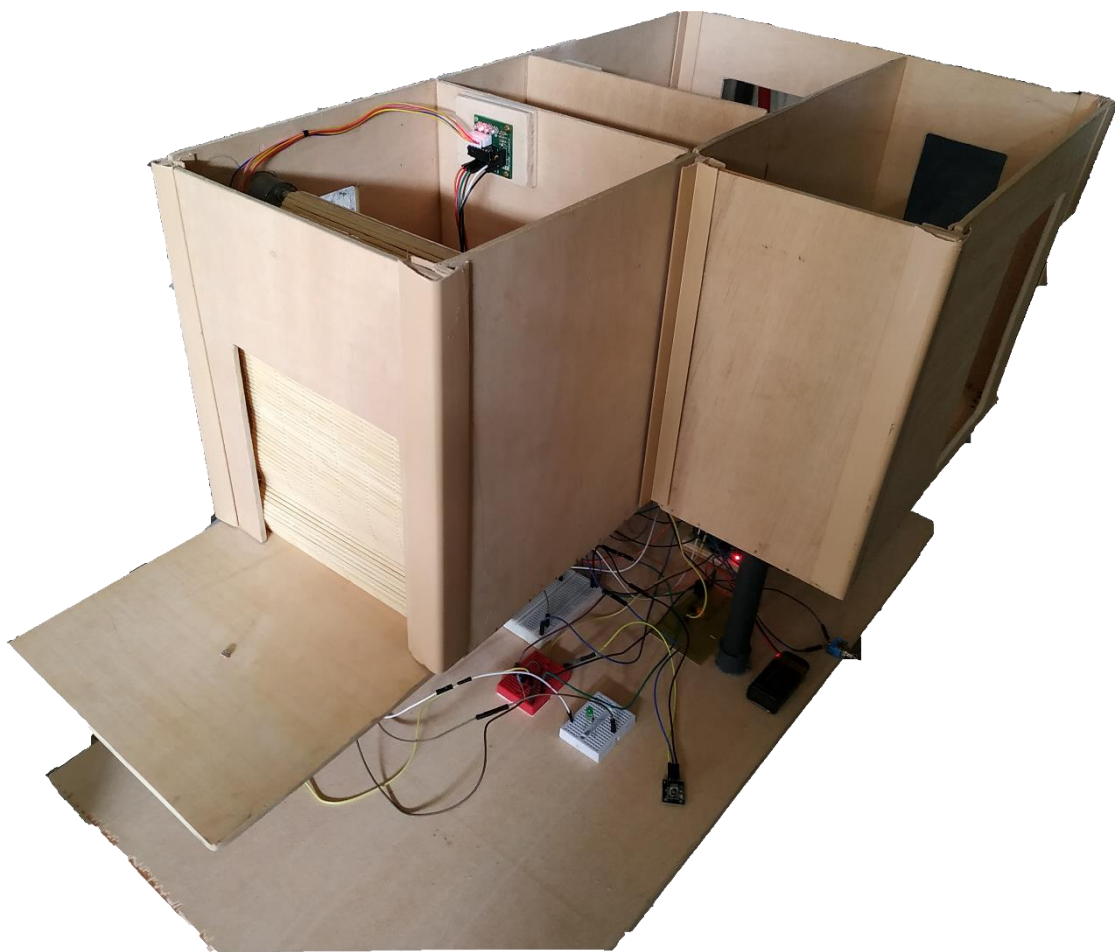


# La domòtica aplicada a un habitatge

## Sistemes domòtics basats en Raspberry Pi



*Arnau Llop Iglesias*

*Tutor: Ernest Messeguer Romero*

*Institut Antoni de Martí i Franquès*

*14 de gener de 2018*

## **Resum científic**

L'objectiu d'aquest Treball de Recerca és construir la rèplica d'una casa, domotitzar-la i estudiar-la.

Es parteix de la base que amb productes assequibles i accessibles es pot domotitzar un habitatge amb un pressupost relativament reduït.

Per tal d'aconseguir els objectius es dissenyen uns plànols de la casa que després es construeix com a maqueta. Seguidament s'ajunten tots els components electrònics i elèctrics i es programen. Finalment s'instal·len a la maqueta.

D'altra banda s'utilitza l'AutoCAD com a programa de disseny tridimensional per fer els plànols. Per la programació s'usa el llenguatge Python (de senzilla sintaxis i processament potent) i es concentra tot en una Raspberry Pi 3B que és un ordinador mono-placa.

Es conclou que el hardware, el sistema d'informatització i els algorismes desenvolupats no són prou eficients, tot i que funcionals cent per cent.

L'estudi realitzat pot ajudar a molts consumidors a adonar-se que informatitzar una casa pot arribar a significar estalvi energètic, comoditat i seguretat.

**Paraules clau:** Eficiència, llenguatge de programació, domòtica, economia domèstica, Raspberry Pi 3B

## **Abstract**

The objective of this Research Project is to build a house, domotize it and study it.

It is assumed that using not expensive and accessible products is possible to make an automated house.

To achieve that goal, a blueprint is drawn up and the prototype is made from that blueprint. First at all, some electronics are gathered and then programmed. Finally they are installed into the prototype.

Secondly, AutoCAD is used; Python as the main programming language; and as the processing core, a Raspberry Pi 3 B.

The investigation led to the conclusion that the hardware configuration is anything but efficient and so is the software, although, it is all fully functional.

The research can help consumers to domotize their own houses, meaning, savings, comfort and security.

**Key words:** Efficiency, programming language, home automation, domestic economy, Raspberry Pi 3 B

Índex .....	1
Introducció .....	2
Fonaments teòrics	
1. Domòtica .....	5
1.1. Definició .....	5
1.2. Antecedents històrics .....	6
1.3. Aplicació .....	7
1.4. Classificació .....	8
1.5. Elements .....	10
2. Raspberry .....	11
2.1. Definició .....	11
2.2. Antecedents històrics .....	12
2.3. Llenguatges de la Raspberry .....	12
3. Programació .....	13
3.1. Llenguatges de programació .....	13
3.2. Llenguatge utilitzat .....	14
3.3. Linux i terminal .....	15
Fonaments pràctics	
4. Descripció del projecte .....	17
5. Maqueta .....	19
5.1. Materials .....	19
5.2. Plànols .....	24
5.3. Construcció .....	25
6. Programació i electrònica .....	31
6.1. Posta a punt de la Raspberry .....	31
6.2. Programació .....	33
7. Funcionament dels sistemes .....	43
8. Conclusions .....	45
Bibliografia	
Annexos	

## INTRODUCCIÓ

Presento a continuació aquest treball de recerca de segon de batxillerat, en el qual he intentat, mitjançant la programació informàtica de circuits electrònics, implantar en un habitatge eines que la fan més segura i més còmoda.

Amb aquest projecte intentaré comprovar si seria possible domotitzar, en certs aspectes, una casa, sense haver de recórrer a les empreses especialitzades.

El motiu pel que he triat aquest projecte és el meu interès per la programació i l'electrònica. Però, més enllà d'això, del fet que em semblava molt interessant la domòtica i m'agradava molt aquest àmbit, també penso que en el món laboral hi ha una gran demanda d'enginyers i estic convençut que tots aquells a qui ens agrada aquest sector podrem viure d'això.

Si aconseguim fer la casa domòtica “definitiva” assumeixo que podrem reduir l'impacte atmosfèric i econòmic de la “sobre-electrificació” domèstica a la que estem sotmesos, cada dia més. Constantment surten nous aparells domèstics connectats a la internet, i això implica un increment en el consum d'electricitat.

Considero que caldria reduir l'impacte mediambiental de certs factors quotidians com ara els automòbils. Sembla ser que no interessa avançar amb els motors de cotxe, sembla que estiguem sotmesos a què perduri el mateix motor que portem fent servir des de fa dècades i que tant consumeix i contamina. Suposo que hi ha interessos econòmics prou importants que limiten la seva adequació a una energia sostenible, però, potser per això mateix, entenc que, en la mesura del possible hauríem d'avançar encara més en altres sectors de la nostra vida i optimitzar les vivendes en seria un.

Considero que un dels sectors en el que hem d'avançar és la domòtica. Un estudi domèstic que ens permet portar la eficiència al límit aconseguint reduir l'impacte mediambiental i augmentar la comoditat que obtenim a casa nostra.

M'he volgut basar en un ordinador anomenat Raspberry Pi, que em permet fer-ho tot des d'una sola placa. Les implementacions electròniques i informàtiques les posaré a prova sobre una maqueta de contraxapat de construcció pròpia.

Així doncs, aquest projecte tracta de la nostra hipotètica comoditat viable i energèticament sostenible: la domòtica; encara que en aquest cas no incidiré tant en la sostenibilitat com en la comoditat o la funcionalitat, perquè el temps per a fer aquest projecte no ha estat suficient en el meu cas.

Això, doncs en aquest treball construiré una maqueta per a instal·lar-hi sistemes que hauré automatitzat per tal de defensar la meva hipòtesi.

La fita que m'he proposat, serà aconseguir comprovar, si és possible o no construir una casa verda i còmode, és a dir, respondrem les següents preguntes:

- És viable una casa domòtica?
- Quan és poc i quan és prou?
- Què és el que he de saber per fer la meua pròpia casa domòtica?
- Quant val?
- Val la pena?

Ja veuré, d'aquí molts mesos, molts maldecaps i moltes proves, quina és la resposta a totes aquestes preguntes. Al final del treball les podré respondre.

Podria considerar d'afegir un objectiu més pel fet d'haver triat com a ordinador base la Raspberry Pi, perquè jo estic acostumat i familiaritzat a programar Arduino, i realitzar aquest

projecte amb una placa Arduino mega, o due, em seria molt més senzill i fàcil. Però d'aquesta manera, a part que la Raspberry Pi té una capacitat de càlcul sublim si la comparem amb l'Arduino, em puc introduir a altres mons, com el de Python, C simplificat, la mateixa Raspberry Pi... Sortir de la zona de confort per aprendre més i millor; aquest seria un altre objectiu.

La Raspberry Pi es pot programar còmodament amb dos llenguatges de programació, el C simplificat i el Python, jo faré servir el Python, tot i que per depèn quina feina, optaré pel C simplificat.

Vull acabar aquesta introducció amb el pensament de Bill Gates que diu:

1. "The first rule of any technology used in a business is that automation applied to an efficient operation will magnify the efficiency.
2. The second is that automation applied to an inefficient operation will magnify the inefficiency."

En aquesta frase, Bill Gates, ens explica que l'automatització multiplica l'eficiència en qualsevol operació eficient, però multiplica la ineficiència en qualsevol operació ineficient. La domòtica n'és un exemple ja que si realment volem obtenir eficiència hem d'anar en compte en quina operació estem automatitzant perquè repetir una operació eficient multiplica l'estalvi però repetir una operació ineficient podria multiplicar el cost i malbaratar recursos.

Un cop acabada la introducció del projecte, em dispo a començar-lo amb moltes ganes.

## FONAMENTS TEÒRICS

### 1. Domòtica

#### 1.1. Definició

Segons el Diccionari de l'Institut d'Estudis Catalans es defineix com la disciplina que s'ocupa de la concepció i de l'aplicació d'automatismes en les instal·lacions dels habitatges.

És a dir, podem entendre el concepte domòtica com l'aplicació de la robòtica en l'àmbit domèstic.

La paraula domòtica prové de la unió de la paraula llatina “domus” (casa) i de “autònom” del grec αὐτόνομος; “que es governa a si mateix” ; o sigui l'automatització de la casa, del domus.

A títol pràctic, domòtica vol dir, per exemple, fer que una casa estigui automatitzada: que pugin les persianes, soni l'alarma, s'apagui la calefacció ( i un llarg etcètera) seguint les directrius que definim.

Per exemple, si li diem que les persianes del menjador baixin automàticament quan el nivell d'intensitat lumínica sigui considerablement elevat (especificant lluminositat), les persianes, al arribar tal punt de llum, es baixaran, però, mentre no s'assoleixi aquest nivell de llum, es mantindran pujades o en repòs.

De la mateixa manera podem manar que quan la llum ja no sigui tan intensa, la persiana es torni a pujar. Podem fins i tot fer que a la nit (quan el nivell de llum és molt baix) les persianes tornin a desplegar-se del tot, per evitar que ens entrin a robar.

