ús i funcionament del producte final.

Font: Elaboració pròpia

3. Objectius i Abast

3.1. Objectius del producte

Primer definim els objectius generals a complir, després es donarà més detall d'aquests i els dividirem en curt, mitjà o llarg termini.

Objectius generals:

- Reconèixer, identificar i recordar persones i usuaris.
- Captar i reconèixer paraules i frases.
- Vigilar la llar de intrusos.
- Poder comunicar via veu.

Objectius mínims del producte:

A curt termini (2 mesos de desenvolupament):

- Capturar i reconèixer ordres donades via veu.
- Reconèixer i identificar el rostre de cada usuari que coneix o persona que interactuï.
- Mantenir una base de dades per a un cop identificat un rostre saber amb qui està parlant o rebent les ordres.
- Mantenir una conversa senzilla amb el usuari.
- Cercar objectes mitjançant la càmera i senyalitzar-los amb el LED frontal un cop trobats via computació de vídeo i machine learning.
- Executar les ordres donades de moviment senzilles, com anar endavant, enrere o girar.

Objectius un cop assolits els minims:

A mitjà termini (1 mes de desenvolupament)

- Canviar a mode vigilància per a la llar. Serà capaç de detectar si les persones o usuaris que hi son dins la llar, un cop activat el mode, després de registrades a la base de dades, si són intruses o no. Si són intruses donar una alarma en forma de notificació per l'aplicació web i emetre un so d'alarma amb leds parpellejant en vermell.

A llarg termini (2 mesos de desenvolupament)

- Via comandes de veu, capaç de configurar alarmes, recordatoris i una agenda.

3.2. Objectius pel client

- Interactuar per via veu amb el robot.
- Donar ordres de moviment.
- Accedir a les imatges en temps real de la càmera frontal.
- Registrar altres persones com a amics de l'usuari.
- Registrar a altres persones com a usuaris.
- Registrar a persones o usuaris amb identificació facial i el seu nom.
- Habilitar el mode de vigilància.

3.3. Públic potencial

En primer grau:

- Públic en general sense que sigui necessari cap coneixement informàtic o tecnològic.
- Edat mínima de 6 anys sense cap màxim d'edat.
- Que disposin d'accés a internet al seu domicili.
- Amb mòbil intel·ligent propi.

Segon grau:

- Entusiastes de la tecnologia

- Estudiants de robòtica, informàtica o qualsevol estudiant del món de la tecnologia que vol adquirir nous coneixements o experimentar.

4. Metodologia

Per tal de poder portar a terme aquest projecte i efectuar les entregues en el temps estimat de sis mesos, es realitzarà una divisió del mateix en cinc fases. No es tracta d'un procés Waterfall o en cascada, ja que en cada un dels cicles es replantejarà la feina de les fases anteriors o es repetiran algunes fases.

El projecte es divideix en diferents fases, la inicial on es farà un primer anàlisis i recerca d'informació, la segona un anàlisis de com executar el desenvolupament i un primer disseny d'aquest.

La tercera fase es desenvolupa el software i la documentació tècnica en paral·lel al desenvolupar el codi. En la quarta fase es realitza el desenvolupament dels escenaris que un client fictici es podrà trobar.

Com a fase final un anàlisis dels objectius i documentació final.

1. Fase Inicial:

Durant aquest període es portarà a terme la recerca de la informació necessària per al desenvolupament del projecte. Cal esmentar que prèviament s'ha fet un petit anàlisis de la tria del kit robòtic i compra dels components necessaris.

En ella es definirà:

- Objecte del Projecte

- Estudi Previ: context, antecedents i necessitats d'informació

- Objectius i Abast

- Metodologia

- Definició dels requisits funcionals i tecnològics

- Estudi de viabilitat del projecte: planificació, pressupost i anàlisis de viabilitat tècnica, econòmica, mediambiental i legal.

2. Fase d'anàlisi i disseny:

En aquesta segona fase s'analitza els objectius a assolir i es fa la captura dels requisits del projecte i s'especifiquen els casos d'ús.

Un cop analitzats es realitza el disseny dels diferents elements que componen el projecte:

- El disseny dels diferents mòduls software que es desenvolupen, disseny de la integració de nou hardware al robot i el disseny del software que s'integra amb el ja existent.

- L'anàlisis i el disseny serà documentat en paral·lel per mantenir la traçabilitat.

3. Fase de desenvolupament:

S'inicia la tercera fase pel desenvolupament del software per mòduls i es conclou quan s'integra tot el software amb el ja existent i el desenvolupat.

En paral·lel es dura a terme la documentació tècnica per a no perdre cap detall del procés.

4. Fase de proves:

En aquesta fase es pensa i es plasma escenaris que un client fictici es pot trobar quan s'utilitza el producte.

S'analitza els resultats obtinguts, es documentarà i es millora el producte.

5. Fase de d'anàlisis dels objectius i documentació final:

Un cop finalitzada les proves es realitza la anàlisis del que s'ha obtingut i els objectius que són treballats, integrarem tots els documents realitzats i realitzarem els retocs finals a tots els documents a lliurar.

5. Desenvolupament

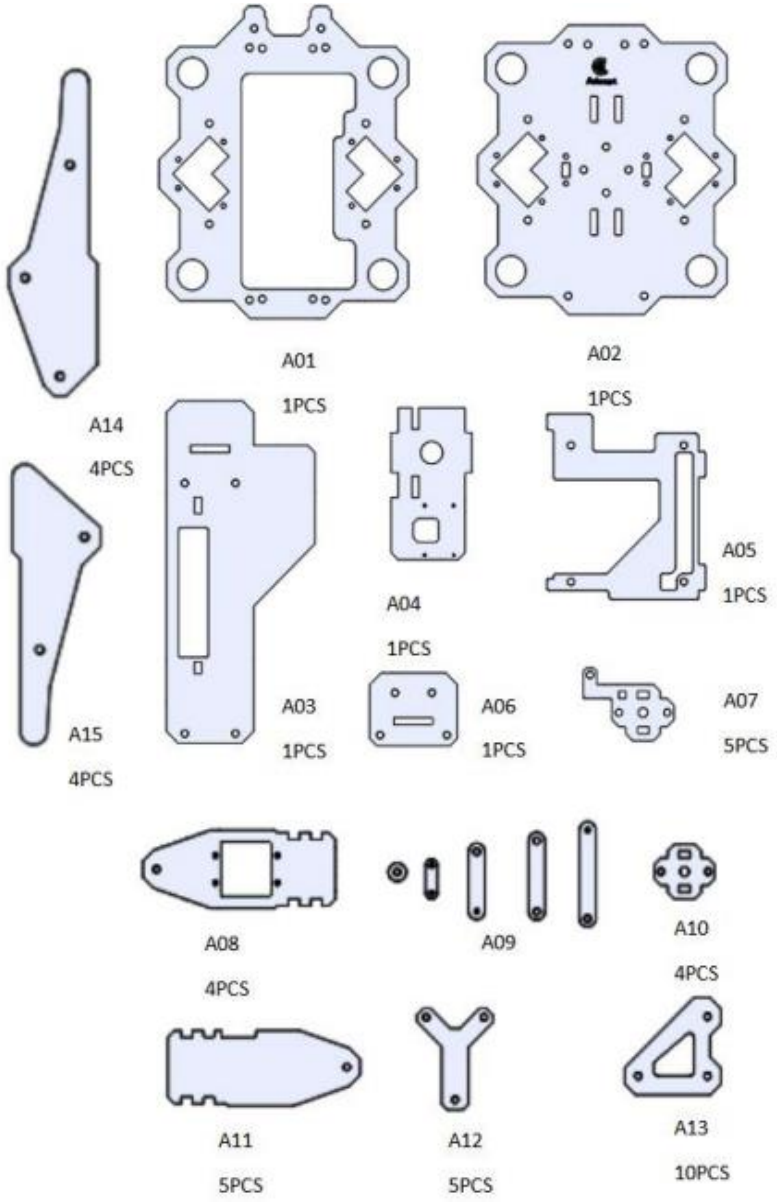
El desenvolupament és divideix en dos nivells. A nivell hardware no s'entra a gran detall, s'explica que s'ha fet i els problemes trobats. A nivell de software s'explicarà més en detall les solucions proposades i les implementades per assolir els objectius.

5.1. Hardware

5.1.1. Muntatge i components del robot

Components de la estructura del robot:

Aquestes imatges son extretes de la documentació facilitada pel fabricant.



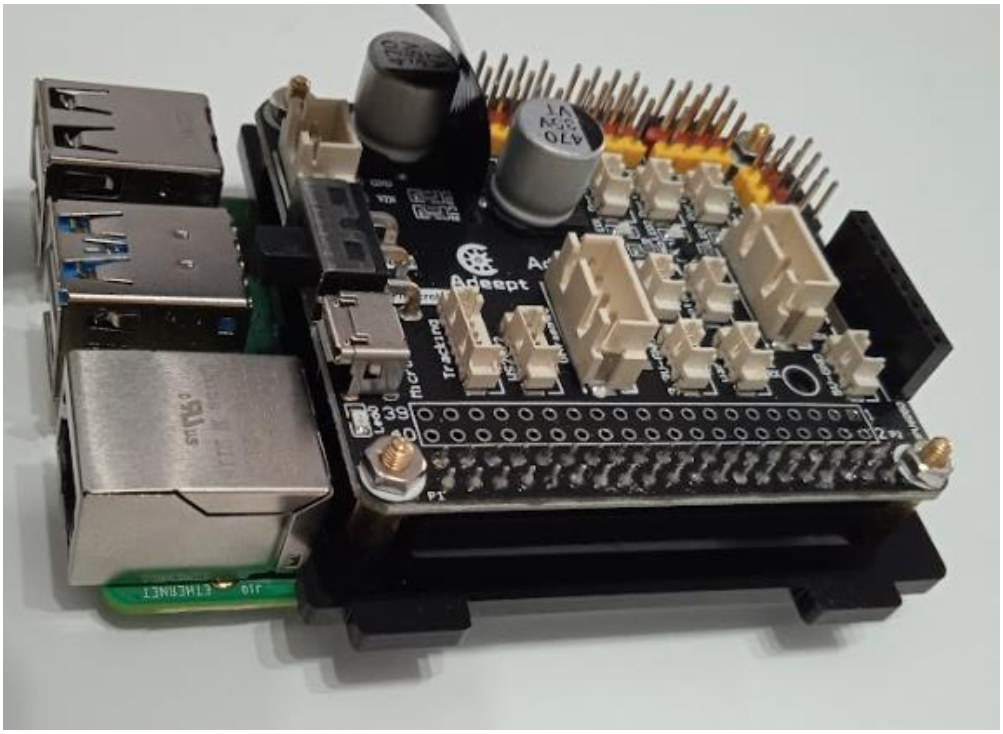
Il·lustració 5.1.1.1 - Representació gràfica de les peces necessàries pel muntatge. Font: Product List - Adeept

En aquesta il·lustració es veu la forma de les peces de material acrílic, un material no molt resistent, tal com es comenta en la documentació del fabricant. Per a cada peça en el dibuix veiem escrit XXPCS on XX son el número de peces que hi ha d'aquest tipus.

Una gran quantitat de cargols i volanderes de goma i metall venen incloses pel muntatge de les peces. També inclou eines necessàries com tornavisos i altres. No es detalla molt aquests components, ja que no és l'objectiu del projecte.

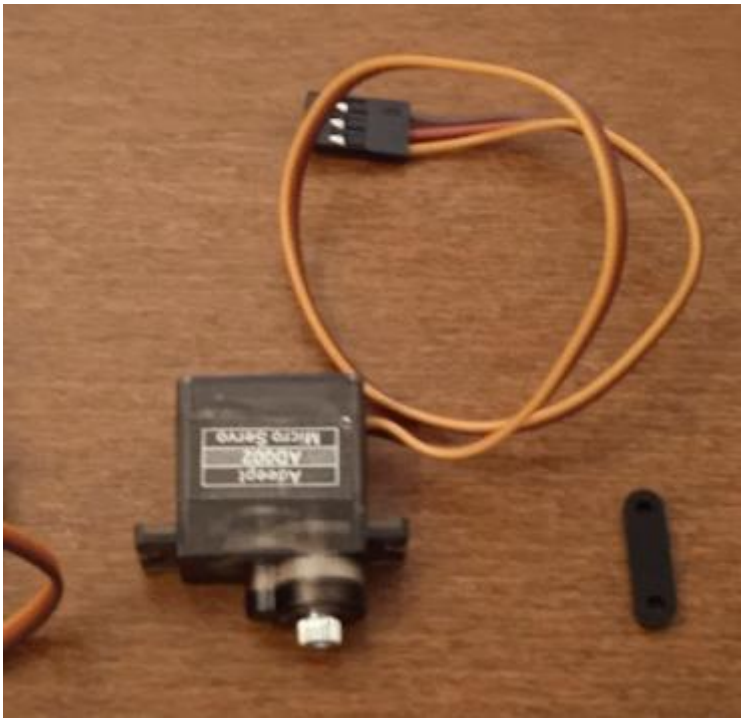
Components elèctrics:

Robot HAT: com diuen les seves sigles ,**Hardware Attached on Top**, es una placa rectangular que s'acobla amb els quatre orificis que té. Es connecta i s'alimenta via els capçals GPIO de 40w. Aquest dispositiu serveix de controlador de totes les peces del quadrúpede via configuració dels GPIO.