## 1.2. Antecedents històrics

El concepte de domòtica fa referència a la disciplina tècnica que estudia i s'encarrega d'aplicar la ciència de la informació a l'àmbit casolà i domèstic. Ara bé, quan va sorgir aquest concepte? Com que aquesta informació només intenta donar una perspectiva històrica sobre els inicis de la domòtica per posar-ho en context amb el treball que vull fer, en aquest cas, transcripció literalment la informació que he consultat.

[“La primera tecnologia domòtica, l'X10 fou desenvolupada el 1975 per Pico Electronics a Glenrothes (Escòcia) per permetre el control remot d'aparells i aplicacions, i encara és de les més usades mundialment, a causa del baix preu dels seus components, tot i que existeixen nombroses alternatives amb més amplada de banda com KNX, INSTEON, BACnet o LonWorks.

Els primers sistemes comercials es van produir en la dècada de 1980 destinats a edificis terciaris, però a escala domèstica la integració comercial dels sistemes es va iniciar amb la popularització d'internet els anys 1990 al Japó, els Estats Units d'Amèrica i el nord d'Europa, fins llavors els electrodomèstics es dissenyaven i fabricaven sense tenir en consideració la possible interconnexió.

Les instal·lacions de domòtica domèstica evolucionen cap a la unificació de tots els sistemes en una única xarxa de comandament, creant el que s'anomena llar digital, requerint protocols d'actuació comuns i connexió a internet per poder comandar els serveis.” ]<sup>1</sup>

Una manera molt senzilla d'entendre la domòtica és fent un símil amb un interruptor. De fet, els interruptors són els precursors d'aquesta disciplina. Un interruptor és una peça que quan l'actues deixa passar l'electricitat, i quan el tornes a actuar, la atura. La domòtica busca això (i molt més) però d'una manera automatitzada mitjançant unes directrius depenent de les necessitats.

---

<sup>1</sup>

Informació extreta de <https://ca.wikipedia.org/wiki/Domòtica>.



La domòtica va néixer de la imperiosa necessitat popular del control de l'energia d'una manera senzilla, tal i com tenia l'enlluernat públic o les indústries. Aquestes tenien mecanismes que controlaven que el consum d'energia elèctrica no fos excessiu. El que va fer la domòtica és, bàsicament, portar aquests mecanismes als habitatges normals.

### **1.3.Aplicació**

Igualment com he fet a l'apartat anterior, he consultat a la web en quins àmbits s'aplica la domòtica i resumeixo i la informació que he trobat:

La domòtica és aplicada bàsicament sobre tres àrees:

- benestar
- estalvi
- seguretat

#### **Benestar**

Molts cops l'augment de comoditat en una llar es busca a través de la domòtica, fent, per exemple, que la porta del garatge s'apugi sola quan sigui necessari, però només si és el cotxe del propietari, o implementant un sistema pel que al caminar per la casa les llums es van encenent i apagant quan sigui adient.

#### **Estalvi**

De la mateixa manera, un sistema domòtic pot estalviar molts diners (i recursos energètics) fent que les operacions (a part d'automàtiques) siguin eficients en mesura del possible. Això vol dir que, per exemple, les llums de casa no estaran enceses quan no siguin necessàries, o més ben dit, s'apagaran automàticament quan hi hagi suficient llum com per no necessitar-les.

## **Seguretat**

Moltes instal·lacions automàtiques tenen implementades una sèrie de mesures de seguretat per tal de garantir que l'habitatge i sobre tot les indústries siguin el més segures possible. Un exemple d'aquestes funcions seria tenir un sensor de gas a la cuina i un sensor de fum al aparcament, de tal manera que si detecta una concentració massa elevada de gas o fum dispari una alarma, obri traus de ventilació i fins i tot truqui als bombers si detecta foc.

La domotització té com a objectiu l'optimització de recursos energètics, hídrics i/o econòmics. Molts cops l'automatització de vivendes, indústries, empreses o fins i tot de serveis públics com l'enllumenat, ve acompanyada, de implementació d'energies renovables i netes com l'eòlica, la solar o la mareomotriu (en una mesura molt menor)

Igualment, un sistema domèstic o industrial requereix de sistemes d'alimentació ininterrompuda (SAI), bateries per l'alimentació de sensors o actuadors i de centrals de processament com ordinadors, mòbils o fins i tot plaques com l'Arduino. Per implementar del tot un sistema domòtic, en molts casos, és necessària tenir connexió constant a la internet, així com (de vegades) certes nocions en electrònica per tal de ser autosuficients si mai sorgeix algun error al sistema.

## **1.4. Classificació**

És molt complicat oferir una classificació de sistemes domòtics, però hi ha factors que diferencien tots els sistemes entre sí. Aquí ofereixo un petit resum:

### ***1.4.1. Tipus d'arquitectura***

El tipus d'arquitectura pot ser centralitzat, descentralitzat o distribuït. El què classifica això és la manera en la que es centra la informació i el processament d'aquesta en el sistema en qüestió.

En un **sistema centralitzat** trobem el controlador central que és el que s'encarrega d'enviar la informació als actuadors i rebre-la dels sensors. Aquest sistema, però, té un problema, si falla el controlador central, falla tot el sistema.

En un **sistema descentralitzat** tots els controladors estan interconnectats, i alhora estan connectats amb els sensors i actuadors. D'aquesta manera s'aconsegueix una configuració menys sensible als errors i les fallides.

Per últim, en un sistema **distribuït** cadascun dels actuadors i dels sensors té la funció d'un controlador, fent així la seva pròpia feina i minimitzant les probabilitats d'error general, ja que si falla un component només es veurà afectada la funcionalitat que l'envolta.

#### ***1.4.2. Mitjans de transmissió***

Aquí és on comença la classificació que involucra més al consumidor directe: El tipus de transmissió d'informació.

Com és sabut, bàsicament hi ha dues tècniques de transmissió d'informació, la que comporta un cablejat i la sense fils; n'hi ha però, també, una altra de no tant coneguda que és la instal·lació per corrent portadora.

Un **sistema amb cablejat** és l'opció més fiable, més a prova d'errors. És la que té una latència de connexió més ràpida, i per tant necessita un temps de reacció més curt. Aquest sistema, però, és molt poc versàtil.

La **versió sense fils** del mateix sistema oferiria un temps de reacció més lent, disminuint la velocitat i augmentant la latència. Com a punt positiu tenim la versatilitat, ja que podem fer-la servir sense necessitat d'estar connectats amb un cable.

Per últim, la versió amb **instal·lació per corrent portadora**, és de les menys conegudes, aquesta versió aprofita les línies elèctriques de la casa on s'està realitzant el procés per portar la informació. Aquest sistema només necessita de dos endolls, l'emissor i el receptor. Com a inconvenient tenim que no es pot passar informació d'una instal·lació elèctrica a una altra.

## 1.5. Elements

Un projecte de domòtica consta de les parts següents:

- Sensors
- Controlador/s (Raspberry Pi en aquest cas)
- Actuadors
- Cablejat

Un **sensor** és un dispositiu capaç de mesurar magnituds físiques o químiques, i traduir-les a senyals elèctriques per tal que siguin completament objectives i les podem manipular.

Per exemple, un sensor de llum mesura la intensitat de llum que li arriba i la transforma en una senyal llegible per l'ordinador o pel controlador.

Un **actuador** és tot allò que realitza una acció o operació en una instal·lació quan rep l'ordre corresponent, normalment provinent d'un microcontrolador o un processador o microprocessador.

Un **controlador** o PLC (*Programmable Logic Controller*) és un ordinador capaç de processar informació per a una instal·lació automàtica domèstica o industrial. En aquest cas el nostre controlador seria la Raspberry Pi.

No s'han de confondre amb els controladors informàtics, ja que aquests són programes que et permeten interactuar amb un component o perifèric connectat a l'ordinador.

El **cablejat** és el medi pel qual transportem la informació, les dades i l'alimentació d'un lloc a un altre, i d'un element (de la nostra instal·lació) a un altre. (mitjans de transmissió).

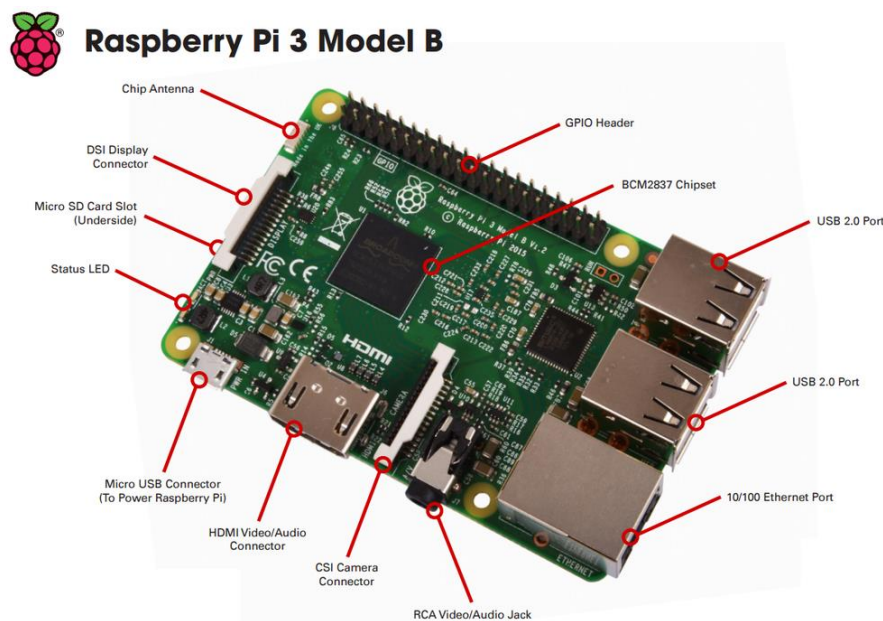
## 2. Raspberry Pi

### 2.1. Definició

La Raspberry Pi és un ordinador complet mono-placa de baix cost desenvolupat en el Regne Unit per la Fundació Raspberry Pi. L'objectiu principal d'aquest disseny és estimular l'ensenyança de les ciències de la computació.

Jo, per aquest projecte faré servir la Raspberry Pi 3 model B.

Aquesta placa disposa d'un xip WI-FI integrat, entrada ethernet amb boca RJ-45, disposa d'una entrada micro USB per alimentar la placa (que funciona a 5V i a 220 mA), disposa d'una sortida HDMI, igual que una sortida jack 3.5 per audio i video i quatre ports USB. A més a més disposa d'un slot per targetes Micro SD, on hi haurem d'instal·lar el Sistema Operatiu pertinent o el que desitgem. Per últim, aquesta placa té uns pins molt interessants que ens permeten controlar sensors i actuadors externs així com disposar de sortides de 3.3 o 5V i de connexions a terra.



[Figura 1: Raspberry Pi 3 Model B]

## **2.2. Antecedents històrics**

Aquest ordinador creat per la *fundació Raspberry Pi* és molt famós. Va sortir a la venda per primer cop l'any 2009, el model que es va vendre va ser la Raspberry Pi, seguida de la Raspberry Pi model B, i, en els anys següents, es van desenvolupar i vendre els models Raspberry Pi model B+, Raspberry Pi 2, Raspberry Pi Zero, seguida de la Raspberry Pi Zero W i de la Raspberry Pi Zero WH.

Després es van crear els models Raspberry Pi 3, Raspberry Pi 3 model B, i l'any 2018 la Raspberry Pi 3 model B+.

## **2.3. Llenguatges de la Raspberry Pi**

La fundació *Raspberry* dona suport per a descàrregues de diferents distribucions Linux adaptades a l'arquitectura ARM: Raspbian (derivada de Debian), RISC OS 5, Arch Linux ARM (derivat d'Arch Linux) i Pidora (derivada de Fedora). Com a eina d'ensenyament de programació promouen principalment el llenguatge de programació Python, però també altres llenguatges com ara Tiny BASIC, C, Perl i Ruby

## 3. Programació

### 3.1. Llenguatges de programació

Un llenguatge de programació és un llenguatge formal amb el que et pots comunicar amb un ordinador perquè pugui entendre les teves ordres i processar les dades corresponents tal i com li manis. Normalment és un llenguatge basat en l'anglès amb una sintaxi creada expressament perquè no hi hagi marge d'error. Els ordinadors tradueixen aquest llenguatge al codi binari a través d'un compilador per posteriorment executar el programa.

L'execució d'un programa és un procés rigorós en el que l'ordinador no fa cap tipus de suposició, sinó llegir les teves ordres codificades i actuar en conseqüència.

Aquest llenguatge de programació es fa servir dins un tipus de processador de textos. Pot ser un .txt, un «word» o, i preferiblement, un IDE.

Un IDE és l'entorn que es fa servir per programar d'una manera relativament còmode. Els avantatges que aquest entorn t'aporta, vers un processador de textos qualsevol, és la gran varietat d'eines i funcionalitats que es sumen a la diferenciació sintàctica cromàtica per facilitar la revisió del codi i que tot estigui correcte.

Amb la diferenciació cromàtica, podem aconseguir que un codi enlloc de veure's així:

```
import time
print(time.strftime
      ('%03:49:14'))
print(time.strftime
      ('%21|06|17'))
```

Es vegi així:

```
import time
print(time.strftime
      ('%03:49:14'))
print(time.strftime
      ('%21|06|17'))
```

Grans exemples de IDE coneguts son: Atom, Eclipse, Netbeans, Geany, IntelliJ...

La funcionalitat més important que et brinden alguns dels IDE és el «debugger». Aquesta eina et deixa executar el programa pas per pas, veient les variabes i els càlculs en temps real i manant tu quan vols que passi al següent pas, t'ho relaciona amb la línia de codi que està executant, per, d'aquesta manera, trobar les falles del codi d'una manera molt més ràpida i fàcil.

### 3.2. Llenguatge utilitzat

El llenguatge que he fet servir és el Python, encara que en algun moment hagi recorregut a la complicada sintaxis del famós llenguatge C.

El Python és un llenguatge de programació d'alt nivell i rendiment i amb una base de propòsits molt ample. Tot i l'amplària d'aquesta base, és capaç de dur a terme qualsevol feina d'una manera eficient i ràpida.

Aquest llenguatge es caracteritza, sobretot, per fer ús d'una sintaxis molt senzilla, poc més complicada que la del llenguatge parlat comú. Està basat en l'anglès, i es podria considerar una evolució (molt positiva) del llenguatge C o del C++ doncs hereta moltes coses positives i es desfà de la sintaxis feixuga.



Jo faré servir el IDE Geany, pensat per molts llenguatges i sintaxis, molt personalitzable i versàtil.

És molt important destacar que, com el Python és un llenguatge de programació de sintaxis senzilla i pensat per ser usat en un document de text, sense IDE, no disposa de compatibilitat amb cap debugger, el que podria dificultar les coses en un futur.

### 3.3. Linux i terminal

Aquí és quan les coses sembla que es compliquin, perquè si bé és cert que córrer un programa amb el Raspbian no és tant senzill com amb l'Arduino (que és donar-li a un botó i llestos), realment no resulta tant difícil.

En primer lloc hem d'entendre com funciona el Linux i el seu sistema d'organització jeràrquica de carpetes. Aquest sistema és bastant semblant al de Windows, i de fet, el que farem ara també es pot fer al Windows, però no és tant freqüent. Per tal d'accedir al nostre programa i executar-lo d'una manera que no sobrecarregui la Raspberry Pi, l'únic que hem de fer és navegar pels repertoris que ens facilita la Raspberry Pi dins el terminal (o consola).

Ens movem de carpeta en carpeta fent servir el comandament «cd», veiem què hi ha a la carpeta en la que estem amb el comandament «ls» i mirem en quina ubicació estem amb el comandament «pwd». Amb això hauria de ser més que suficient. Per acabar, és necessari que introduïm la línia de comandaments següent per tal de córrer el programa (si el nostre programa es diu `programa1.py` i es troba a la mateixa carpeta en la que estem):

- «**sudo**» (que marca que volem tenir drets d'administrador)
- «**python**» (que marca que el fitxer que executem és un arxiu \*.py)
- «**programa1.py**» (és molt important posar el nom complet de l'arxiu així com la seva extensió)

Així doncs la línia de comandaments ens quedarà així:

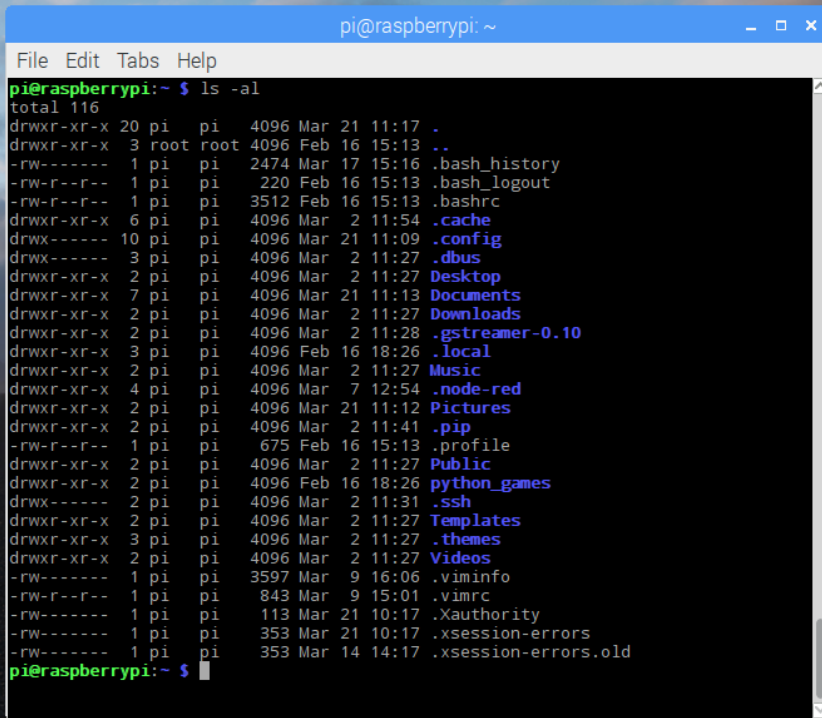
**“sudo python programa1.py”**

En el nostre cas hi hem d’afegir un «1» darrera perquè tenim un motor pas a pas al programa al qual li hem de marcar la velocitat.

**“sudo python programa1.py 1”**

Llavors s’obrirà la configuració del programa al terminal i s’executarà.

A continuació insereixo una imatge de la consola de la Raspberry Pi quan opera sobre Raspbian.

A screenshot of a terminal window titled 'pi@raspberrypi: ~'. The window has a menu bar with 'File', 'Edit', 'Tabs', and 'Help'. The terminal shows the command 'pi@raspberrypi:~ \$ ls -al' and its output, which is a detailed directory listing of the home directory. The output includes permissions, owner, group, size, date, and filename for various files and directories like ., .., .bash\_history, .bash\_logout, .bashrc, .cache, .config, .dbus, Desktop, Documents, Downloads, .gstreamer-0.10, .local, Music, .node-red, Pictures, .pip, .profile, Public, python\_games, .ssh, Templates, .themes, Videos, .viminfo, .vimrc, .Xauthority, .xsession-errors, and .xsession-errors.old.

```
pi@raspberrypi:~ $ ls -al
total 116
drwxr-xr-x 20 pi pi 4096 Mar 21 11:17 .
drwxr-xr-x 3 root root 4096 Feb 16 15:13 ..
-rw-r----- 1 pi pi 2474 Mar 17 15:16 .bash_history
-rw-r----- 1 pi pi 220 Feb 16 15:13 .bash_logout
-rw-r----- 1 pi pi 3512 Feb 16 15:13 .bashrc
drwxr-xr-x 6 pi pi 4096 Mar 2 11:54 .cache
drwx----- 10 pi pi 4096 Mar 21 11:09 .config
drwx----- 3 pi pi 4096 Mar 2 11:27 .dbus
drwxr-xr-x 2 pi pi 4096 Mar 2 11:27 Desktop
drwxr-xr-x 7 pi pi 4096 Mar 21 11:13 Documents
drwxr-xr-x 2 pi pi 4096 Mar 2 11:27 Downloads
drwxr-xr-x 2 pi pi 4096 Mar 2 11:28 .gstreamer-0.10
drwxr-xr-x 3 pi pi 4096 Feb 16 18:26 .local
drwxr-xr-x 2 pi pi 4096 Mar 2 11:27 Music
drwxr-xr-x 4 pi pi 4096 Mar 7 12:54 .node-red
drwxr-xr-x 2 pi pi 4096 Mar 21 11:12 Pictures
drwxr-xr-x 2 pi pi 4096 Mar 2 11:41 .pip
-rw-r----- 1 pi pi 675 Feb 16 15:13 .profile
drwxr-xr-x 2 pi pi 4096 Mar 2 11:27 Public
drwxr-xr-x 2 pi pi 4096 Feb 16 18:26 python_games
drwx----- 2 pi pi 4096 Mar 2 11:31 .ssh
drwxr-xr-x 2 pi pi 4096 Mar 2 11:27 Templates
drwxr-xr-x 3 pi pi 4096 Mar 2 11:27 .themes
drwxr-xr-x 2 pi pi 4096 Mar 2 11:27 Videos
-rw-r----- 1 pi pi 3597 Mar 9 16:06 .viminfo
-rw-r----- 1 pi pi 843 Mar 9 15:01 .vimrc
-rw-r----- 1 pi pi 113 Mar 21 10:17 .Xauthority
-rw-r----- 1 pi pi 353 Mar 21 10:17 .xsession-errors
-rw-r----- 1 pi pi 353 Mar 14 14:17 .xsession-errors.old
pi@raspberrypi:~ $
```

[Figura 2: Consola o terminal Raspberry Pi]

## FONAMENTS PRÀCTICS

### 4. Descripció del projecte

Com ja he explicat fins ara, els objectius que engloba aquest projecte es poden agrupar en un de sol: domotitzar una casa mitjançant una placa de baix cost com ho és la Raspberry Pi i veure si podria ser més o menys viable.

Per això, aquest projecte consistirà en els punts següents:

- Introducció i posada a punt de la Raspberry Pi.
- Definir les funcions que ha de tenir la casa i els sensors que necessitem.
- Disseny i maquetatge de la casa.
- Construcció de la casa.
- Programació i integració dels sensors i actuadors a la maqueta.

Al final del projecte veuré quines de les dificultats que se m'han presentat he pogut superar, quines he hagut d'evitar i faré un resum de tot fins a donar per acabat el treball de recerca.

A més a més aniré fent un seguiment del cost aproximat del projecte.

Per començar he d'aclarir exactament el que pretenc fer i, per això, he de determinar quines funcions automàtiques vull instal·lar a l'habitatge. La meva idea és, en un primer moment:

- fer un garatge que permeti l'entrada automàtica del vehicle del propietari (amb un bloqueig mitjançant alarma),
- un detector de gas a la cuina
- un detector de moviment per infrarojos (PIR) que ens avisi quan hi ha algun moviment no desitjat o esperat a casa nostra
- un sensor d'IR (infrarojos) per tenir el control total de l'alarma.

D'igual manera, una instal·lació domòtica, requereix d'un factor internet molt important. Així doncs intentaré desenvolupar també:

- una espècie de plataforma per a Android que es pugui connectar intranacionalment
- un sistema d'avís per notificacions a l'smartphone.

Com que aquestes son les meves idees inicials restringiré la maqueta que construiré a una versió de vivenda que només disposi d'un garatge, un hall, una cuina i d'un saló (on pretenc donar refugi als cables).

Entenent que el meu projecte es veu limitat per la quantitat de recursos econòmics que són necessaris, la gran quantitat de temps necessària i sobretot les limitacions físiques que m'aporta la Raspberry Pi, com ja he avançat, el meu objectiu de treball no és fer un habitatge realista, sinó una maqueta on pugui demostrar què es pot fer, encara que no sigui cent per cent extrapolable a un habitatge real. No pretenc fer un prototip a petita escala d'un habitatge real (perquè seria massa costós en tots els aspectes) però el que vull és demostrar que és possible fer-ho a casa sense gaire ajuda.

Per exemple, seria molt senzill fer que totes les finestres de la maqueta tinguessin unes persianes motoritzades que s'apugessin i baixessin soles, però la Raspberry Pi té un nombre bastant limitats de pins GPIO (que són els que s'usen per a donar entrades i sortides digitals de la Raspberry Pi) i cada motor n'ocupa sis, així doncs no és viable posar gaires motors. Ho substituiré tot per un sol motor per demostrar que es pot fer ja que el codi seria gairebé el mateix.