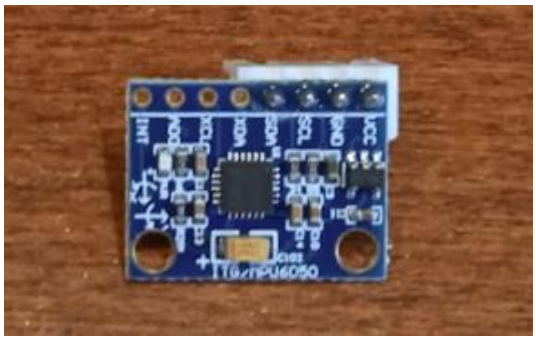
Il·lustració 5.1.1.2 - Robot HAT. Font: Elaboració pròpia

Servomotor: és un dispositiu similar al motor, però amb la capacitat de ubicar-se en una posició dins del rang d'operació i mantenir-se estable.



Il·lustració 5.1.1.3 - Fotografía d'un servomotor de Adeept. Font: Elaboració pròpia

MPU-6050: El MPU6050 és un Micro Electro-Mechanical Systems (MEMS) que consta d'un acceleròmetre de 3 eixos i un giroscopi de 3 eixos al seu interior. Això ens ajuda a mesurar l'acceleració, la velocitat, l'orientació, el desplaçament i molts altres paràmetres relacionats amb el moviment d'un sistema u objecte.

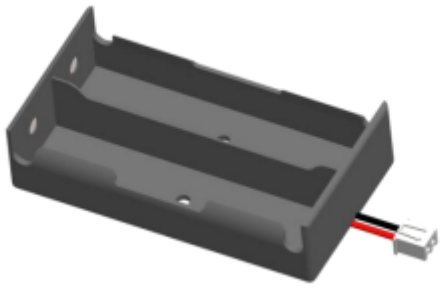


Il·lustració 5.1.1.4 - Fotografía del dispositiu MPU-6050. Font: Elaboració pròpia

Raspberry Pi camera: Admet gravació 1080p @ 30fps, 720p @ 60fps i 640x480p 60/90. Interfície sèrie de la càmera MIPI de quince pins: es connecta directament a la placa Raspberry Pi.

Il·lustració 5.1.1.5 - FotografIa de la Raspberry Pi camera . Font: Elaboració pròpia

Caixa de bateries 18650: suport per a les bateries que donaran autonomia al robot.

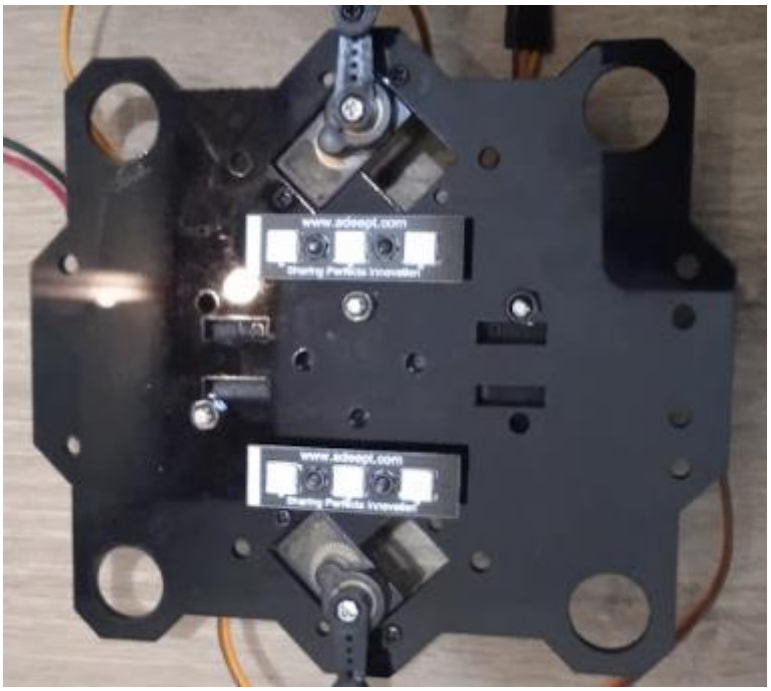


Il·lustració 5.1.1.6 - Il·lustració grafica de la caixa per a les bateries 18650. Font: Product List - Adeept

Llum led blava: led d'un sol color, blau.

Il·lustració 5.1.1.7 - Fotografia del llum led, fa una funció semblant a una llanterna. Font: Elaboració pròpia

Leds RGB: dos panells petits de tres leds multicolors.



Il·lustració 5.1.1.8 - Fotografia on es mostra els dos panells led ja muntats al cos principal. Font: Elaboració pròpia

Procés de muntatge:

Abans de muntar el fabricant ens indica que es necessari instal·lar el software de control abans a la Raspberry pi, perquè els servomotors han de ser tots configurats en la seva posició inicial, a zero graus, quan son muntats.

Il·lustració 5.1.1.9 - Components necessaris per a poder executar el software del robot. Font: Elaboració pròpia

Al realitzar aquest procés previ, no és possible executar el software del fabricant, ja que no executa el codi sense estar la càmera instal·lada al port serie de la Raspberry Pi resultant un error d'execució aturant el programa.

El port càmera de la Raspberry Pi usada es defectuosa i no dona cap senyal de vídeo.

Com a proposta es crea un senzill programa Python amb l'objectiu de configurar els servomotors segons l'angle que es vulgui donar.

El codi que es utilitzat per a continuar amb el muntatge del quadrúpede, s'usen les llibreries **time** i **Adafruit_PCA9685** utilitzada pel fabricant. Es configura la freqüència en la que es mouen a l'entrar un angle, o més ben dit la velocitat en què es mouran aquests servos.

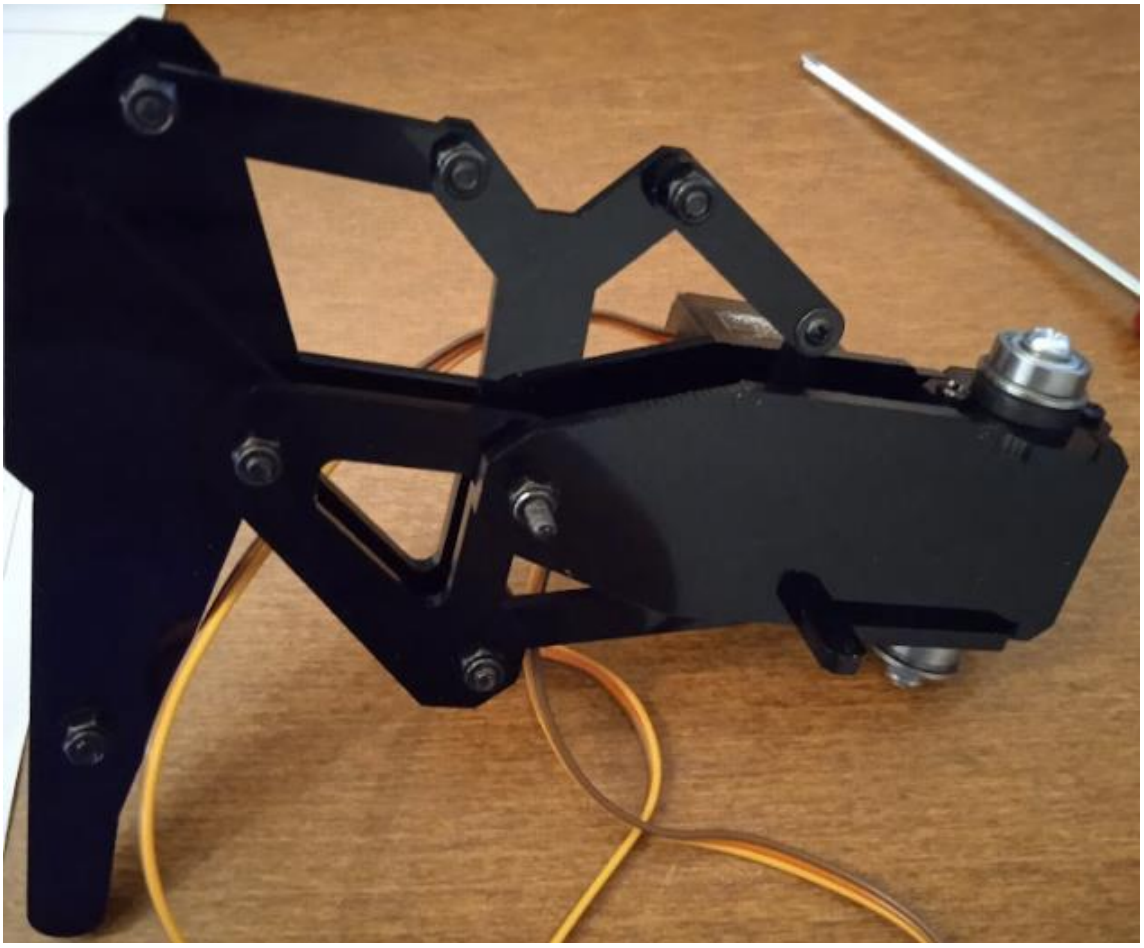
```
import time
import Adafruit_PCA9685
pwm = Adafruit_PCA9685.PCA9685()
pwm.set_pwm_freq(50)

while 1:
    wInput = input("Enter Servo No and Position : ")
    wInputArr = wInput.split()
    pwm.set_pwm(int(wInputArr[0]),0,int(wInputArr[1]))
    time.sleep(3)
```

Il·lustració 5.1.1.10 - Estructura del codi per a configurar els servomotors en la posició desitjada. Font: Elaboració pròpia

I com a pas final, dins del bucle infinit, es guarda les dades introduïdes en **wInputArr**, amb la funció **set_pwm** configura el servomotor amb el PWM especificat.

Al executar aquest codi simplement introduïda la posició del servomotor en la placa ROBOT HAT i la PWM (Pulse With Modulation) que es entre el 0 i 600, on 300 seria el punt inicial o angle zero.

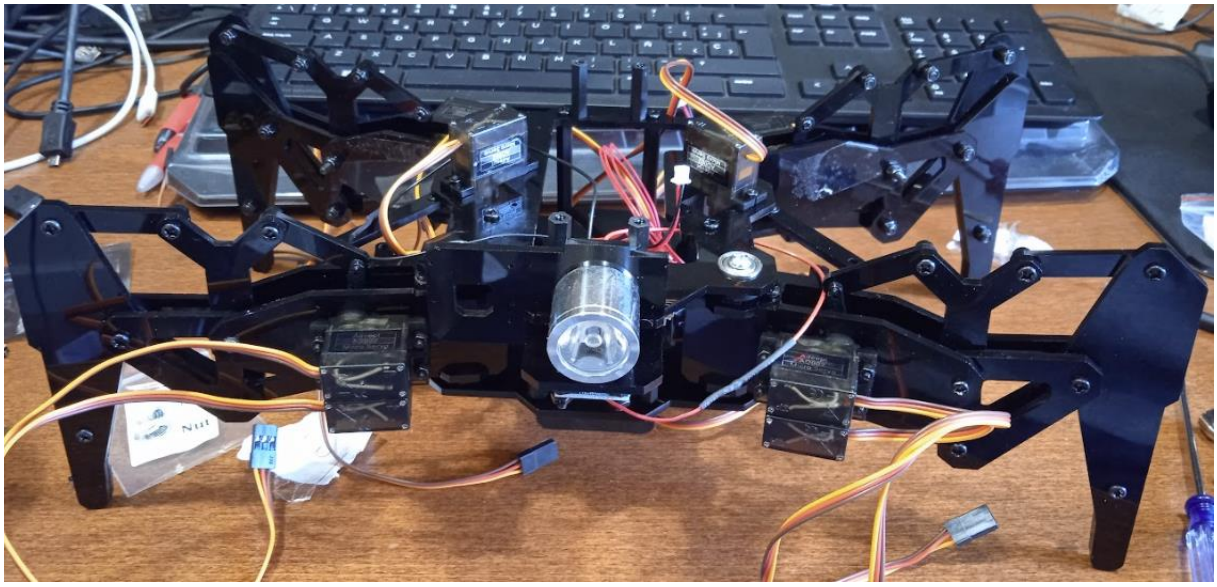
Un cop es finalitza els passos previs al muntatge, es comença per les potes del robot tal com ens indica la guia del fabricant. Com a resultat del muntatge que es representa a la següent fotografia:



Il·lustració 5.1.1.11 - Fotografia d'una pota finalitzada, on al seu costat contrari hi té els dos servomotors que permet el moviment. Font: Elaboració pròpia

Per a cada cama del quadrúpede es muntat de manera diferent, ja que aquestes hi estan connectades a un tercer servomotor instal·lat al cos. És important tenir localitzades o etiquetades les potes per saber quina pota correspon segons la seva posició.