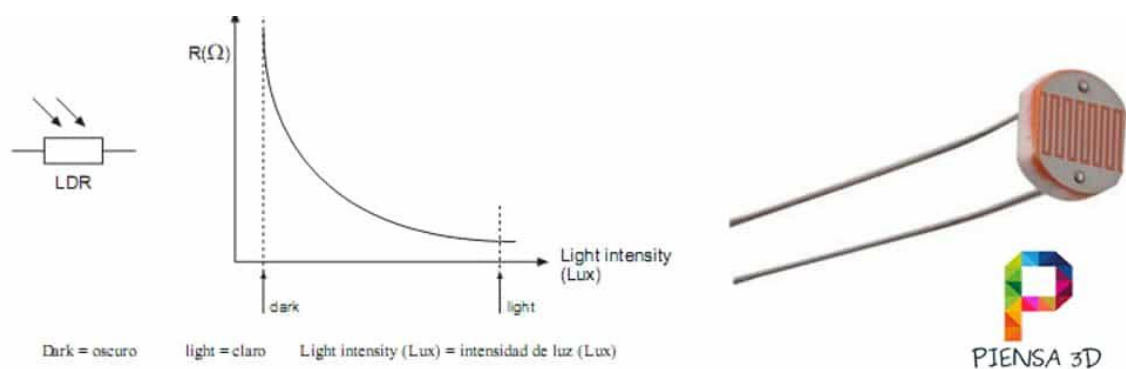
## 5. Maqueta

### 5.1. Materials

En aquest projecte, per aconseguir assolir les meves fites usaré els següents sensors i actuadors.

- LDR (Light Dependant Resistor)
- LED (Light Emitting Diode)
- Stepper (Motor pas a pas)
- Detector de gas/fum
- Receptor IR
- Emisor IR
- PIR

L'**LDR** té una particularitat: pateix canvis en la seva resistivitat depenent de la intensitat lumínica que rep. Quan rep molta llum deixa passar el corrent elèctric amb molta més facilitat que quan en rep menys.

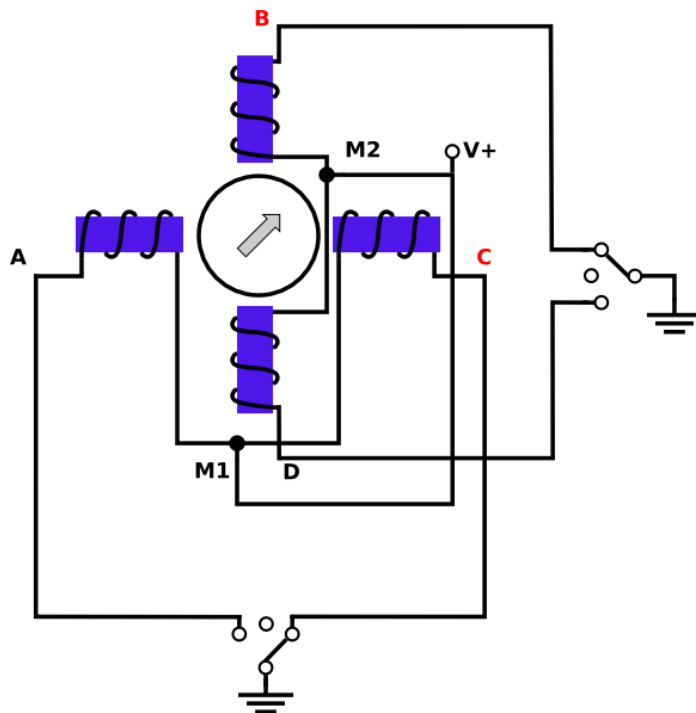


[Figura 3: funció LDR]

El **LED** és un díode (només deixa circular el corrent elèctric en un sentit) que al estar influenciat per l'electre produeix llum. Normalment aquesta tecnologia s'usa perquè el

consum d'aquests llums sol ser molt inferior al de les bombetes incandescent tradicionals, produint la mateixa (o més) quantitat de llum.

**L'stepper** és un motor que es mou (com el seu nom indica) pas a pas. És a dir, si en un stepper el recorregut complet de  $360^\circ$  està format per 38 passos, pots fer que l'stepper faci exactament mitja volta (19 passos), exactament  $30^\circ$  o cinc voltes senceres o simplement un pas. Ens ofereix un gir precís mitjançant un conjunt d'electroimants, i dents cisellades sobre el cos interior de l'stepper.



[Figura 4: Esquema motor pas a pas]

El **detector de gas o fum** pot detectar mitjançant tècniques òptiques (digitals o analògiques) o mitjançant tècniques de diferenciació iònica la presència de fum o de certs gasos en la nostra atmosfera. Nosaltres, però, farem servir sensors de gas (per detectar  $\text{CO}_2$  o  $\text{C}_4\text{H}_{10}$ ) mitjançant semiconductors. En aquests sensors, una reacció química té lloc quan el gas en qüestió hi entra en contacte.



[Figura 5: Sensor de gas]

En el nostre cas, el sensor de gas és un sensor classificat com a MQ-2, amb capacitat per a detectar alguns gasos inflamables (com ara metà, butà, alcohol, oxigen, nitrogen...) o diòxid de carboni. Això fa d'aquest sensor l'ideal per a col·locar tant a la cuina com al garatge.

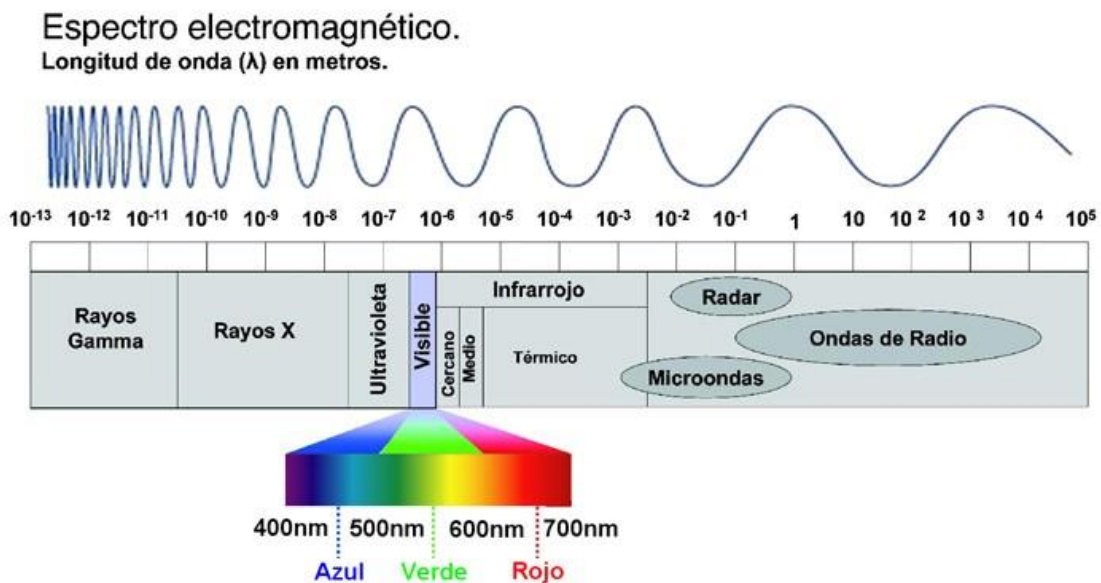
Però com pot ser que aquest sensor detecti tant oxigen i nitrogen, que són dos gasos molt presents en l'atmosfera, i gasos com el metà? Doncs perquè per detectar metà o butà, la concentració que es necessita és baixíssima, en canvi, per detectar oxigen o nitrogen les ppm (parts per milió) necessàries són molt elevades.

Aquest sensor concretament necessita un voltatge d'entre 4,9 i 5,01 Volts. Al ser un sensor que tot i funcionar correctament pateix un escalfament exagerat, no és gaire bona idea superar els 5 volts, i una bona opció és posar-li una placa de dissipació tèrmica de coure o alumini. sempre que aïllem les potes del sensor amb una mica de Plastidip o de qualsevol plàstic fixe, no hauria de passar res.

El **Receptor d'IR** (d'infrarojos) ens dona la capacitat de rebre i interpretar senyals en una freqüència menor que la de la llum visible, és a dir, una longitud d'ona d'entre 0.7 i 1000 micròmetres.

La radiació infraroja és emesa per qualsevol cos que estigui per sobre dels 0° Kelvin, és a dir, per sobre del 0 absolut, dels -273.15 °C

L'**emissor d'IR** (d'infrarojos) ens deixa enviar ordres amb llum infraroja, ens deixa enviar informació d'una manera molt còmode i que no sigui visible a la vista.



[Figura 6: Espectre de la llum visible]

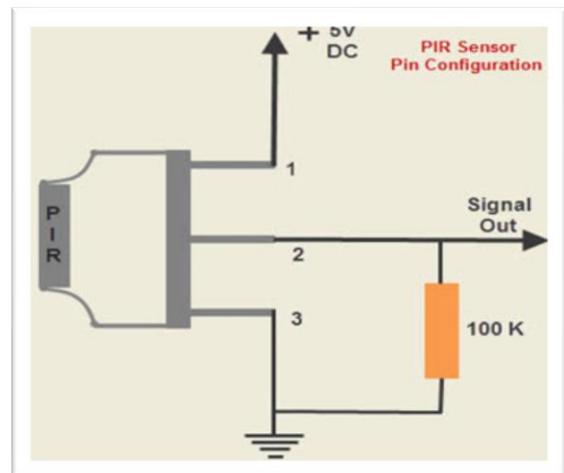
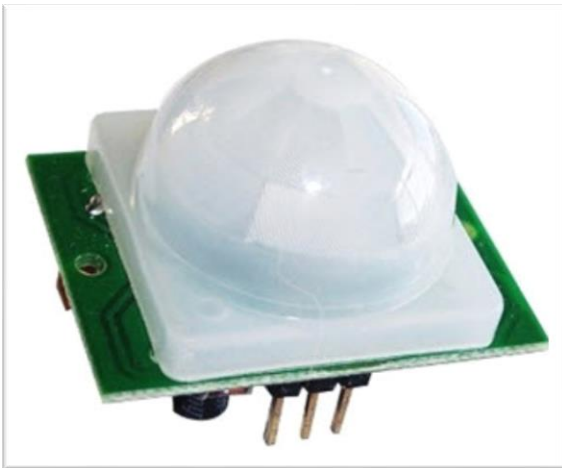
El sensor PIR o Passive Infrared Sensor és un component electrònic que detecta els rajos infrarojos que emet qualsevol cos. Com ja he explicat abans, qualsevol cos que estigui per sobre del zero absolut emet rajos infrarojos, i aquests poden ser fàcilment detectats.

Aquest aparell fa justament això, s'ocupa de rebre tots els rajos infrarojos dins l'espectre de distància que li especifiquem mitjançant un potenciòmetre, i quan en detecta de suficient intensitat ens avisa amb una senyal HIGH (3,26 volts, virtualment 3,3 volts).



El model de PIR que vaig comprar és bastant versàtil ja que accepta una alimentació amb una alta tolerància (5 – 25 volts) així que no tindria cap inconvenient en alimentar-lo encara que els pins de 5 volts de la Raspberry Pi estiguessin tots ocupats.

A més, aquests sensors passius d'infrarojos tenen un angle de detecció de 120 graus aproximadament, això vol dir que la peça de plàstic que cobreix el receptor té una espècie de dibuixos amb una funció molt important: concentrar tots els rajos que venen dels 120 graus en un punt molt més petit.



[Figures 7 i 8: Sensor passiu d'infrarojos]

## 5.2. Plànols

Abans de fer els plànols, o tan sols de començar a pensar com serà de disseny la nostra maqueta, hem d'aconseguir arribar a alguna conclusió que ens ajudi a l'hora de dissenyar el que serà la nostra maqueta.

Jo tinc clar que vull fer:

- Un pàrquing
- Una cuina
- Un saló
- Un Hall

Sabent això ens podem posar a pensar en com de gran la volem. Jo vull una cosa que no sigui massa gran, que sigui senzilla i sobretot funcional.