Al finalitzar el muntatge de les potes, es continua muntant per separat el cos del robot, que està dividit en dos. Un cop acabades les dues meitats del cos, on s'instal·la més endavant la Raspberry Pi i el ROBOT HAT, es muntaran en la superfície inferior les potes amb els seus eixos i amb la part superior es tanca unint tota l'estructura. Amb un resultat com la de la imatge:

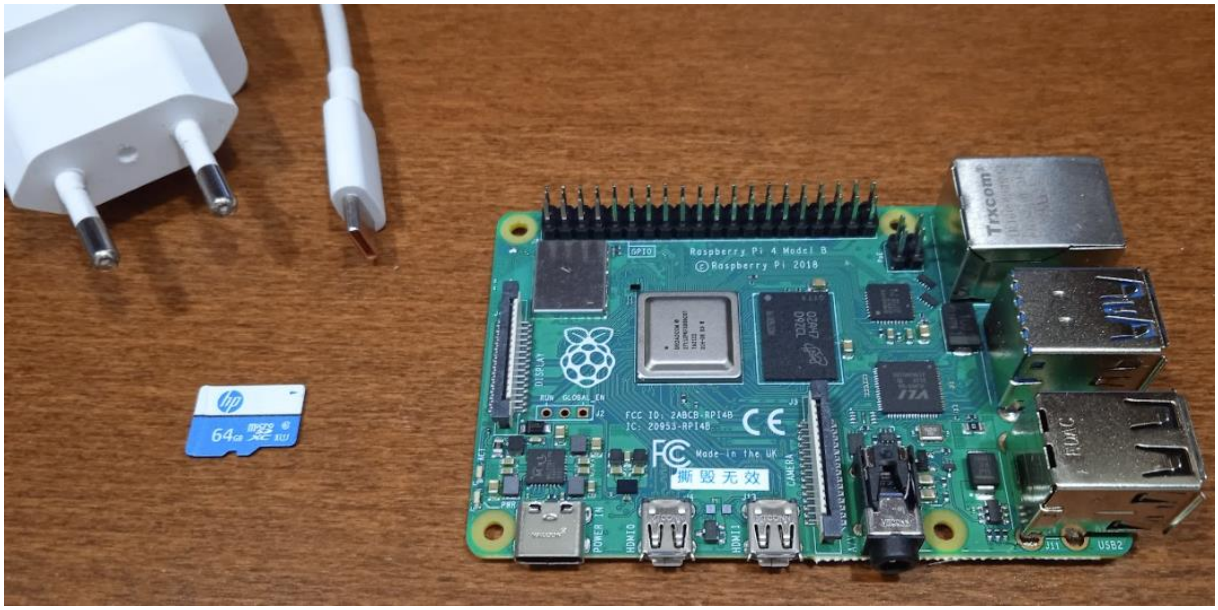


Il·lustració 5.1.1.12 - Estructura final sense el controlador i la Raspberry Pi instal·lades. Font: Elaboració pròpia

5.1.2. Hardware necessari pel kit

El kit no inclou el miniordinador necessari pel seu funcionament, la Raspberry Pi. El fabricant especifica que ha de ser la versió tres o la més actual la quatre.

En aquest cas s'usa la Raspberry Pi 4 de 8gb de RAM. També és necessari una microSD mínim de 8gb, però s'escull una de 64gb per tenir suficient espai per fer el projecte.



Il·lustració 5.1.1.13 - Components no inclosos en el kit, Raspberry Pi4 i tarjeta microSD de 64gb. Font: Elaboració pròpia

5.1.3. Hardware adicional

El hardware necessari per a poder assolir els objectius són un micròfon, o en el seu defecte una webcam, ja que un dels problemes que s'ha tingut, i es comenta seguidament, es el mal funcionament del port sèrie de la càmera de la Raspberry Pi.

S'ha escollit la webcam **Logitech C920HD Pro** amb resolució FULL HD i micròfon incorporat de bona qualitat.

Com a últim component requerit es un altaveu portàtil o capaç de ser integrat en el robot amb facilitat.

Per a realitzar les proves s'utilitzen altaveus analògics d'entrada jack, compatible amb el model de Raspberry Pi que es usat.

5.1.4. Problemes

El pas previ al muntatge, no és possible utilitzar el software del fabricant, ja que no pot executar aquest sense la càmera instal·lada i funcional.

La llibreria **opencv** no troba el dispositiu i dóna una excepció resultant una parada de l'execució.

Com a alternativa, es crea el codi Python mencionat anteriorment amb el resultat de poder configurar els servomotors al angle cero.

Per a solucionar el problema de la càmera, es va proposar provar vàries càmeres connectades al port serial, però cap sense èxit determinant que el problema provenia de la Raspberry Pi. I com a solució final es proposa canviar aquesta càmera per una connectada via USB.

5.2. Software

5.2.1 Software del fabricant

En aquest apartat s'explicarà el software que proporciona el fabricant del kit robòtic per a la seva manipulació sense entrar en un gran detall.

El fabricant ens dóna dues opcions per instal·lar el software necessari, la primera es a partir del sistema operatiu Raspbian més actual i executant el programa Python **setup.py** instal·larà totes les dependències i llibreries necessàries.

La segona opció es una imatge completa del sistema operatiu en una versió anterior ("Raspbian GNU/Linux 10 (buster)") ja integrades totes les dependències i llibreries necessàries, provada pel fabricant, per poder treballar i que és funcional.

Al provar la primera opció i executar el programa **setup.py** dona errors i les instal·lacions no són completades, ja que la majoria de llibreries i dependències pel controlador ROBOT HAT del quadrúpede no estan actualitzades.

Al contactar amb el fabricant s'aconsella utilitzar la seva imatge estable com a base per a desenvolupar i usar les seves funcionalitats.

El llistat de llibreries i paquets son el següents extret del script Python setup.py:

- python-dev
- python-pip
- libfreetype6-dev
- libjpeg-dev
- build-essential
- luma.oled
- i2c-tools
- adafruit-pca9685
- flask
- flask_cors
- websockets
- rpi_ws281x

- python3-smbus
- mpu6050-raspberrypi
- numpy
- opencv-contrib-python==3.4.3.18
- libqtgui4
- libhdf5-dev
- libhdf5-serial-dev
- libatlas-base-dev
- libjasper-dev
- libqt4-test
- imutils
- zmq
- pybase64
- psutil

El codi base del fabricant és dividit en dues carpetes, Client i server. Client conté els programes necessaris per a comunicar-se amb els del server, per a executar les funcions del robot, i aquests programes interactius des de la pàgina web que és creada.

Dins de la carpeta server, conté totes les funcions i comunicació amb els components hardware del robot. També inclou diversos tests per poder provar la funcionalitat de cada component del quadrúpede.

A continuació es descriuen els mòduls necessaris per a realitzar el projecte:

webServer: és el programa principal del fabricant que s'executa quan el sistema s'encén o es reinicia.

La funció primària d'aquest és aixecar un servidor web per rebre les comandes enviades del client web i executar les funcions de moviment, led i altres funcionalitats més complexes que són definides als mòduls responsables del control de moviment, led i càmera.

Les funcions definides més interessants pel projecte són:

- **functionSelect**. Segons la comanda que rep com a paràmetre executa els següents modes:
    - findColor: detecta el color especificat en la web app via vídeo.

    - motionGet: detecta el moviment que registre dels frames que rep del vídeo
    - police / policeOff: configura els leds RGB per a que simuli les senyals de policia
- **robotCtrl**. Aquesta funció permet tots els moviments possibles del quadrúpede que són importants del mòdul **SpiderG**. Executa els moviments especificats via paràmetre.

SpiderG: la responsabilitat d'aquest mòdul es la gestió dels servomotors i el giroscopi MPU6050 per a l'execució de les comandes de moviment i funcionalitats possibles amb els dos tipus de dispositius.

Importa les llibreries **Adafruit_PCA9685** pel control dels servomotors i **mpu6050** per la lectura dels valors del giroscopi.

Al iniciar el mòdul **SpiderG** comprova, dins d'un try/except, que els servos són connectats configurant la freqüència de moviment dels servos i també del giroscopi instanciant-lo.

Tot hi que es produís errors el programa continuaria executant-se i es configuren els dotze servomotors a la posició inicial.

A continuació es descriu breument les funcions més rellevants un cop s'inicia el programa:

- **move_init**. Configura el PWM de tots els servomotors a 300, en altres paraules a la posició inicial. Aquesta funció és obligatòria d'executar abans de donar ordres de moviment.

- **command_GenOut**. Executa els moviments o posicions especificats en una variable global **now_command,** les quals han de ser **forward, backward, turnleft, turnright i stop** com a ordres de moviment i posició **StandUp i StayLow.** Aquests moviments són ja preconfigurats en un altre funció **status_GenOut**, on configura els valors PWM als servomotors.

- **walk.** Especificant una de les ordres de moviment o posició esmentades anteriorment, es guarda dins la variable **goal_command**. Finalment, crida la funció **resume()** i substitueix el valor de **now_command** amb el valor de goal_command.

- s**ervoStop.** Atura el moviment dels servomotors.

- **steadyModeOn/Off.** Al cridar aquesta funció llegeix els valors del sensor MPU6050 i manté estable el quadrúpede executant la funció **status_GenOut** segons la inclinació.

robotLight: permet el control dels leds RGB i de la llanterna de led blava. La majoria de funcions són definides per les tires de llum led RGB i per la llanterna no, ja que és un dispositiu de funcions limitades.

L'inici d'aquest mòdul és primer configurar els valors dels leds RGB, com la brillantor, el número de leds i la freqüència de senyal led. També es configuren els ports **GPIO** que són usats lògicament, encara que són connectats al **HAT** i que el driver tradueix les senyals.

Les funcions més rellevants són:

- **RobotLight** (classe) inicia el programa configurant tots els valors necessaris pels leds.
- **setColor.** Configura els colors dels leds RGB per paràmetre.
- **police**. Al executar els llum leds simulen la senyal de policia.
- **breath**. Dona un efecte de respiració preconfigurada al cridar aquesta funció.
- **frontLight**. Encén o apaga la llanterna frontal blava del quadrúpede

Tots aquests mòduls en comú comparteixen una llibreria, menys **webServer**, **threading**. Aquesta llibreria permet executar varis programes en diferents processos i gestionar-los tot al mateix temps.

Per aquesta raó es crea un *thread* diferent per a cada funcionalitat, pel moviment (**SpiderG**), control de llum (**robotLight**) i la càmera.

5.2.2 Sistema de reconeixement facial

Per a que sigui capaç el robot d'identificar a les persones o els usuaris, s'implementa un sistema de reconeixement facial.

El reconeixement facial és una tecnologia capaç d'identificar o verificar un subjecte mitjançant una imatge, un vídeo o qualsevol element audiovisual del seu rostre. Generalment, aquesta identificació s'utilitza per accedir a una aplicació, sistema o servei.

És un mètode d'identificació biomètrica que utilitza les mesures d'aquest cos, en aquest cas la cara i el cap, per verificar la identitat d'una persona mitjançant el seu patró biomètric facial i les seves dades. La tecnologia recull un conjunt de dades biomètriques úniques de cada persona associades a la seva cara i expressió facial per identificar, verificar i/o autenticar una persona.[12]

5.2.2.1 Software proposat

Com que l'objectiu no és la creació d'un sistema complet des de zero, es usada una llibreria ja existent fàcil d'utilitzar i de màxima precisió.

Es proposa com a primera llibreria Python a implementar **face-recognition** basat amb la llibreria *dlib* i tecnologia *Deep Learning.*

Dlib és un conjunt d'eines C++ modern que conté algorismes d'aprenentatge automàtic i eines per crear programari complex en C++ per resoldre problemes del món real.

Aquesta llibreria té una precisió del 99.38% en el *benchmark* LFW (Labeled Faces in the Wild) una base de dades de fotografies facials dissenyada per estudiar el problema del reconeixement facial. El conjunt de dades conté més de 13.000 imatges de cares recollides d'internet.[13]

5.2.2.2 Requeriments

Els requeriments per l'ús d'aquest software són:

- Python 3.3+ or Python 2.7
- macOS o Linux

Tots dos requeriments es compleixen, el sistema operatiu de la Raspberry pi és una distribució Linux i el software proporcionat pel fabricant és basat amb llibreries de la versió Python 3.7.3.

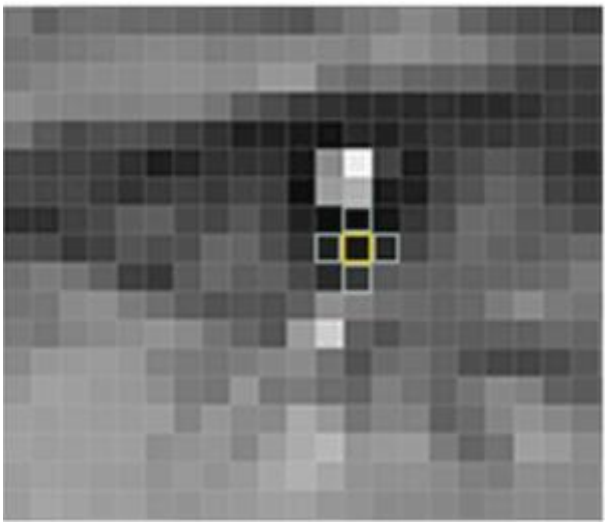
5.2.2.3 Funcionament intern de la llibreria

A continuació s'explica com aquest algoritme funciona. És dividirà en varios pasos[14]:

Pas 1 : Trobar totes les cares

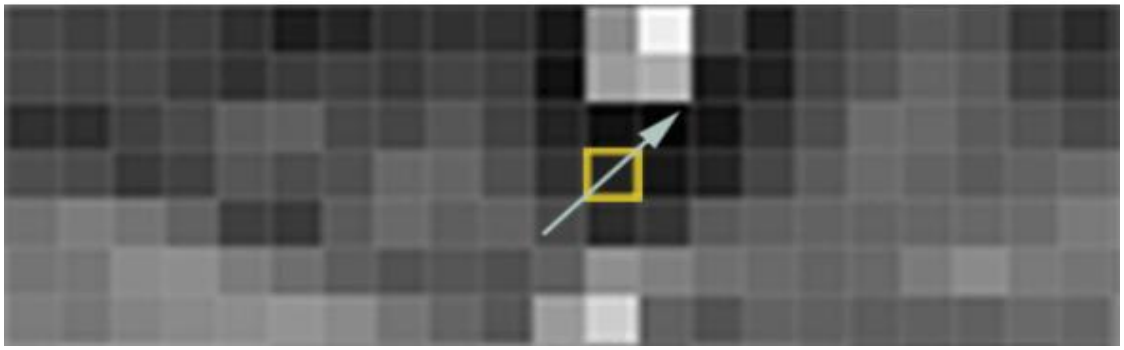
Per trobar cares en una imatge, es comença fent la imatge en blanc i negre perquè no necessitem dades de color per trobar cares.

A continuació, es veu cada píxel de la nostra imatge d'un en un. Per a cada píxel, es vol observar els píxels que l'envolten directament.



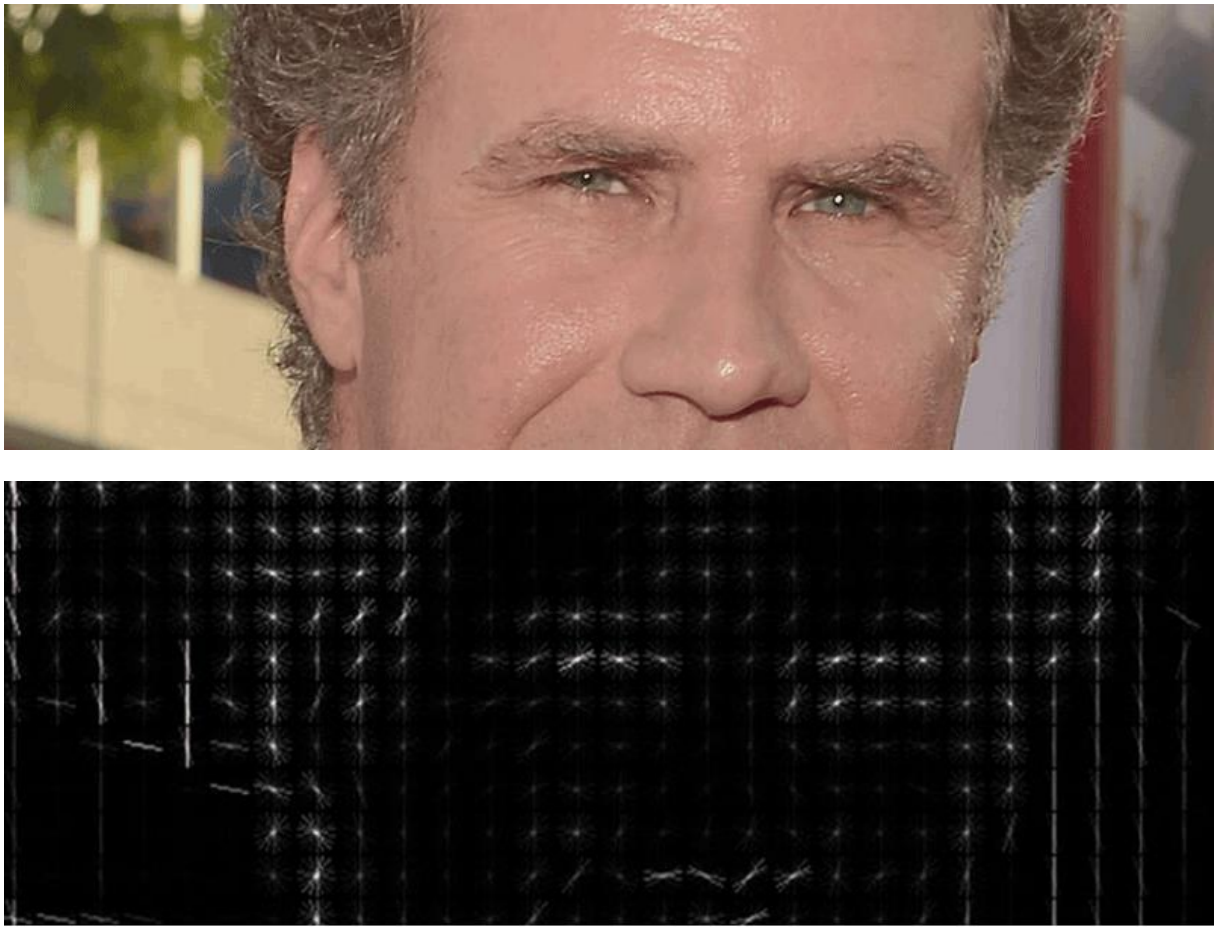
Il·lustració 5.2.2.3.1 - Representació gràfica del procés iteratiu del anàlisis dels píxels. Font: [14]

L'objectiu és esbrinar com de fosc és el píxel actual en comparació amb els píxels que l'envolten directament. Aleshores es dibuixa una fletxa que mostri en quina direcció la imatge s'enfosqueix.



Il·lustració 5.2.2.3.2 - Exemple d'un gradient mostrant el flux de la llum. Font: [14]

Si es repeteix aquest procés per a cada píxel de la imatge, es mostra la imatge amb cada píxel substituït per una fletxa. Aquestes fletxes s'anomenen gradients i mostren el flux de la llum a la foscor per tota la imatge.



Il·lustració 5.2.2.3.3 - Resultat de substituir els píxels per gradients. Font: [14]

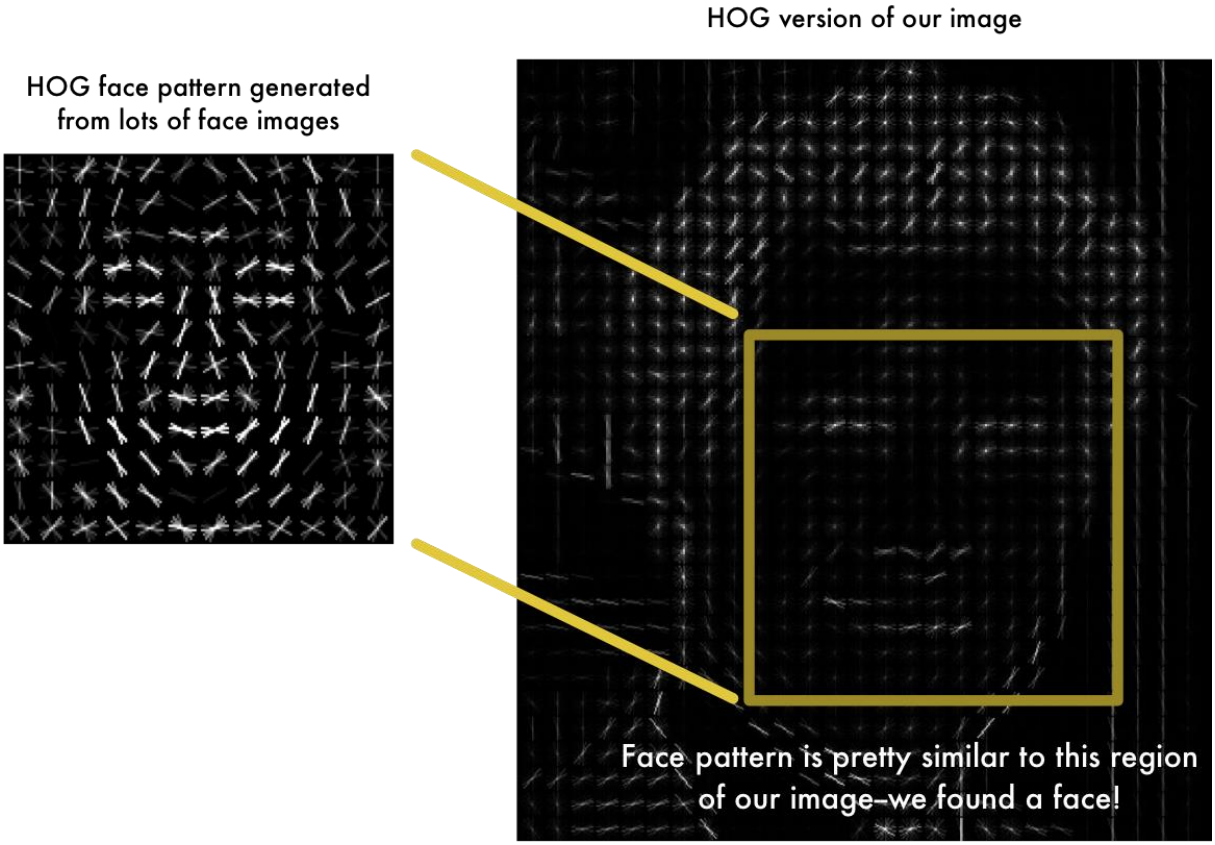
Si s'analitzen els píxels directament, les imatges molt fosques i les imatges realment clares de la mateixa persona tenen valors de píxels totalment diferents. Però només és te en compte la direcció en què canvia la lluminositat, tant les imatges molt fosques com les molt brillants acabaran amb la mateixa representació exacta.

Però desar el degradat per a cada píxel dona massa detalls, que realment no són necessaris. És millor que només es pogués veure el flux bàsic de llum/foscor a un nivell més simple per poder veure el patró bàsic de la imatge.

Per fer-ho, dividirem la imatge en petits quadrats de 16 x 16 píxels cadascun. A cada quadrat, comptarem quants gradients apunten en cada direcció principal (quants apunten cap amunt, cap

amunt a la dreta, cap a la dreta, etc.). A continuació, substituïm aquest quadrat de la imatge per les direccions de les fletxes que eren més fortes.