El nivell estètic, no serà la meva prioritat ja que el que busco es poder implementar un nivell d'automatització en una maqueta, així que segurament no pararé a parlar gaire sobre l'aspecte d'aquesta.

Arribat a aquest punt, on em dispo a començar a dissenyar la maqueta, me'n adono que necessito una manera; és clar que podria fer-ho a ma, agafar regle, escaire i cartabó i començar a dissenyar-la, però segurament els resultats no serien òptims.

Així doncs he optat per fer servir un programa molt conegut, l'AutoCAD, en la seva versió 2019. L'únic inconvenient és que es tracta d'una eina molt completa, però extraordinàriament cara i totalment inassequible, per a mi. Buscant una solució vaig accedir al programa d'AutoCAD per estudiants.

Primer dissenyaré la casa com si hagués de ser una casa real, com si algú hi hagués de viure.

Es poden visionar els plànols originals, sense caixetí ni retocs al següent enllaç:

[https://drive.google.com/drive/folders/1YBRwxNU9nuJCHU15GRZH2sK\\_BnME3UMg?usp=sharing](https://drive.google.com/drive/folders/1YBRwxNU9nuJCHU15GRZH2sK_BnME3UMg?usp=sharing)

Aquest és el disseny que he trobat més adient.

A l'apartat d'Annexos inseriré els plànols que expressen la mida real que tindrà la maqueta en mil·límetres. Annex núm. 1

És molt important destacar que els plànols originals, tal com s'exporten de l'AutoCAD, surten amb una marca que indica que el programa que uso és una versió gratuïta per estudiants. Jo he retocat aquests plànols amb un processador d'imatges i posteriorment els he afegit un caixetí.

Això vol dir que encara que no es vegin les marques als plànols que inseriré, no vol dir que hagi comprat una llicència, és més, deixo a l'enllaç anterior els plànols sense retocar, és a dir, amb marca d'AutoCAD i sense caixetí.

### **5.3. Construcció**

La maqueta la elaboraré amb planxes de contraxapat de fusta de 0.5 cm de gruix, 80 cm de llarg i 50 cm d'ample. A més a més faré servir una planxa de contraxapat de les mateixes dimensions però d'1 cm de gruix per elaborar una base rígida i resistent on cargolar la Raspberry Pi i tot el cablejat necessari.

A més, faré servir una vara d'alumini per a assegurar alguns sensors i actuadors i un tub de Policlorur de Vinil així com unes cantoneres del mateix material.

Faré servir una Dremel 3000 i una serra circular de mà Worx per elaborar la maqueta.

La maqueta, que és la base per a muntar els sensors i demostrar el funcionament que he programat, la vaig fer al magatzem del meu avi perquè la mida és considerable. El cablejat, en canvi, el vaig anar connectant a casa meua, on tinc wifi, el portàtil i podia anar fent a estones. De moment tenia l'estructura separada de la part electrònica.

En el procés de muntar la maqueta, és relativament important parlar de seguretat: totes les màquines que vaig fer servir superaven, en potència màxima, els 87 dB. Segons el Reial Decret 286/2006 de 10 de març, sobre la protecció de la salut i la seguretat dels treballadors contra els riscos relacionats amb l'exposició al soroll, una exposició prolongada més de 8 hores a una intensitat acústica de 87 dB pot ser nociva per la salut auditiva, per tant, vaig fer servir taps de protecció per a les orelles. D'igual manera, la potència de la maquinària que vaig usar reclamava l'ús d'accessoris de protecció com guants i ulleres.

Després de superar els processos de disseny i compra de material, va ser relativament senzill muntar la maqueta. Ajuntant les parets entre si amb claus petits i amb cantoneres de Policlorur de Vinil i enganxant els envans exteriors al terra amb clauets i cola blanca i, posteriorment, enganxant els envans interiors al terra i als exteriors amb claus, grapes i cola blanca.

Els procés de retallar les peces de fusta va ser més complicat de l'esperat, doncs el meu objectiu era optimitzar la fusta al màxim; després de diversos càlculs i proves amb les eines, serrar les peces de fusta i PVC va ser relativament fàcil.

Més endavant, però, se'm va presentar un problema: La meva Dremel va morir sobtadament. Després d'investigar què podria haver sigut, vaig arribar a la conclusió que, molt probablement, era el bobinatge del motor de la Dremel, que s'havia trencat. Vaig comprovar que era això, i després d'arreglar-la, se'm va tornar a trencar. Com que les Dremels son aparells cars, vaig decidir comprar-me'n una de més assequible.

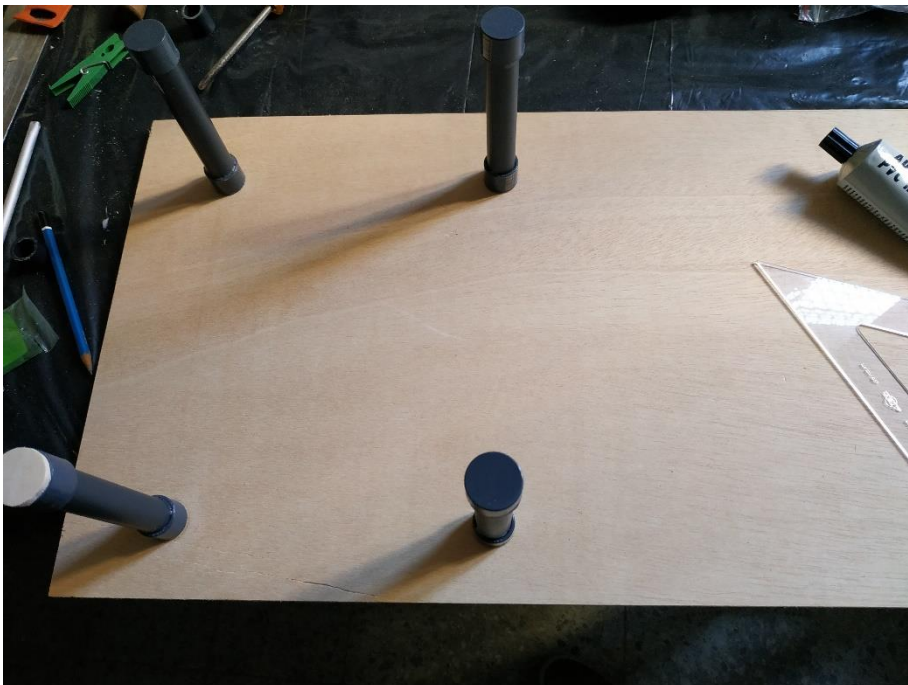
Un cop superat el procés de construcció vaig decidir que no pintaria la maqueta, doncs encara quedava bastanta feina per fer, i l'estètica de la maqueta no és el punt més important per a mi.

El primer problema que se'm va presentar quan ja tenia la maqueta acabada i muntada va ser que la fusta de contraxapat té una energia superficial molt elevada. L'energia superficial es

defineix com l'energia necessària per a trencar els enllaços intermoleculars per tal de crear una superfície.<sup>2</sup>

El PVC té una peculiaritat, no s'enganxa amb qualsevol tipus de cola o adhesiu.

Què passa doncs? Que la meua base està elevada per tal de poder encabir la Raspberry Pi i tots els cables. Per elevar-la vaig utilitzar tubs de PVC, que s'han d'enganxar a la fusta. Això va ser especialment complicat, doncs la cola blanca que és perfecta per la fusta no enganxa el PVC i la cola de PVC que és perfecta pel PVC no enganxa la fusta (amb la fusta). Curiosament, la cola de PVC enganxa (lleugerament) la fusta i el PVC entre si; no és una unió gaire fiable, però assegurant-ho amb uns tirafons queda suficientment resistent com per a poder-ho manipular 10 o 20 minuts després i així ho vaig fer. Vaig posar quatre columnes i una de recolzament, de manera que va quedar com un soterrani per a poder instal·lar el cablejat i la Raspberry Pi.



[Figura 9: Elevacions de la maqueta]

---

<sup>2</sup>

Contingut extret de la wikipèdia: [https://es.wikipedia.org/wiki/Energ%C3%ADa\\_superficial](https://es.wikipedia.org/wiki/Energ%C3%ADa_superficial)

Quan tenia la maqueta amb el soterrani preparat vaig haver de reubicar tots els cables que tenia connectats per separat.

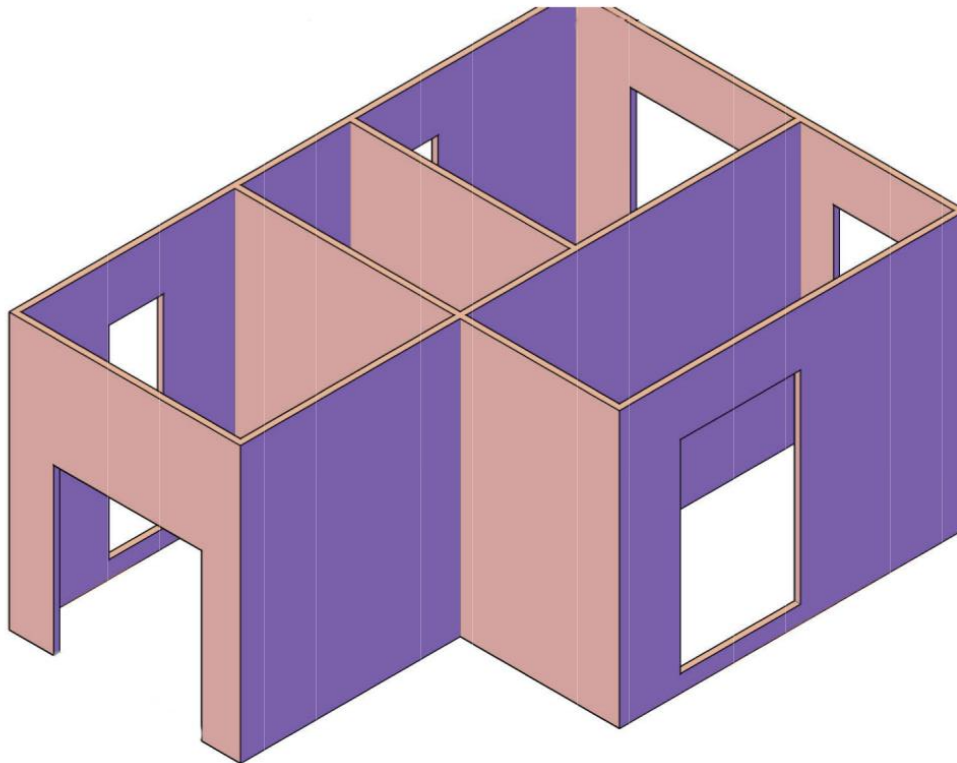
Vaig cargolar la Raspberry Pi i alguns dels sensors a la maqueta, alguns els vaig enganxar amb cola (prèviament m'havia assegurat que fos dielèctrica) i altres simplement vaig encaixar-los a la fusta per un parell de forats per on vaig fer anar els sensors. Va ser un procés que vaig haver de fer ben poc a poc, perquè per a traslladar els cables havia de desconnectar-los un a un i tornar-los a connectar respectant escrupolosament l'ordre.

El cost aproximat de tot el material de construcció va ser d'uns 53 euros (comptant la compra de la nova «dremel»)

Per tal de fer un seguiment aproximat del cost del treball a l'annex 2 es pot trobar les despeses econòmiques que ha suposat aquest treball.

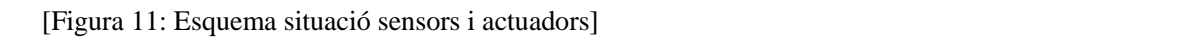
Com aprenentatge en aquest apartat de construcció de la maqueta vull compartir una famosa frase que diu: *Measure twice, cut ones...*

La maqueta (sense comptar el suport inferior ni la cavitat pels cables) hauria de quedar així, segons els meus plànols passats a una perspectiva 3D mitjançant AutoCAD:



[Figura 10: Perspectiva de la maqueta]

Vaig considerar molt important, al moment de dissenyar la maqueta, començar a planejar on volia instal·lar els sensors o actuadors ja que els motors pas a pas són bastant grans considerant les dimensions de la maqueta; així que vaig fer aquest esquema en el que explico, més o menys, on instal·laré els sensors i actuadors que tenen més volum (i rellevància).