El resultat final és la conversió de la imatge original en una representació molt senzilla que capta l'estructura bàsica d'un rostre, aleshores per trobar cares és analitzar la imatge per a trobar un patró que s'assembli a un rostre.



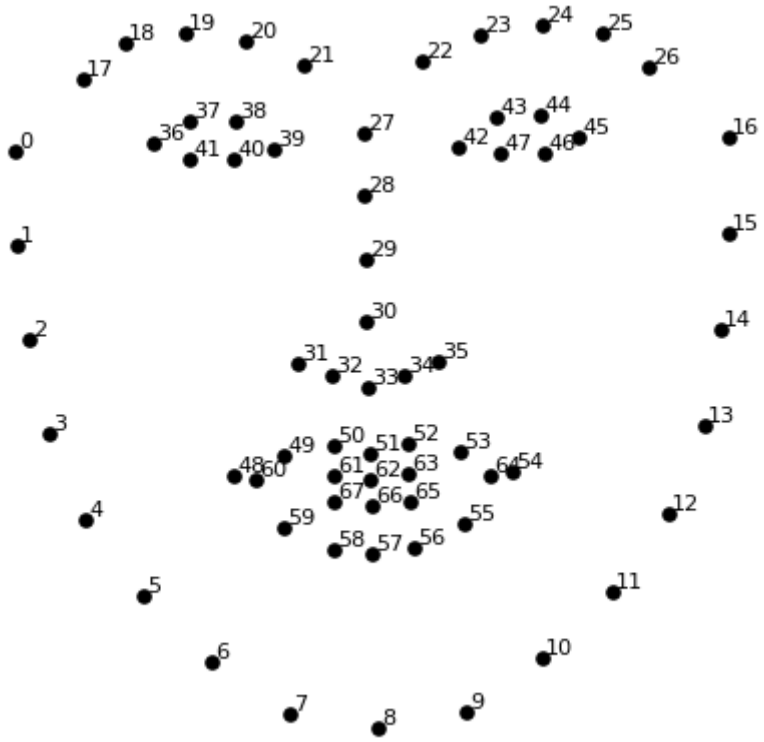
Il·lustració 5.2.2.3.4 - Representació gràfica de la comparació entre els dos patrons HOG generats (histogram of oriented gradients) Font: [14]

Pas 2 : La posició i projecció de la cara en la imatge

Per fer-ho, utilitzarem un algorisme anomenat estimació de punt de referència de cara. Hi ha moltes maneres de fer, però s'utilitza el proposat el 2014 per Vahid Kazemi i Josephine Sullivan.

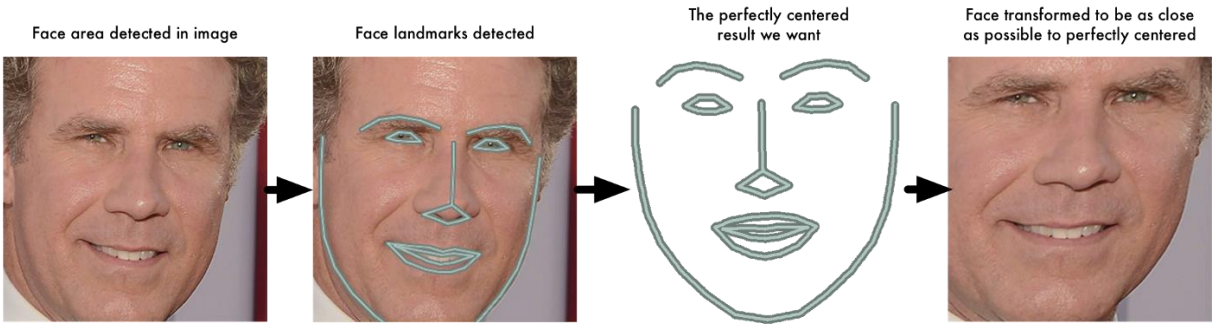
La idea bàsica és que es troben 68 punts específics (anomenats punts de referència) que existeixen a cada cara: la part superior de la barbeta, la vora exterior de cada ull, la vora interna

de cada cella, etc. Després s'entrena un algorisme de *machine learning* per poder trobar aquests 68 punts específics en qualsevol cara:



Il·lustració 5.2.2.3.5 - Representació gràfica dels 68 punts específics en cada cara. Font: [14]

Ara simplement es gira, s'escala i talla la imatge perquè els ulls i la boca estiguin centrats el millor possible. Només utilitzarem transformacions bàsiques d'imatge com la rotació i l'escala que conserven línies paral·leles (anomenades transformacions afins):



Il·lustració 5.2.2.3.6 - Procés de transformació d'imatge per centrar la cara. Font: [14]

Ara no importa com es giri la cara, podem centrar els ulls i la boca estan aproximadament a la mateixa posició a la imatge. Això farà que el següent pas sigui molt més precís.

Pas 3 : Codificació de la cara

El pas més important és aquest, els investigadors han descobert que l'enfocament més precís és calcular les mesures de la cara. El *deep learning* dur a terme un treball més exhaustiu que els humans en determinar quines parts de la cara són més importants a mesurar.

La solució és entrenar a una xarxa neuronal per a que generi 128 mesures per a cada rostre.

El procés funciona mirant tres imatges de cares a la vegada. L'algoritme analitza les mesures que està generant per cada una d'aquestes tres imatges i s'ajusta lleugerament la xarxa neuronal.

Aquest algoritme no cal entrenar cada cop que es troba amb una nova cara d'una persona, i hagi de fer totes les passes anteriors, sinó que ja esta preentrenada i en qüestió de segons ho processa.

Aleshores per cada imatge que tracta aquest algorisme genera 128 mesures.

Pas 4 : Trobar el nom de la persona reconeguda

Aquest darrer pas és el més senzill, consisteix en trobar la persona en la base de dades de persones conegudes que tinguin les mesures més properes a la imatge que s'analitza.

5.2.2.4 Implementació

El codi implementat es basa en el *workshop* de Murtaza's Workshop - Robotics and AI [15] , aquest es adaptat i més extens que l'original.

Es mostrarà les funcions més claus del codi:

Primer de tot, dins el codi de reconeixement facial, la funció findEncodings te com a objectiu en el primer bucle guardar en la variable **images** els arxius que siguin una fotografia.

El bucle final codifica i guarda en una variable, **encodeList,** totes les imatges codificades que prèviament han sigut passades a blanc i negre, necessari a la seva codificació tal i com s'ha explicat.

```
def findEncodings(images):
    myList = os.listdir(path)
    print(myList)
    for cl in myList:
        curImg=cv2.imread(f'{path}/{cl}')
        images.append(curImg)
        classNames.append(os.path.splitext(cl)[0])

    encodeList=[]
    for img in images:
        img = cv2.cvtColor(img,cv2.COLOR_BGR2RGB)
        encode = face_recognition.face_encodings(img)[0]
        encodeList.append(encode)
    print('Encoding complete')
    return encodeList
```

*Il·lustració 5.2.2.4.1 - Codi de la funció **findEncodings** pel sistema de reconeixement facial. Font: Elaboració pròpia*

Seguidament es detalla, també part del codi i part neuràlgic del codi de reconeixement facial, la funció **recognizedFaceFromFrame**. Aquesta funció rep com a paràmetres la llista d'imatges codificada **encodeListKnow**, comentada anteriorment i la imatge a processar **img**.

El primer pas de la funció és convertir la imatge que se li ha passat en blanc i negre. Converteix el format de la imatge captura del vídeo a una mida més petits per a un processament de reconeixement facial més ràpid i com a passos previs abans de començar el bucle localitza i codifica la cara o cares que trobi en la imatge.

Dins del bucle per a cada cara codificada es compara amb la nova i es troba si hi ha alguna codificació amb molt poc marge de diferència i es determina que és la mateixa persona.

```
def recognizeFaceFromFrame(img , encodeListKnown):

    uknownHuman=False

    img = cv2.cvtColor(imgS, cv2.COLOR_BGR2RGB)

    # Resize frame of video to 1/4 size for faster face recognition processing
    imgS = cv2.resize(img, (0,0), None, 0.25, 0.25)
    facesCurFrame = face_recognition.face_locations(imgS)
    encodesCurFrame = face_recognition.face_encodings(imgS,facesCurFrame)

    for encodeFace, faceLoc in zip(encodesCurFrame,facesCurFrame):
        matches = face_recognition.compare_faces(encodeListKnown, encodeFace)
        faceDis = face_recognition.face_distance(encodeListKnown,encodeFace)

        # # If a match was found in encodeListKnown, just use the first one.
        # if True in matches:
        #     first_match_index = matches.index(True)
        #     name = known_face_names[first_match_index]

        # Or instead, use the known face with the smallest distance to the new face
        print(faceDis)
        matchIndex = np.argmin(faceDis)
        if matches[matchIndex]:
            humanRecognizedName=classNames[matchIndex].upper()
            print("HUMAN RECOGNITED: " + humanRecognizedName)

        else:
            print("HUMAN NOT RECOGNITED")
            uknownHuman=True

```

*Il·lustració 5.2.2.4.2 - Codi de la funció **recognizeFaceFromFace** pel sistema de reconeixement facial. Font: Elaboració pròpia*

Per últim a destacar la funció **takeOnePhotoOfNewHuman** que forma part dins del mòdul encarregat de conèixer persones i registrar-les.

Bàsicament, si la foto presa per conèixer aquest nou humà ja existeix, cridant la funció prèviament explicada, no el registrarà de nou però si ho farà si cap codificació coincideix amb la actual.

```
def takeOnePhotoOfNewHuman(encodeListKnown, ):
    cap = cv2.VideoCapture(0)
    success, img = cap.read()
    if(face_recognizer.recognizeFaceFromFrame(img, encodeListKnown)):
        cv2.imwrite(os.path.join('/home/pi/adeept_darkpaw/server/recognizedHumans'
                                 , newHumanName +'.jpg'), img)
    else:
        print ("Human already known."+face_recognizer.getRecognizedHumanName())
    cap.release()

```

*Il·lustració 5.2.2.4.3 - Codi de la funció takeOnePhotoOfNewHuman de la classe **meet_new_human**. Font: Elaboració pròpia*

5.2.2.5 Resultats

La primera fase del desenvolupament va ser executar el codi proposat, de l'exemple del Murtaza's Workshop - Robotics and AI, en l'entorn Pycharm, prèviament descarregat en la Raspberry Pi i les llibreries instal·lades exclusivament per l'espai de treball del IDE Pycharm.

Els resultats són els esperats, la imatge que es recull a través de la càmera webcam es compara amb les imatges dins del directori amb la seva identificació com a nom de la imatge. Al trobar una cara en una de les imatges del vídeo en viu, la localitza i es tradueix en una codificació que més endavant compara amb el resta d'imatges ja codificades.

La segona fase és la prova fora de l'entorn de Pycharm, prèviament ja instal·lades les llibreries necessàries sense problema. A l'executar via terminal el fitxer Python el seu comportament és exactament l'esperat igual que la primera fase.

La tercera fase del desenvolupament del sistema, és l'optimització i la separació de les funcionalitats per a ser usades segons les necessitats.

Per assolir una millora d'escalabilitat s'ha dissenyat una estructura dividida en tres mòduls, el sistema de reconeixement i identificació facial, responsable de retornar un booleà si reconeix la cara que se li ha passat per imatge y retornar el seu nom. S'entén com a mòduls una forma de divisió de codi i unificada en un controlador.

Com a següent mòdul és el sistema de desar una nova cara d'una persona. D'una fotografia presa de la cara d'una nova persona guardar-la dins el directori de persones conegudes, prèviament haurem comprovat amb el mòdul de reconeixement facial que aquesta persona no estés ja guardada.

I com a mòdul final i punt d'unió entre el resta, el controlador de la càmera, responsable de gestionar totes funcionalitats futures relacionades amb aquest hardware com podria ser el reconeixement d'objectes.

5.2.2.6 Problemes

En l'entorn de treball que s'usa no hi ha cap problema en la instal·lació de la llibreria que s'esmenta i de les seves dependències com *ldib* tampoc.

Els resultats del codi són acceptables però no del tot esperats. Aquest algoritme de reconeixement facial no és precís totalment, ja que al detectar una cara d'una mateixa persona pot donar diferents valors de codificació segons l'angle en que es troba la cara.

Quan es capturada una cara gairabé de perfil on es poden identificar els dos ulls, la boca, nas i el contorn de la cara, codifica aquesta amb uns resultats totalment diferents si es capturés la cara totalment de front.

Per tant el millor mètode d'identificació és capturar la cara completament de front.

Cal comentar també que no és òptim la càrrega d'imatges de cares (**findEncodings**) ja enregistrades, ja que el temps empleat per fer-ho es molt costos i atura el programa.

5.2.3 Sistema de reconeixement de veu

Un sistema de veu a text (STT) és, com el seu nom indica, una manera de transformar les paraules parlades mitjançant el so en fitxers textuals que es poden utilitzar posteriorment per a qualsevol propòsit.

La tecnologia de reconeixement de veu és molt útil. Es pot utilitzar per a moltes aplicacions com l'automatització de la transcripció, escriure llibres/textos només amb la pròpia veu, permetre anàlisis complicades de la informació mitjançant els fitxers de text generats i moltes altres coses.

Avui dia hi han moltes eines i llibreries de veu a text de codi obert.

5.2.3.1 Software proposat.

Per tant, amb aquesta tecnologia és assequible l'objectiu de interpretar i executar les ordres donades per veu.

La primera proposta es basa en un exemple, de l'autor Satyajit Pattnaik[16] ,simulant una Alexa senzilla que serveix com a base d'implementació de la primera fase.

Destaquem la llibreria usada per la transcripció de veu a text "SpeechRecognition" que suporta diferents motors i APIs, offline i online:

- CMU Sphinx (funciona sense connexió)
- Reconeixement de veu de Google
- API de Google Cloud Speech
- Wit.ai
- Reconeixement de veu de Microsoft Bing
- API de Houndify
- Speech to Text d'IBM
- Snowboy Hotword Detection (funció sense connexió)

En la fase de proves de l'exemple esmentat anteriorment aquestes no són exitoses. Més endavant es descriu en detall perquè aquesta llibreria s'ha descartat.

Com a solució es proposa la nova llibreria **VOSK[17]**, un dels sistemes de reconeixement de veu de codi obert més nous, ja que el seu desenvolupament acaba de començar el 2020.

A diferència d'altres sistemes d'aquesta llista, Vosk està bastant a punt per utilitzar-se després de la instal·lació, ja que admet 10 idiomes (anglès, alemany, francès, turc...) amb models portàtils de 50 MB ja disponibles per als usuaris (hi ha altres models més grans fins a 1,4 GB si ho necessiteu).

També funciona en dispositius Raspberry Pi, iOS i Android, i proporciona una API de transmissió que us permet connectar-vos-hi per fer les vostres tasques de reconeixement de veu en línia. Vosk té enllaços per a Java, Python, JavaScript, C# i NodeJS.

Les seves avantatges a remarcar són:

- Admet més de vint idiomes i dialectes: anglès, anglès indi, alemany, francès, espanyol, portuguès, xinès, rus, turc, vietnamita, italià, holandès, català, àrab, grec, farsi, filipí, ucraïnès, kazakh, suec, japonès, Esperanto, hindi, txec..
- Funciona fora de línia, fins i tot en dispositius lleugers: **Raspberry Pi**, Android, iOS
- S'instal·la amb: **pip3 install vosk**
- Els models portàtils per idioma només tenen 50 Mb cadascun, però hi ha models de servidor molt més grans disponibles.
- Proporciona API de transmissió per a la millor experiència d'usuari (a diferència dels populars paquets Python de reconeixement de veu)
- També hi ha enllaços per a diferents llenguatges de programació: java/csharp/javascript, etc.
- Permet una reconfiguració ràpida del vocabulari per obtenir la millor precisió.
- Admet identificació dels parlants a més del simple reconeixement de veu.

5.2.3.2 Requeriments

Requeriments de VOSK:

- Linux on x86_64
- **Raspbian on Raspberry Pi**
- Linux on arm64
- OSX (only x86, not M1)
- Windows
- Python version: 3.5-3.9
- pip version: 20.3 o més nou.

Requisits de la llibreria SpeechRecognition :

- Python 2.6, 2.7 o 3.3+ (obligatori)
- PyAudio 0.2.11+ (només obligatori si necessiteu utilitzar l'entrada de micròfon, micròfon)
- PocketSphinx (només és necessari si necessiteu utilitzar el reconeixedor Sphinx, reconocer_instance.recognize_sphinx)
- Biblioteca de client de l'API de Google per a Python (només obligatori si necessiteu utilitzar l'API de Google Cloud Speech)
- Codificador FLAC (només necessari si el sistema no està basat en x86 Windows/Linux/OS X)

5.2.3.3 Funcionament intern de la llibreria

El software de reconeixement de veu funciona desglossant l'àudio d'un enregistrament de parla en sons individuals, analitzant cada so, utilitzant algorismes per trobar la paraula més probable que s'ajusti a aquest idioma i transcrivint aquests sons a text.

El reconeixement de veu utilitza processament del llenguatge natural (NLP) i xarxes neuronals basades en *deep learning*. "La PNL és una manera perquè els ordinadors analitzin, entenguin i derivin el significat del llenguatge humà d'una manera intel·ligent i útil", segons el bloc *Algorithma*.[23]

Això vol dir que el programari divideix el discurs en bits que pot interpretar, el converteix en un format digital i analitza els continguts.

A partir d'aquí, el software pren determinacions basades en la programació i els patrons de parla, fent hipòtesis sobre el que realment està dient l'usuari. Després de determinar el que probablement diuen els usuaris, el programari transcriu la conversa a text.

5.2.3.4 Implementació

Com que la implementació completa de la primera proposta no ha sigut possible, es detalla a continuació el codi exemple que usarem per a implementar en el projecte.

```
from vosk import Model, KaldiRecognizer
import pyaudio

model = Model(r'/home/pi/vosk-test/en') #read model
recognizer = KaldiRecognizer(model, 16000)

#Recognize from the microfone

cap = pyaudio.PyAudio()
stream = cap.open(format=pyaudio.paInt16, channels=1,
                  rate=16000, input=True, frames_per_buffer=8192)
stream.start_stream()

while True:
    data = stream.read(4096)
    #if len(data) == 0:
    #    break
    if recognizer.AcceptWaveform(data):
        print(recognizer.Result())
```

Il·lustració 5.2.3.4.1 - Codi exemple d'implementació de la llibreria VOSK. Font: Elaboració pròpia

Com a inici s'importen les llibreries VOSK i PyAudio, de VOSK s'utilitza els components Model i KaldiRecognizer.

Carreguem el model prèviament descarregat amb el mètode Model on hi passarem la seva ruta de directori on es guardada i tot seguit inicialitzem el reconeixedor (**recognizer**) amb el model carregat.

Es declara la variable **cap** on hi instanciem els sistema d'àudio PyAudio i seguit la variable **stream** obrim el flux d'áudio d'entrada al dispositiu desitjat, aquest cas el micròfon de la webcam amb els paràmetres desitjats. Com a pas final s'obre aquest canal creat per escoltar.

En un bucle infinit es comença a capturar les dades de so en la variable **data** i aquestes traduïdes a text amb la funció AcceptWaveform del component KaldiRecognizer.

El text a retornar després de cada transcripció te aquest format de sortida:

{

"text" : "hello world"

}

5.2.3.5 Resultats

Un cop finalitzada la instal·lació de la llibreria VOSK i PyAudio , programem un primer test basat en el codi anterior per a comprovar la seva funcionalitat i adquirir els coneixements pel seu ús.

Abans d'executar les proves, es requeria tenir un dels models portàtils per idioma que en la pàgina oficial són disponibles per a descarregar. Sense un model d'idioma la llibreria VOSK no és capaç de poder dur a terme la funcionalitat més bàsica, la transcripció.

Un cop descarregat el model d'idioma anglesa, entrem en fase de proves. És escollit l'idioma anglès en el reconeixement de veu per evitar problemes en quant a estabilitat del model ja que és una eina molt nova, d'aquesta manera també és més accessible a un major public i és internacional.

Al tenir les comandes per veu en anglès també fa més comprensible per a les persones que s'interessen en continuar un desenvolupament a partir d'aquest codi.

La fase de proves que és realitza es exitosa, però trobem algunes anomalies o problemes menors a l'hora de recollir la veu. La seva transcripció no era la esperada, al cap de varis intents

s'aconseguia transcriure la paraula desitjada, la causa més probable és la pronunciació ja que era angles americà i no britànic.

Per confirmar que el problema era derivat en anglès es realitzen proves amb altres models d'anglès i català.

5.2.3.6 Problemes

Al moment de desenvolupar el codi amb aquesta tecnologia, trobem que el Sistema Operatiu del fabricant no té controlador d'àudio.

Com a primera proposta provem d'instal·lar els paquets **alsa-utils**, **mpg321** i **cojo** per a la configuració d'àudio tal com aquesta guia ens indica[18].

A continuació carreguem el driver especificat: **sudo modprobe snd_bcm2835** i especifiquem la sortida del audio tal com ens indica: **sudo amixer cset numid=3 1.**

Finalitzat la guia d'instal·lació el resultat no es el esperat, l'àudio de sortida no reprodueix cap mena de so al fer les proves.

La sortida de so no és especificat correctament, era necessari especificar que la sortida de so era analògica amb aquesta comanda: **sudo amixer cset numid=1 1**

L'anterior comanda indicava que era per sortida HDMI.

Ja que aquest procés de configuració del controlador d'àudio no és persistent s'opta per automatitzar les comandes esmentades anteriorment amb un script:

#!/bin/sh

sudo modprobe snd_bcm2835

sudo amixer cset numid=1 1

Un cop comprovat que funciona correctament les comandes, es crea un script per a que quan arrenqui el sistema operatiu configuri automàticament el controlador de so i especifica el tipus de sortida d'àudio.

En la fase de proves de l'exemple esmentat anteriorment no són exitoses. La captura de veu no es pot realitzar per falta del paquet FLAC.

L'instal·lació d'aquest paquet tampoc ha estat possible ja que el programa de gestió de paquets és desactualitzat i la ruta de descarrega ja no existeix.

Per evitar desestabilitzar el sistema operatiu i perjudicar altres paquets usats pel funcionament del robot s'opta per a buscar un altre alternativa de llibreria a usar, com ja s'ha explicat VOSK.

5.2.4 Sistema de text a veu

Text to Speech (TTS)[19] és un procés de modelització de llenguatge natural que requereix canviar unitats de text en unitats de veu per a la presentació d'àudio. Això és el contrari de la parla al text, on una tecnologia pren les paraules parlades i intenta registrar-les amb precisió com a text.

El text a veu és ara comú a les tecnologies que busquen renderitzar la sortida d'àudio del text digital per ajudar aquells que no poden llegir o per a altres tipus d'usos.

5.2.4.1 Software proposat

Una de les primeres propostes és, ja usada en l'anterior codi de reconeixement de veu, la biblioteca pyttsx3 per a Python.

Aquesta llibreria causa un error de compilació per falta d'un fitxer en el paquet instal·lat que més endavant es comenta i és per això que es proposa com a solució l'API Text to Speech de Google coneguda com a API gTTS. gTTS és una eina molt fàcil d'utilitzar que converteix el text introduït en àudio que es pot desar com a fitxer mp3.

L'API gTTS admet diversos idiomes, com ara anglès, hindi, tàmil, francès, alemany i molts més. El discurs es pot oferir a qualsevol de les dues velocitats d'àudio disponibles, ràpida o lenta. Tanmateix, a partir de l'última actualització, no és possible canviar la veu de l'àudio generat.

5.2.4.2 Requeriments

pyttsx3 A diferència de les biblioteques alternatives, funciona fora de línia i és compatible amb Python 2 i 3.

gTTS Requereix de connexió a internet per a funcionar.

5.2.4.3 Funcionament intern de la llibreria

pyttsx3[20] és una biblioteca de conversió de text a veu en Python. A diferència de les biblioteques alternatives, funciona sense connexió a internet.

És una eina molt fàcil d'utilitzar que converteix el text introduït en veu. La llibreria **pyttsx3** admet dues veus, la primera és femenina i la segona és masculina, proporcionada per "sapi5" per a Windows.

gTTS[21] una biblioteca de Python i una eina CLI per interactuar amb l'API de text a veu de Google Translate.

5.2.4.4 Implementació

En l'apartat d'implementació es descriu la llibreria gTTS ja que ha donat els resultats esperats.