## **6. Programació i electrònica**

### **6.1. Posta a punt de la Raspberry Pi**

La placa Raspberry Pi ens arriba a casa en blanc, o més ben dit sense sistema operatiu ja que és molt important destacar que NO disposa de memòria interna i s'abastirà de memòria externa mitjançant la Micro SD i sticks USB.

Jo, la vaig comprar per Amazon, en el mateix paquet em venia la Raspberry Pi 3 B, la seva carcassa, el seu carregador amb entrada anglesa i europea, la micro SD amb el NOOBS instal·lat i un adaptador SD..

Però és clar, la Raspberry Pi és un ordinador, no té pantalla, ni teclat, ni ratolí, ni tan sols té un bon sistema de dissipació de calor. La Raspberry Pi és un ordinador que pateix moltíssim de sobre-escalfament, es pot portar al límit amb molta facilitat i que la CPU estigui, quan te'n adonis, a més de 80 °C. Això es un gran problema que s'ha de solventar. La CPU d'aquesta placa està preparada per treballar entre 60 i 80 °C, assolint com a màxim els 95 - 100 °C si es tracta d'una emulació de consola retro. A partir del 90 °C, però, la placa pot patir conseqüències negatives i / o no tornar a funcionar adequadament.

El problema de l'escalfament de la Raspberry Pi es pot solucionar fàcilment amb uns dissipadors de coure fets especialment. Són bastant fàcils de trobar així que no hi haurà cap problema.

Com que es tracta d'un ordinador necessitem un monitor HDMI, un ratolí i un teclat per USB. El principal problema amb això és que es tracta de compres molt cares i no ens aportaria una experiència gaire pràctica, ja que sempre hauríem d'estar connectats a la xarxa d'alimentació tant per la Raspberry Pi com pel monitor... A més, el monitor és molt gran. Així que aquesta idea va quedar descartada. En canvi podria aprofitar tant el teclat i ratolí del meu ordinador de torre com el seu monitor.

En aquest punt se'm van presentar els primers problemes. La Raspberry Pi disposa de sortida de vídeo/àudio en HDMI, senyal digital, però el meu monitor funciona amb entrada VGA,

senyal analògica. Per solucionar això (i tenint en compte que amb el mateix monitor també hem de poder fer anar l'ordinador de torre) vam comprar unes quantes coses:

- Switch HDMI bi-direccional (2→ 1) (1→ 2)
- Joc de tres cables HDMI
- Adaptador (HDMI → VGA)

Amb això podem fer el següent: Mitjançant el Switch li podem dir “ara deixa'm veure la Raspberry Pi, ara deixa'm veure l'ordinador” en una mateixa pantalla i només fent un *click*. Amb els cables HDMI què n'hem de fer? doncs ja que el switch és HDMI i que la Raspberry Pi funciona amb aquesta sortida digital hem de connectar un cable HDMI a la sortida de la placa, un altre a la sortida de l'ordinador, i l'altre a l'entrada VGA que hem convertit mitjançant l'adaptador, que ens converteix la senyal digital en analògica perquè el monitor la pugui interpretar correctament.

Immediatament després de pensar i provar aquesta solució vaig pensar en un inconvenient que té... S'ha d'estar sempre amb un monitor, i això redueix bastant les nostres possibilitats de moure'ns amb la Raspberry Pi.

## **Establiment d'un servidor i un visualitzador**

Doncs això és el que vaig fer: vaig agafar la meua Raspberry Pi (amb sistema operatiu base Linux, Raspbian, Debian) i vaig agafar el meu portàtil petit (amb sistema operatiu base Linux, Linuxmint, Ubuntu, Debian) i vaig fer que la Raspberry Pi fos el servidor que enviés les dades i el portàtil fos el monitor que les rebés.

- 1r problema: La pantalla del portàtil era massa petita fins per la Raspberry Pi.
- 2n problema: Després de moltíssims intents no vaig aconseguir establir una IP estàtica fixe (eth0) al portàtil, cosa necessària per la comunicació.
- 3r problema: No disposava d'un cable ethernet mascle-mascle RJ45.

Els dos primers problemes van ser tant fàcils de solucionar com canviar el portàtil per algun altre. I la solució al tercer problema era.. bé... comprar-me un cable RJ-45 mascle-masclé.

Ara, vaig connectar tot el plec de cables que comunicaven la Raspberry Pi amb el monitor del meu ordinador i em vaig posar a instal·lar i configurar el programa. No crec que faci falta entrar en detalls, ja que entres a internet i et surten molts tutorials de com fer-ho, i la veritat és que treballar amb la consola o el terminal és força simple a aquestes alçades amb una mica d'ajuda.

El programa que hem d'instal·lar a la Raspberry Pi és el `tightvncserver` mitjançant aquestes dues ordres:

```
sudo apt-get update
```

```
sudo apt-get install tightvncserver
```

I el programa que hem d'instal·lar al portàtil el trobarem a <http://www.realvnc.com/en/connect/download/viewer/> i en cas que ens donés algun problema al entrar-hi, canviem el “en” de la URL per un “es” i llestos!

Triem el sistema operatiu i l'arquitectura adients i l'instal·lem.

Llavors només hem de posar a la Raspberry Pi , al terminal:

```
ifconfig eth0 o ifconfig i hi busquem eth0 manualment.
```

ens quedem amb la I.P inet addr i els aparellem, ja podem veure la Raspberry Pi des del portàtil mitjançant el cable ethernet!

Ara que ja podem visualitzar la Raspberry Pi des del portàtil, és a dir ja tenim un monitor portàtil per aquesta, ja podem començar a desenvolupar el projecte d'una manera més o menys còmode.

## 6.2. Programació

La programació em va donar diversos problemes, però, després de pensar-hi molt, i de visitar alguns fòrums, els he pogut solucionar tots (menys un).

Aquest problema que no vaig poder resoldre és el de connectar a la Raspberry Pi el sensor passiu d'infrarojos o PIR, en cap moment va sortir bé aquesta part del projecte, sempre m'enviava senyals altes de 3.3 volts, quan en principi me n'havia d'enviar de com a molt 1 volt. Això va ser un problema que encara ara no sé com solucionar. La “solució” que hi he trobat és fer del sistema de detecció d'intrusos un apèndix independent de la Raspberry Pi i del sistema domòtic fent-lo analògic. Això significa que el sistema del PIR i la Raspberry Pi no tenen res a veure, que pot funcionar un sense l'altre. Deixo el circuit que he muntat per aquest sensor passiu a l'Annex número 3.

Això és un inconvenient i un punt positiu alhora: Al ser analògic no hi ha cap risc d'error si la Raspberry Pi falla, però al no estar connectat de cap manera amb la Raspberry Pi, no puc enviar ni rebre senyals digitals d'un lloc a un altre, el que limita molt les possibilitats d'automatització o fins i tot les redueix a nul·les.

Aquest sistema analògic l'he realitzat mitjançant una PCB soldable i fent ús d'un transistor, un LED i un brunzidor (de 12 volts) en forma de díode passiu. Tot això alimentat per una pila de 9V. (veure el circuit a l'Annex número 3)

Un cop explicat el problema que vaig tenir, comencem de veritat amb la programació, que és, per mi, el més interessant.

D'entrada, he de dir que he desenvolupat el programa d'una manera que m'ha resultat molt útil.

Primer de tot vaig crear un fitxer «.py» (extensió .python) on aniria escrivint el codi final. Després, mitjançant molts altres documents de python, vaig anar escrivint el codi de cada sensor, cada millora, cada prova, cada implementació nova; en un document nou. D'aquesta manera no he perdut cap programa, les implementacions noves sempre les he pogut tirar enrere si no m'han convençut, i, més important, tenia tot el codi explicat, per junt i separat.

Així doncs, dins la carpeta «TDR», pots trobar la carpeta «TDR\_proves», i dins aquesta, la carpeta «LDR», «LDR+garatge», «IR»... dins de cada una d'aquestes carpetes, hi he guardat els seus respectius programes.

Quan decidia que una implementació era definitiva o volia provar-la, creava un fitxer nou on enganxava tot el codi que ja tenia desenvolupat i hi afegia la part nova.

He d'esmentar de manera particular la influència de dos fòrums en la meva programació, sobre tot al començament, quan acabava de començar a programar en Python amb les funcionalitats i peculiaritats Raspberry Pi . Aquests dos fòrums son:

<https://stackoverflow.com/>

<https://www.raspberrypi.org/forums/>

Accedia a cada un d'aquests fòrums si, i només si, portava un temps encallat en la programació (o electrònica també en el cas del fòrum Raspberry Pi) i ja havia consumit els meus recursos, llavors demanava ajut a persones amb més experiència (o fins i tot a alguns enginyers que estan disposats a ajudar de manera completament altruista). Normalment accedia al *stackoverflow* quan necessitava ajuda estrictament de codi, i al fòrum de la fundació Raspberry Pi quan era un problema focalitzat en la Raspberry Pi, el seu sistema operatiu o en la programació dels pins GPIO, una funció exclusivament de la Raspberry Pi.

Mentre anava programant, anava construint l'electrònica, i viceversa.

Un dels principals inconvenients que té programar en Raspberry Pi vers en Arduino és la limitació d'entrades. En l'Arduino tens entrades i sortides tant digitals com analògiques. Per entendre això és necessari saber les principals diferències entre una senyal digital i una senyal analògica. Aquesta diferència és molt important si es pretén fer feina amb electrònica.

La **senyal digital** té una forma booleana, el que vol dir que es representa amb encès o apagat: una senyal fàcil de transcriure en binari. En canvi, la senyal analògica té un gradient (finit en aquestes plaques) de possibilitats, normalment del 0 al 255, això indica la intensitat de la senyal que arriba, i igualment és més fàcil controlar el voltatge de sortida amb el mateix mètode.

Què vol dir això i per què va ser un inconvenient a l'hora de programar-ho tot? Doncs això vol dir que la Raspberry Pi només és capaç d'entendre intensitats HIGH o LOW, és a dir, 1 o 0, encès o apagat. Això em va limitar molt perquè no volia recórrer a traductors externs ja que són difícils de controlar, en lloc de fer servir aquests aparells vaig haver de desenvolupar un codi molt més feixuc per tal de poder llegir senyals que no fossin només 1 o 0; com per exemple la intensitat lumínica.

Això ho vaig fer (consultant alguna web<sup>3</sup>) mitjançant variables de temps; mesurant el temps que tardava una senyal de 3.3 volts a retornar-me per complet a través d'un resistor lumínic-dependent i un condensador electrolític de 1 $\mu$ F. D'aquesta manera detectant 1 o 0 vaig ser capaç de definir la intensitat que arribava al LDR.

Després de desenvolupar el codi per a l'LDR, em vaig posar a fer el codi pel motor pas a pas, ja que a la pràctica tot serà un mateix programa.

El **motor pas a pas**, com explica el diagrama d'aquest motor que he incorporat a l'apartat de materials, està format per diversos electroimants que s'activen i desactiven en sincronia

---

<sup>3</sup> Pàgina d'instructables

per tal de generar un moviment d'allò més harmònic i suau. Això fa que la programació d'aquest motor tant precís no sigui gaire senzilla.

A la placa transmutadora del motor pas a pas li he d'assignar una sortida de la Raspberry Pi a cada pin de senyal. Amb aquests pins anomenats al programa he de fer una matriu que engegui i apagui el motor tant ràpid com faci falta, i he de fer que aquesta matriu corri també en sincronia amb tota la resta de programa.

Hauria de quedar quelcom així:

```
Seq = [[1,0,0,1],  
        [1,0,0,0],  
        [1,1,0,0],  
        [0,1,0,0],  
        [0,1,1,0],  
        [0,0,1,0],  
        [0,0,1,1],  
        [0,0,0,1]]
```

Amb aquest codi ja podrem activar el motor pas a pas quan ens convingui.

Després d'haver realitzat aquest codi la resta va ser bastant senzill, excepte per un problema que em pensava que no superaria. El que passava és molt senzill: bàsicament quan fas un programa lineal, sense classes, sense fitxers cooperatius, només pots tenir un bucle infinit. Un bucle infinit és aquell que passi el que passi s'ha d'executar, que no depèn d'una estructura selectiva com els "if" o el "switch" en Java. Això vol dir que jo només podia realitzar un programa infinit per cada Raspberry Pi que estigués operativa, i evidentment, no surt gaire a compte tenir tres Raspberry Pi, una per cada bucle infinit...