Per a implementar la llibreria gTTS i usar-la analitzarem el següent codi, el qual conté les funcions claus pel seu funcionament:

```
# Import the required module for text
# to speech conversion
from gtts import gTTS
# This module is imported so that we can
# play the converted audio
import os

# The text that you want to convert to audio
mytext = 'Welcome to geeksforgeeks!'

# Language in which you want to convert
language = 'en'

# Passing the text and language to the engine,
# here we have marked slow=False. Which tells
# the module that the converted audio should
# have a high speed
myobj = gTTS(text=mytext, lang=language, slow=False)

# Saving the converted audio in a mp3 file named
# welcome
myobj.save("welcome.mp3")

# Playing the converted file
os.system("mpg321 welcome.mp3")
```

Il·lustració 5.2.4.4.1 - Codi exemple d'implementació de la llibreria gTTS. Font: Elaboració pròpia

Primer de tot, com tota llibreria s'ha d'importar i en aquest cas es requereix la llibreria **os** per a poder reproduir el audio resultant del text.

A destacar la funció gTTS, en la línea 18 de la representació gràfica anterior, on especifiquem quin text volem traduir a veu, el llenguatge en el qual volem que aquest text sigui parlat i com a últim atribut un booleà que indica a quina velocitat es vol que sigui llegida.

El resultat de la funció gTTS es recull en un objecte, que aquest serà guardat en format mp3.

Com a pas final es reprodueix aquest àudio creat resultant la traducció de text a veu.

5.2.4.5 Resultats

Els resultats de la llibreria pyttsx3 no són els esperats, al ser executat el test retorna un error relacionada amb el sistema operatiu: **OSError: libespeak.so.1: cannot open shared object file: No such file or directory** es comenta amb més detall el següent apartat.

En canvi els resultats amb la llibreria gTTS són exitosos, s'aconsegueix traduir a veu els textos desitjats sense cap error.

5.2.4.6 Problemes

Al compilar la llibreria **pyttsx3** donava un error al compilar, per falta d'un fitxer que proporciona una llibreria que com ha passat anteriorment el gestor de paquets del sistema operatiu no es actualitzat.

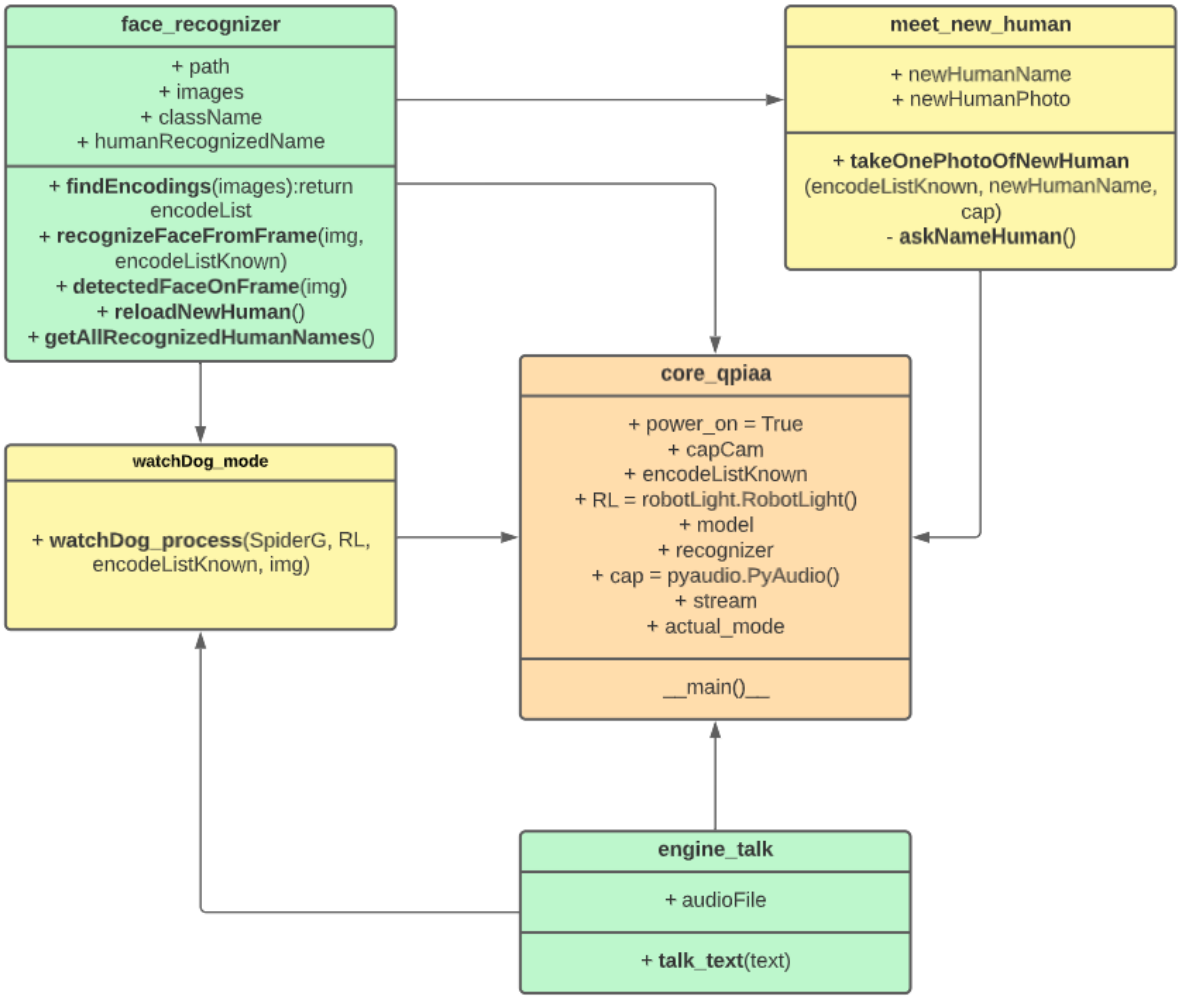
Al provar varies solucions, s'ha decidit usar una altre llibreria tot hi necessitar de connexió a internet.

Aleshores com a solució final i més ràpida a aplicar s'implementa la llibreria **gTTS** tot hi dependre de connexió a internet.

5.2.5 Integració

Un cop s'ha instal·lat i provat individualment cada mòdul, el pas final es la integració de cada un i el desenvolupament final del codi que permet assolir els objectius del projecte.

Per a facilitar el desenvolupament es decideix separar les funcionalitats per llibreries, que són anomenats "mòduls", i aquests seran gestionats pel core que serà el punt d'unió de tots els mòduls que s'executarà amb tots els privilegis.



Il·lustració 5.2.5.1 - Diagrama dels mòduls i el core desenvolupats. Font: Elaboració pròpia

A continuació es defineix amb més detall cada mòdul i el core.

Core

El nucli del programa, o core, es el responsable de gestionar, mantenir les dades i executar les funcionalitats dels inputs que rep de l'exterior.

La lògica que s'hi implementa en el core es basat amb el programa **webServer**, software proporcionat pel fabricant del kit robòtic. Els mòduls necessaris del fabricant que s'importen són:

- **SpiderG**, responsable del control dels servomotors. Permet executar les comandes de moviment del quadrúpede i altres funcionalitats relacionades amb el giroscopi.
- **RobotLight**, proporciona funcions per a poder controlar els led RGB.
- **Switch**, mòdul més simple que la resta, que permet el control dels ports del HAT i que en aquest cas és útil per activar o desactivar el led blau llanterna.

A més, els mòduls desenvolupats prèviament per usar les funcionalitats de:

- reconeixement facial (**face_recognizer**)
- conèixer un nou humà (**meet_new_human**)
- capturar la veu(**vosk**)
- de text a veu (**engine_talk**)

La resta de mòduls desenvolupats (**watchDog_mode, face_recognizer, meet_new_human, engine_talk**) es defineixen més endavant en aquesta secció.

Els primers passos del core son inicialitzar els dotze servomotors del quadrúpede en la posició inicial i comprovar si es connectat el giroscopi MPU-6050.

Es declara el dispositiu webcam que usarem dins la variable **capCam** i en la variable **encodeListKnown** carregarem la llista de totes les cares codificades.

S'importa la llibreria Vosk en el core, ja que el flux del core será determinat per les comandes de veu que rep del usuari i per tant no hi ha cap mòdul de reconeixement de veu.

S'iniclializa el **model** de l'idioma que en aquest cas és el model anglès, el **KaldiRecognizer** inicialitzat en la variable **recognizer** que com a paràmetres rep el model i la freqüència d'àudio

processada. Es pren la decisió d'usar el model anglès per a que sigui entenedor a un major públic.

La variable **cap** que captura l'àudio pel micròfon de la webcam, es defineix com es la transmissió d'àudio en la variable **stream** i es iniciada la captura de la transmissió.

Les següents comandes tenen com a objectiu comprovar el moviment dels servomotors, puja i baixa el cos del quadrúpede i torna a l'estat inicial. D'aquesta manera es possible detectar si un servomotor no executa el moviment esperat.

Un cop finalitza la comprovació dels servos, s'executa tota la primera fase del procés de reconeixement facial al usuari.

Dins d'un bucle analitza cada frame de la transmissió vídeo, que rep de la càmera, accessible desde la variable **capCam**.

Al detectar una cara en un dels frames el bucle s'atura i es recupera el nom de la cara si es registrada amb la funció **detectFaceOnFrame** del mòdul **face_recognizer** , del contrari es retorna el nom per defecte **"Uknown human".**

Si el nom guardat dins la variable **humanName** és 'Uknown human', es tracta d'un usuari desconegut o la captura de la cara ha generat una codificació diferent.

En aquest cas es pregunta a l'usuari el seu nom i s'executa al procediment de conèixer a un nou humà amb la funció **takeOnePhotoOfNewHuman** del modul **meet_new_human**.

A continuació s'actualitza la variable que contenia la llista de cares codificades afegint la del nou usuari i es guarda el nom del nou usuari amb qui tractarem.

D'altre manera si aquest usuari ja es conegut, es retorna el nom d'aquest i no cal registrar-lo de nou.

Abans de començar el *loop* principal, es dona la benvinguda via veu a l'usuari, informació del mode actual, per defecte sempre hi serà en remot control, es a dir el mode vigilància no es activat i les comandes de veu son executables.

Dins del loop principal es comprova quin mode es seleccionat, si és en mode remot comença a transcriure el que rep per la transmissió d'audio. La transcripció es guarda dins la variable

command_input i en una bateria de condicionals s'executa la comanda o s'ignora si no coincideix cap.

Si és en mode vigilància, s'executa el codi de la funció **WatchDog_process** del mòdul WatchDog_mode en bucle, alhora es transcriu la entrada d'àudio i al rebre la comanda de canvi de mode es surt del bucle on s'executa el procés de vigilància retornant al mode remot control.

Engine talk

El mòdul **engine_talk** te com a funció única traduir aquell text que rep a veu. Son importades la llibreria **gTTS** esmentada anteriorment (google Text To Speech) i **os**, utilitzada per a reproduir els arxius mp3 generats a partir del text.

En aquest mòdul és definit una funció **talk_text** amb un paràmetre, **text**. Al executar aquesta funció la cadena de caràcters es guarda en la variable **text** i es passada com a paràmetre a la funció gTTS de la llibreria. També es especificat com a paràmetre el idioma i la velocitat en la que será parlada.

En aquest cas l'idioma anglès és el utilitzat i la velocitat de parla serà lenta. Un cop aquest àudio és generat del text, es guardat dins la variable **textToSay** i es guardat com a arxiu mp3.

Les decisions per a usar l'anglès són semblants al cas anterior del reconeixement de veu, per arribar a un públic més gran i internacional.

Finalment es reproduït amb la llibreria **so el** fitxer mp3 generat.

Face recognizer and identification

Com s'ha vist anteriorment, la seva funció és identificar i reconèixer cares humanes. S'importen les llibreries **cv2, numpy,** i **face_recognition**.

I conte les següents funcions:

- **findEncodings.** Per paràmetre rep el directori on es conte totes les imatges de les cares conegudes.

  És format per dos loops, el primer itera sobre els noms de les imatges, per a guardar els noms que pertany a cada cara.

El segon loop itera sobre les imatges i per a cada una d'elles , amb la llibreria **cv2,** es transformen a blanc i negre, i es codifiquen. Cada codificació de les cares es guarda dins d'una *array* i aquesta es retorna.

- **recognizeFaceFromFrame.** Per paràmetre rep la imatge que es vol analitzar i la llista de cares codificades. La imatge es canvia a blanc i negre, i es redimensiona per una major rapidesa en el procés de reconeixement.

  Es localitzen les cares identificades i es codifiquen. Dins una iteració, per cada codificació de les cares que s'han trobat es comprova si coincideix amb la llista de les cares ja conegudes.