Finalment vaig haver de recórrer a un dels fòrums esmentats anteriorment, al de la Raspberry Pi. Allí, vaig explicar el meu problema i van ser molt ràpids en respondre'm dient-me que podria fer servir dues funcions molt interessants que jo desconeixia per complet. La primera funció era compartir les variables mitjançant variables globals públiques i modificables, però

cada una d'aquestes variables pertanyia a una funció i això feia que el programa anés lent i no funcionés del tot bé (a part de tenir moltíssims errors).

L'altra opció era fer servir el **multi-threading**. Caldria definir doncs què és el multi-threading?

Un thread <sup>4</sup>és la unitat de processament més petita que pot dur a terme un sistema operatiu, i per tant un processador. Aquesta unitat de processament molts cops s'usa per a classificar els processadors per tal que els consumidors sàpiguin quins poden dur a terme més operacions alhora (no té res a veure amb la potència).

Per tant, el multi-threading no és més que l'habilitat que té una CPU (o un nucli en el cas d'una CPU multi-nucli) d'executar diverses operacions alhora, sempre en concordança amb el sistema operatiu que és el que posa el límit si no el posa el processador.

A continuació em vaig posar a investigar sobre com podia prendre avantatge fent ús del multi-threading per aconseguir que cada nucli executés un bucle infinit.

Això es pot aconseguir fent ús de l'estructura següent:

```
def BucleInfinit():  
    #codi  
  
thread1 = threading.Thread(target=Bucleinfinit())  
thread1.start()
```

Aquestes senzilles línies de codi fan que el programa torni a funcionar bé distribuint les operacions infinites a realitzar entre tots els nuclis i entre tots els fils.

---

<sup>4</sup> Thread en català s'acostuma a dir "fil". Així doncs un procesador que tingui quatre nuclis i vuit threads tindrà quatre nuclis i vuit fils.



El receptor d'infrarojos (IR) avisa la Raspberry Pi quan detecta una senyal. Quan rep un pols infraroig l'analitza i comprova que la senyal correspongui a una fuga de gas a la cuina. Per la seva banda el codi LDR calcula la intensitat de la llum que li arriba.

Això fa que el programa es pugui a una situació real, però, com que la Raspberry Pi té una potència bastant limitada, encara que el programa funcioni correctament, de vegades va lent. Això és perquè els perifèrics estan consumint més intensitat de la que la Raspberry Pi se suposa que pot donar per a anar bé, ja que la Raspberry Pi està literalment tota la estona fent càlculs i comparacions amb estructures selectives, imprimint a la consola, comprovant que tot funcioni bé...

Un dels factors que considero molt important en un habitatge domòtic és la seguretat. Vaig pensar en una solució molt interessant per tal de fer la casa una mica més segura: es tracta d'un sistema d'alarma que inhabilita la funció automàtica de la porta del garatge per tal que no pugui entrar cap cotxe o persona. Aquest sistema funcionarà de la manera següent: primer es connecta un **receptor d'infrarojos** a la casa, això ens permetrà controlar el sistema d'alarma d'una manera còmode mitjançant un comandament remot (jo he aprofitat el d'un antic equip de música).

Per tal que la Raspberry Pi reconegui els polsos emesos pel comandament he hagut d'accedir al LIRC<sup>5</sup> (que és una de les coses que més problemes m'ha portat). El LIRC és una organització que dona suport al software lliure i també facilita llibreries per controlar ordres infraroges en Linux. Amb l'ajuda del LIRC per Raspberry Pi i després de molts intents vaig poder configurar-ho tot correctament. Als annexos número 4 i 5 hi afegeixo les instruccions que he fet servir així com l'arxiu de configuració lircd que he usat pel meu comandament a distància.

---

<sup>5</sup> LIRC: Linux Infrared Remote Control

Després d'incorporar al codi unes quantes línies que iniciaven el repositori del LIRCd i analitzaven si els polsos rebuts requerien acció de resposta mitjançant unes estructures selectives; el codi estava llest per rebre i interpretar polsos infrarojos. Tot i això hi ha vegades que aquesta part del codi falla (encara no he trobat l'error) el que ocasiona que tot el programa, des de la porta fins els infrarojos, s'aturin. Això és un problema de seguretat i funcionalitat molt greu que no he pogut solucionar.

Igualment em semblava molt interessant tenir una plataforma Android a la qual poder accedir per tal de controlar tot l'habitatge, però, després de considerar-ne les dificultats vaig veure que amb el poc temps que em quedava i tots els errors que havia de solucionar, no podia iniciar aquesta part del projecte. Així que al final no he fet ni una plataforma web ni tan sols una manera d'accedir al control de la casa per bluetooth, enlloc d'això ho he fet per infrarojos com he explicat anteriorment.

El que he fet per aconseguir que l'habitatge sigui una mica més segur és **establir una connexió via correu** de la meua Raspberry Pi al smartphone o ordinador. Això vol dir que en qualsevol moment que jo li indiqui em pot enviar un informe d'estat al meu correu. Fer això fa que la nostra domotització agafi un caràcter més segur, però vulnera el nostre correu obrint moltes portes que es poden aprofitar per entrar-hi sense permís. Així que jo em vaig crear un correu només per a fer això, en cap cas vaig usar el meu correu personal.

Per tal d'aconseguir això vaig haver d'autoritzar al correu l'accés remot mitjançant el protocol SMTP Server. Hi ha molta informació a internet sobre com fer això amb adreces “@gmail.com” així que no entraré gaire en detall.

El que vaig aconseguir amb això és que cada cop que l'alarma es desactiva m'envia un correu informant-me. Aquest sistema que he instal·lat a la cuina és només una demostració perquè el mateix procediment es pot fer servir per rebre avisos sobre els registres del sensor de gas a casa o per fer qualsevol cosa. És molt versàtil i realment no té límits.

Cal tenir en compte que tot el que he desenvolupat són exemples de com es poden fer servir les diferents funcions. Això vol dir que encara que jo només he posat un motor pas a pas per la porta del garatge, es pot fer servir molt fàcilment per apujar i abaixar les persianes segons la intensitat lumínica; encara que jo només he posat un sensor de gas a la cuina, no vol dir que no se'n pugui posar un de fum (hauria de valdre el mateix tipus) al garatge per si et deixes el cotxe amb el motor encès. I de la mateixa manera, tot i que jo només he implementat un sistema d'avís per correu electrònic, aquesta funció es podria extrapolar a qualsevol altra avís o alarma. He de dir que, com que he buscat els sensors més barats, n'he hagut de retornar un per defectuós, però després d'això, aquesta part del projecte ha sigut senzilleta.

Després de finalitzar el codi i les proves del sistema de comunicació de correu electrònic em vaig posar a considerar sobre com podria com a mínim utilitzar una mica als nostres aliats; els smartphones. Els portem cada dia sobre nostre, en tot moment, sempre els tenim carregats i normalment encesos. Estem pendents en tot moment d'ells i mai no passem per alt el que ens notifiquen; així que pot ser una bona idea implantar un sistema de notificacions que ens avisi que la porta s'està obrint o que l'alarma s'ha desactivat...

Aquesta part del projecte (que funciona moltes vegades però no sempre) la he fet d'una manera molt senzilla mitjançant una app que es diu **NotifyMyDevice**. Aquesta aplicació està disponible tant per Android com per iOS com fins i tot per Windows Phone (en tots tres casos de manera gratuïta). Aquesta app et permet establir notificacions automàtiques en sistemes com la Raspberry Pi mitjançant un programa. Jo el programa el vaig implementar en un programa Python, però també es pot implementar en Java, C# o nodejs, segons la seva pròpia pàgina web.

Vaig aconseguir el codi en Python de la seva pàgina web i només el vaig haver d'adaptar a les meves necessitats i a l'estructura de multi-threading que té el meu programa. A l'annex 6 hi afegeixo el programa original aconseguit de la seva pàgina web.

De la mateixa manera, a la seva pàgina web, t'ensenyen a configurar l'aplicació per tal de rebre les notificacions.

## 7. Funcionament dels sistemes

En aquest punt del treball explicaré de manera resumida les funcions que he aconseguit implementar a la maqueta i posaré sobre la taula totes les seves peculiaritats (ja siguin virtuts, errors o problemes).

El sistema domòtic detecta quan hi ha un cotxe esperant per entrar al garatge mitjançant un sensor de llum (LDR). Aquest sistema té diversos inconvenients com ara la dependència que té d'estar sempre il·luminat (per una bombeta de LEDs o quelcom similar) per tal de reduir les influències climàtiques. De la mateixa manera, al no estar relacionat amb el vehicle del consumidor directament, podria fer servir aquest sistema qualsevol vehicle, persona, o fins i tot animal que passés per sobre. Per això hem assumit que la casa que s'està domotitzant està dins d'una finca tancada per tal d'evitar vehicles o persones alienes. Igualment, al activar l'alarma mitjançant el comandament d'infrarojos, la funció de la porta automàtica es desactiva, però la llum que il·lumina el sensor es manté encesa per donar a entendre que la vivenda està ocupada. Per això, es recomana una bombeta de baix consum, i no es necessari que sigui gaire potent ja que el llistó límit d'il·luminació es pot configurar fàcilment canviant una línia del codi. El sistema automàtic t'envia una notificació al mòbil cada cop que s'obre la porta. Aquesta notificació normalment funciona molt bé, però si el mòbil tanca les aplicacions en segon pla de manera automàtica no funcionarà (o hauràs d'anar-la obrint).

El sistema d'alarma controla el caràcter automàtic del garatge, així doncs és important que no falli mai. Malauradament no ho he aconseguit, el codi del receptor d'infrarojos falla de tant en tant i ocasiona una aturada en el codi general. És a dir, si el receptor falla, no funciona res. Aquest sistema d'alarma t'avisa per correu quan l'alarma ha estat desactivada.

És per això que no he pensat en cap moment d'implementar aquest sistema domòtic en una casa només automàtica, tot bon sistema domèstic automatitzat ha d'estar acompanyat d'un sistema manual; cosa que jo no he fet, però pel mateix motiu de sempre, el meu treball és per demostrar el que es pot fer, no per extrapolar-ho a un habitatge real ja que no seria gaire pràctic.

El sensor de gasos funciona molt bé, detecta si hi ha una concentració de gasos explosius suficient a l'atmosfera domèstica com per ser perillosa i llavors fa sonar una alarma sonora. La meua puntualització aquí és la de sempre, l'alarma envia una alerta sonora, però podria no ser suficient si estàs treballant o simplement fora de casa. Trobo necessària una alarma que t'avisi per correu si s'ha detectat gas (a més a més del sistema sonor), però ja he fet servir el correu com avís de desactivació d'alarma, i no seria complicat implementar-ho en l'alarmar de la fuga de gas (encara que compromet una mica el rendiment de la Raspberry Pi, per això no ho he fet).

Per últim, el sensor d'infrarojos passiu o PIR, funciona bastant bé, detecta qualsevol cos calent en un rang de 120°. El problema que té aquesta part del treball és que no he aconseguit automatitzar-ho, per tant és una part analògica i totalment aliena a la Raspberry Pi. Si bé és cert que ho podria haver fet (sense gaires problemes) fent un circuit amb un parell o tres de transistors NPN, no ho vaig fer perquè carregava massa la Raspberry Pi. D'igual manera, a posteriori, me'n vaig adonar que es podria haver fet amb només un transistor i sense carregar gaire la Raspberry Pi, i a més hagués funcionat tot concorde amb el sistema d'alarma. Però el pressupost que tenia ja se m'havia esgotat ja que aquest treball ha sigut bastant car.

Així doncs el sistema PIR (alimentat per una pila de 9V) és independent a la Raspberry Pi i disposa d'un interruptor que l'activa o desactiva.

## 8. Conclusions

Per concloure el treball faré una breu comparació del que van ser les meves idees inicials amb el que ha sigut el resultat final. De la mateixa manera avaluaré si he complert els meus objectius satisfactòriament i respondré les preguntes que em vaig plantejar. Faré la comparació mitjançant una taula que separi les idees de la realitat.