  Si la cara coincideix amb la codificació d'una ja coneguda es retorna el nom que pertany aquella cara que es coneguda, d'altre manera es retorna 'Uknown Human'.

- **detectedFaceOnFrame.** Detecta si hi ha almenys una cara en la imatge que rep per paràmetre. Un procés molt semblant a l'anterior, es modifica la imatge a blanc i negre, i es redimensiona.

  Si al codificar la imatge retorna valors buits, no ha detectat cap aleshores es retorna un *False* i de manera contraria un *True*.

- **reloadNewHuman.** Crida la funció **findEncoding** i com a paràmetre rep la direcció de la carpeta on es guarden les imatges de les cares conegudes.

  Quan el procés de reconèixer a una persona es crida aquesta funció, per a que s'afegeixi a la llista de les cares conegudes.

Meet a new Human

L'objectiu principal d'aquest mòdul és el de guardar la cara d'una persona com a coneguda. S'importa el mòdul face_recognizer per a comprovar que la nova cara a guardar ja no sigui coneguda, si no és així guarda la cara d'aquesta nova persona i el seu nom.

Les següents funcions que s'implementen son:

- **takeOnePhotoOfNewHuman.** Rep per paràmetre **encodeListKnown** (llista de cares conegudes), el nom de la nova persona a conèixer i la retransmissió video d'on capturar la imatge.

Es comprova de la imatge capturada si ja es coneix la cara nova a guardar, si no es així guarda la imatge amb el nom de la persona nova. Del contrari dona una alerta de que ja la coneix.

- **askNameHuman.** Pregunta el nom de la nova persona a conèixer, si aquest no és ja enregistrat continua amb el procés i retorna el nom.

Watch dog mode

En aquest mòdul es preconfigura el procés que fa el quadrúpede quan s'activa el mode de vigilància.

Es comprova si es detecta una cara en els frames del vìdeo que va analitzant en cada volta. Si detecta una cara, la identificarà i si aquesta cara es reconeguda continuarà girant sobre si mateix.

Si aquesta cara no es pot identificar, dona una alarma per veu i activa els led RGB en mode policia.

Aquest procés es definit en una única funció, **watchDog_process**, que rep per paràmetre:

- **SpiderG.** La instancia del mòdul de moviment inicialitzat en el core.
- **RL.** Mòdul de control de leds inicialitzat. (**RobotLight**)
- **encodeListKnown.** La llista de les cares codificades que son conegudes.
- **img.** El frame capturat en el *loop* del core.

S'importen els mòduls **engine_talk** per a donar l'avís per veu en cas de no identificar la cara d'una persona, **face_recognizer** per reconèixer i identificar els rostres, i **cv2** per la captura dels frames vìdeo.

6. Conclusions

Com a primera conclusió destacable, es que el resultat d'aquest projecte com a primera idea ha estat molt ambiciós.

La primera proposta ha perdut forma durant el desenvolupament d'aquest projecte, les dificultats i problemes trobats durant el muntatge i integració del software extra, necessari per donar funcionalitats noves al quadrúpede, han suposat un gran hàndicap en quant al factor temps.

Durant la pandèmia, la crisis de semiconductors va afectar a nivell mundial i aconseguir una Raspberry Pi vas ser difícil, per a no perdre el temps es va continuar el muntatge del quadrúpede sense aquesta.

A finals de l'any passat es va poder aconseguir una Raspberry Pi compatible amb el kit i es va aconseguir fer les primeres proves software del fabricant.

Com s'ha comentat anteriorment, l'inconvenient relacionat amb la càmera, va suposar un consum de recursos molt gran per determinar finalment que la causa provenia del miniordinador.

Es va prendre la decisió de continuar el muntatge desenvolupant un programa senzill per a configurar els servomotors al angle d'inici sense el software del fabricant.

Paral·lelament es busca una solució al inconvenient de la càmera i es contacta amb l'equip de suport. Aquesta decisió de continuar buscant una solució a la càmera via serial no va ser acertada.

La decisió de analitzar el codi del fabricant i comprovar que es podia usar una cámara USB va ser una bona inversió de temps, d'aquesta manera solucionavem dos punts a nivell hardware, instalar un micròfon i el problema de rebre senyal video.

A mesura que es desenvolupa el software es prefereix no invertir en més material, i s'usarà altaveus inalambrics per a les proves finals i durant les proves via connexió jack. Es va prendre la decisió d’usar altaveus externs per a no invertir més recursos en fer funcionar hardware afegit al quadrúpede.

A nivell software hi van sorgir mes imprevistos, el script de instalación del software necessita llibreries desactualitzades o versions ja no existents. Com a alternativa vem usar la imatge que el fabricant ha desenvolupat i inclou totes les llibreries necessàries pel funcionament bàsic d'aquest.

La decisió d'usar la imatge proporcionada pel fabricant va ser la més adient per no perdre més el temps, en re-escriure tot el codi del fabricant i buscar les llibreries més adequades.

Com s'ha esmentat en els sistemes de reconeixement de veu i de text a veu, els gestor de paquets i llibreries eren desactualitzats en el sistema operatiu de la imatge, causant contratemps.

Tot i que a nivell tècnic ha estat un gran repte per el nivell d'imprevistos i tindre que solucionar constantment problemes, la gran majoria de objectius han estat assolits, menys donar avisos per a l'aplicació web i el sistema de cerca i detecció d'objectes.

Es pot afirmar que l'objectiu d'integrar nou software i afegir noves funcionalitats és possible, però es molt limitat a nivell de hardware ja que el reconeixement facial funciona a curta distància i l'autonomia de les bateries no és molt alta.

6.1. Posibles ampliacions

Una de les ampliacions, es afegir la funcionalitat de cercar i identificar objectes, que per falta de temps no s'ha pogut implementar. Aquesta funcionalitat seria activada amb una comanda de veu, i seria executada de la mateixa forma que el mode vigilància.

Altre ampliació i que tampoc s'ha pogut desenvolupar és l'aplicació web per a rebre notificacions d'alerta d'intrusos, visualitzar càmera i control remot a distancia. L'aplicació client-servidor desenvolupada pel fabricant pot ser modificada per a oferir aquestes funcionalitats, a mes a nivell usuari, es mes còmode l'ús del robot i ofereix més prestacions.

A nivell de programació, una ampliació podria ser l'optimització del codi gestionant i executant els mòduls en diferents processos.

Com s'ha comentat anteriorment el mòdul de moviment del quadrúpede, per exemple, s'executa en un procés a part fent possible executar una comanda de moviment, mentres es processa la veu que escolta.

Per a optimitzar el temps de càrrega en la codificació de cares i desocupar memòria física, una possible ampliació seria importar totes les cares codificades amb els seus noms en una base de dades en línia com ara Firebase, accedint de manera ràpida i sense codificar de nou totes les cares del directori.

Per acabar, com a ultima ampliació es podria afegir un sistema de mesura de distàncies mitjançant retransmissió vídeo[26]. Seria necessari fer un anàlisis previ si el software actualment gratuït permet identificar la distancia a objectes mentres és en moviment.

Aquest sistema permet enriquir els altres dos mòduls de vigilància i cerca d'objectes de manera que es pogués moure lliurement sense necessitat de mapejar una estància.

7. Bibliografia

[1] Raspberry Pi [en línia] [consulta: 1 de febrer de 2022]. Disponible a https://es.wikipedia.org/wiki/Raspberry_Pi

[1] Usos de la Raspberry Pi [en línia] [consulta: 1 de febrer de 2022]. Disponible a https://computerhoy.com/noticias/hardware/15-usos-raspberry-pi-que-no-sabias-que-podias-darle-74905

[3]Las mejores rivales y alternativas a la Raspberry Pi 4 [en línia] [consulta: 8 de febrer de 2022]. Disponible a https://www.adslzone.net/listas/gadgets/alternativas-raspberry-pi/

[4]Adeept RaspClaws [en línia] [consulta: 8 de febrer de 2022]. Disponible a https://www.amazon.es/Adeept-RaspClaws-Hexapod-Raspberry-transmisi%C3%B3n/dp/B085DF292N/

[5]OSOYOO Mega2560 [en línia] [consulta: 8 de febrer de 2022]. Disponible a https://www.amazon.es/OSOYOO-Mega2560-Controlled-Educational-Mechanical/dp/B082DBYH9G/

[6]Adeept-RaspTank [en línia] [consulta: 8 de febrer de 2022]. Disponible a https://www.amazon.es/Adeept-RaspTank-inteligente-inal%C3%A1mbrico-seguimiento/dp/B085C6Q48P/

[7] Projecte de Saral Tayal [en línia] [consulta: 8 de febrer de 2022]. Disponible a https://www.instructables.com/Object-Finding-Personal-Assistant-Robot-Ft-Raspber/

[8] Adeept DarkPaw recursos [en línia] [consulta: 5 de febrer de 2022]. Disponible a https://www.adeept.com/learn/detail-38.html

[9] Certificat LPI Linux [en línia] [consulta: 5 de febrer de 2022]. Disponible a https://www.udemy.com/course/certificacion-lpi-linux-essentials-temario-oficial-examen/

[10] SSTec Tutorials [en línia] [consulta: 5 de febrer de 2022]. Disponible a https://www.youtube.com/c/SSTecTutorials

[11] Heap Art Coding [en línia] [consulta: 5 de febrer de 2022]. Disponible a https://www.youtube.com/channel/UCqhzrDyBdR8iqKTnjbqqHCw

[12] Face Recognition: how it works and its safety [en línia] [consulta: 22 d'abril de 2022]. Disponible a https://www.electronicid.eu/en/blog/post/face-recognition/en

[13] Face Recognition PyPI.org [en línia] [consulta: 22 d'abril de 2022]. Disponible a https://pypi.org/project/face-recognition/

[14] Machine Learning is Fun! Part 4: Modern Face Recognition with Deep Learning [en línia] [consulta: 22 d'abril de 2022]. Disponible a https://medium.com/@ageitgey/machine-learning-is-fun-part-4-modern-face-recognition-with-deep-learning-c3cffc121d78

[15] FACE RECOGNITION + ATTENDANCE PROJECT | OpenCV Python | Computer Vision [en línia] [consulta: 22 d'abril de 2022]. Disponible a https://www.youtube.com/channel/UCYUjYU5FveRAscQ8V21w81A

[16] GitHub:pik1989 [en línia] [consulta: 22 d'abril de 2022]. Disponible a https://github.com/pik1989/Alexa/blob/main/Alexa%20App.py

[17]Top 10 Open Source Speech Recognition Systems [2022] [en línia] [consulta: 22 d'abril de 2022]. Disponible a https://fosspost.org/open-source-speech-recognition/

[18] Raspberry Pi – Setting Up Your Audio [en línia] [consulta: 22 d'abril de 2022]. Disponible a https://iwearshorts.com/blog/raspberry-pi-setting-up-your-audio/

[19] Text to Speech (TTS) [en línia] [consulta: 22 d'abril de 2022]. Disponible a https://www.techopedia.com/definition/23843/text-to-speech-tts

[20] Python | Text to Speech by using pyttsx3 [en línia] [consulta: 22 d'abril de 2022]. Disponible a https://www.geeksforgeeks.org/python-text-to-speech-by-using-pyttsx3/

[21] gTTS[en línia] [consulta: 22 d'abril de 2022]. Disponible a https://gtts.readthedocs.io/en/latest/

[22] Handling OSError exception in Python [en línia] [consulta: 22 d'abril de 2022]. Disponible a https://www.geeksforgeeks.org/handling-oserror-exception-in-python

[23] Introduction to Natural Language Processing [en línia] [consulta: 22 d'abril de 2022]. Disponible a https://www.datarobot.com/blog/what-is-natural-language-processing-introduction-to-nlp/

[24] ¿Por qué la Raspberry Pi es tan exitosa dentro del mercado? [en línia] [consulta: 22 d'abril de 2022]. Disponible a https://www.islabit.com/81569/por-que-la-raspberry-pi-es-exitosa.html

[25] Los siete mejores proyectos con una Raspberri Pi para construirte tu propia consola de juegos [en línia] [consulta: 22 d'abril de 2022]. Disponible a https://www.xataka.com/robotica-e-ia/los-siete-mejores-proyectos-con-una-raspberri-pi-para-construirte-tu-propia-consola-de-juegos

[26]Find distance from camera to object/marker using Python and OpenCV [en línia] [consulta: 14 junio de 2022]. Disponible a https://pyimagesearch.com/2015/01/19/find-distance-camera-objectmarker-using-python-opencv/