Idees inicials:

- Realitzar un garatge automatitzat que permeti l'entrada del vehicle del propietari
- Detecció de gas a la cuina
- Detecció d'intrusos automàtica amb alertes
- Sensor d'IR per tenir controlada l'alarma.
- Plataforma a la internet o al mòbil per a controlar-ho tot

Resultat final:

- He realitzat un garatge automatitzat que permet l'entrada de qualsevol vehicle
- He realitzat un sistema de detecció de gas a la cuina amb alerta sonora
- He realitzat un sistema de detecció d'intrusos analògic i manual
- He realitzat un sistema d'alarma controlat per un receptor d'infrarojos que de vegades falla.
- He realitzat un sistema d'avís per correu i notificació per tal que el propietari estigui informat de l'estat de la casa en tot moment si ho desitja

Concloent, el meu treball es podria considerar un quasi èxit, ja que he aconseguit complir part de les meves idees, però també es podria considerar un quasi fracàs, doncs he aconseguit poques de les meves idees inicials i a més no són funcionals 100%.

Les preguntes que em vaig plantejar són aquestes:

1. Quan és poc i quan és prou (al domotitzar una casa)

2. Què és el que he de saber per fer la meva pròpia casa domòtica?
3. Quant val?
4. Val la pena?

Respondré les preguntes en ordre:

1. Jo diria que una casa domòtica no té sentit si no li poses un mínim de funcions al sistema automàtic, doncs els processadors consumeixen temps i energia quan estan en ús, i si només vols controlar una porta potser no val la pena. En canvi, si et passes amb les funcions domòtiques, podria deixar de valdre la pena també. Això vol dir que no crec que sigui pas útil domotitzar una cosa que no fas servir cada dia a totes hores o que no és de vital importància, però en canvi, és molt útil en funcions de seguretat com un sensor de gas que et pot salvar la vida.
2. Jo diria que no fa falta conèixer res en especial per tal de realitzar un sistema domòtic. És veritat que serà molt més fàcil si tens nocions de programació i electrònica, però no ho veig crucial donada la gran accessibilitat a projectes de codi obert a la internet (com per exemple al github<sup>6</sup>)
3. El preu d'un sistema domòtic (així com el seu cost real) varia molt depenent de les funcions que hi vulguis implementar i de les dimensions de l'habitatge, així que aquesta pregunta no té una resposta genèrica. De totes maneres, la resposta a “quant ha valgut fer la meva maqueta domòtica?” es pot consultar a l'annex número 2.
4. Aquesta pregunta és una mica complicada. Tot depèn de les funcionalitats que s'hi vulguin implementar, diria que no és eficient, ergo no val la pena, si no té suficients implementacions, de la mateixa manera que si en té masses podria deixar de ser eficient

.

---

<sup>6</sup> Github: página web on els usuaris hi publiquen els codis dels seus projectes de manera gratuïta en forma de codi obert. Les úniques condicions d'ús són que no es facin servir amb ànim de lucre i que s'esmenti sempre l'autor del codi.



Els meus objectius eren: veure si una casa domòtica pot ser verda i còmode, i aprendre una mica de Python i familiaritzar-me amb la Raspberry Pi i el seu entorn.

Segons la meva opinió, considero que una casa domòtica verda i còmode és totalment possible (de fet ja és una realitat).i és possible fer-te-la tu a “casa”. No és complicat, hi ha moltíssims fragments de codi per internet que pots adaptar i ajuntar per fer un programa general sense saber programar gaire. De la mateixa manera no necessites gaires coneixements d’electrònica. Però, tot i això, no ho puc afirmar, doncs els meus objectius no eren d’eficiència (donat el poc temps del que disposava).

Puc afirmar que he après una mica de Python, així com m’he familiaritzat amb la Raspberry Pi i tot el que l’envolta, però no puc afirmar que hagi après a programar en Python. Només he adquirit unes nocions bàsiques.

Jo, sempre que em pregunten si sé programar o no, dic: Jo no sé programar, jo sé fer “*copy/paste*”. I realment és veritat, he vist que s’aprèn moltíssim fent “*copy/paste*”. Entens estructures sintàctiques tant complicades per tu que mai haguessis pensat que entendries, compares aquests coneixements amb els que has adquirit programant en altres llenguatges de programació... Crec, sincerament, que el millor professor per la programació és un mateix. No crec que existeixi una “assignatura” en la que es pugui ser més autodidacta, i això fa de la programació una eina molt interessant. Una eina que et canvia la manera de veure les coses, que canvia la manera en la que veus les altres matèries fins i tot si no tenen res a veure com les assignatures de lletres.

Entrant a terreny més personal, des que he començat a endinsar-me en el món de la programació considero que la programació és fonamental per al desenvolupament mental, i crec que hauria de ser una opció a les escoles, tant primària com secundària. Et desenvolupa, et fa aprendre una manera de pensar que no haguessis après sense la programació i sobre tot, et fa adquirir una posició vers els problemes que et fa recapacitar lentament i pensar bé.

Com a exemple de la meua consideració diria que perquè un LED faci pampallugues no li diries “LED, fes pampallugues!”? Li diries, “LED, estigues encès durant dos segons, estigues apagat durant dos segons i fes-ho infinitament”. Doncs aquest pensament s’adquireix amb la programació i és una eina molt poderosa per enfrontar-se a problemes que se’t puguin presentar, encara que no ho sembli.

Per acabar m’agradaria, tenint en compte el cost total del treball que es pot trobar a l’annex 2, reflexionar una mica sobre si val la pena o no fer aquest procés d’automatització a una casa real. Per una banda, part dels costos han sigut de construcció de la maqueta i aquests costos no s’haurien de tenir en compte, però el cost dels motors i sensors, seria més elevat en un habitatge real, pel simple fet que han de ser més segurs i duradors; i no és el mateix aixecar una persianeta de paper que aixecar una porta de metall. Tot i això, crec que si l’automatització es realitza bé pot valdre la pena, encara que el cost es dispari una mica més sense perdre el procés “DIY” que crec pot ser tant interessant.

Encara que fet de manera casolana es factible i pot valdre la pena, crec, que una empresa que es dediqui a realitzar solucions domòtiques ho farà millor, més segur, precís i amb garanties.

#### Resumint:

Considero els resultats d’aquest projecte acceptables, encara que no estic content del tot. Potser em vaig posar un llistó molt alt que no he pogut assumir amb els pocs coneixements que tinc, recursos econòmics dels que disposava (que m’han limitat molt) i el poc temps que he pogut emprar. Igualment no considero el projecte un fracàs perquè he creat coneixement dins meu, i aquest era l’objectiu més important que buscava assolir; i, realment, en major o menor mesura, l’he assolit.

## Bibliografia i webgrafia

Asociación de Fabricantes de Material Eléctrico

<http://docplayer.es/7875682-Clasificacion-de-los-sistemas-domoticos-y-normalizacion-en-el-area-domotica-beatriz-novel-responsable-de-normalizacion-domotica-afme.html>

(consulta: 02.09.2018)

Blog. Domótica. El futuro de las viviendas

<https://domoticaudem.wordpress.com/> (consulta: 02.09.2018)

Blog. La domòtica en la actualidad

<https://sites.google.com/site/proyectededomotica/1--roduccion/1-1-la-evolucion-de-la-domotica-en-espana> (consulta: 02.09.2018)

Casadomo

<https://www.casadomo.com/biblioteca/estudio-mercado-sector-domotica-inmotica>

CEAC

<https://www.ceac.es/blog/tipos-de-instalaciones-domoticas-para-el-hogar> (consulta: 02.09.2018)

CEDOM. Asociación espanyola de domòtica e inmótica

<http://www.cedom.es/sobre-domotica/tabla-de-niveles-para-evaluacion-de-instalaciones-domoticas> (consulta: 02.09.2018)

Domotica sistemas

[https://domoticasistemas.com/tienda/tutoriales/1\\_sistemas-existentes-tipos-y-estandares.html](https://domoticasistemas.com/tienda/tutoriales/1_sistemas-existentes-tipos-y-estandares.html) (consulta: 02.09.2018)

Domotizados

<https://domotizados.co/origen-palabra-domotica/> (consulta: 10.07.2018)v

Educalingo.

<https://educalingo.com/es/dic-es/domotica> (consulta: 10.07.2018)

Gran Enciclopèdia Catalana.

<https://www.enciclopedia.cat/search/obrae/GEC/dom%C3%B2tica> (consulta: 02.09.2018)

**Hogar.Tec**

<http://hogartec.es/hogartec2/sistemas-domoticos-centralizados-descentralizados-y-distribuidos/> (consulta: 02.09.2018)

**Institut d'estudis catalans. Diccionari de la llengua catalana:**

<https://mdlc.iec.cat/results.asp?txtEntrada=domotica&operEntrada=0> (consulta: 10.07.2018)

**Instructables (LDR)**

<https://www.instructables.com/id/Raspberry-Pi-GPIO-Circuits-Using-an-LDR-Analogue-S/> (consulta: 10.7.2018)

**LIRC**

<http://www.lirc.org/> (consulta: 20.7.2018)

**LIRC instruccions**

<https://gist.github.com/prasanthj/c15a5298eb682bde34961c322c95378b> (consulta: 20.7.2018)

**NotifyMyDevice**

<https://www.notifymydevice.com/> (consulta: 28.7.2018)

**Wikipedia**

<https://ca.wikipedia.org/wiki/Dom%C3%B2tica#Hist%C3%B2ria> (consulta: 11.07.2018)

<https://ca.wikipedia.org/wiki/Concentrador> (consulta: 02.09.2018)

<https://ca.wikipedia.org/wiki/Encaminador> (consulta: 02.09.2018)

[https://ca.wikipedia.org/wiki/Bus\\_S%C3%A8rie\\_Universal](https://ca.wikipedia.org/wiki/Bus_S%C3%A8rie_Universal) (consulta: 02.09.2018)

<https://es.wikipedia.org/wiki/Dom%C3%B3tica> (consulta: 10.07.2018)

[https://ca.wikipedia.org/wiki/Raspberry\\_Pi](https://ca.wikipedia.org/wiki/Raspberry_Pi) (consulta: 12.07.2018)

## Imatges

Figura 1 (Raspberry Pi 3 Model B)

<http://www.steg-electronics.ch/gfxbin00997268big/Raspberry-Pi-3-Model-B.jpg>

Figura 2 (Consola o terminal Raspberry Pi)

<https://www.raspberrypi.org/documentation/usage/terminal/>

Figura 3 (Funció LDR)

<https://piensa3d.com/tutorial-programacion-arduino-ldr-sensor-luz/>

Figura 4 (Esquema motor pas a pas)

<https://blog.ars-electronica.com.ar/2015/04/tutorial-stepper-motor-paso-a-paso.html>

Figura 5 (Sensor de gas)

<https://blog.330ohms.com/2016/07/11/como-funcionan-los-sensores-de-gas/>

Figura 6 (Espectre de la llum visible)

<https://imgcop.com/img/Clasificacion-De-Las-Ondas-55904768/>

Figures 7 i 8 (Sensor Passiu d'infrarojos)

<https://www.elprocus.com/passive-infrared-pir-sensor-with-applications/>

Figura 9 (Elevacions de la maqueta)

Imatge presa amb el meu dispositiu mòbil durant la construcció de la maqueta.

Figura 10 (Perspectiva de la maqueta)

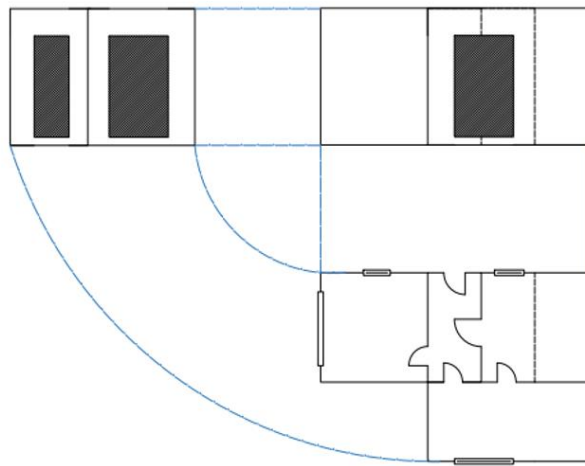
Perspectiva pròpia realitzada per AutoCAD.

Figura 11 (Esquema situació dels sensors i actuadors)

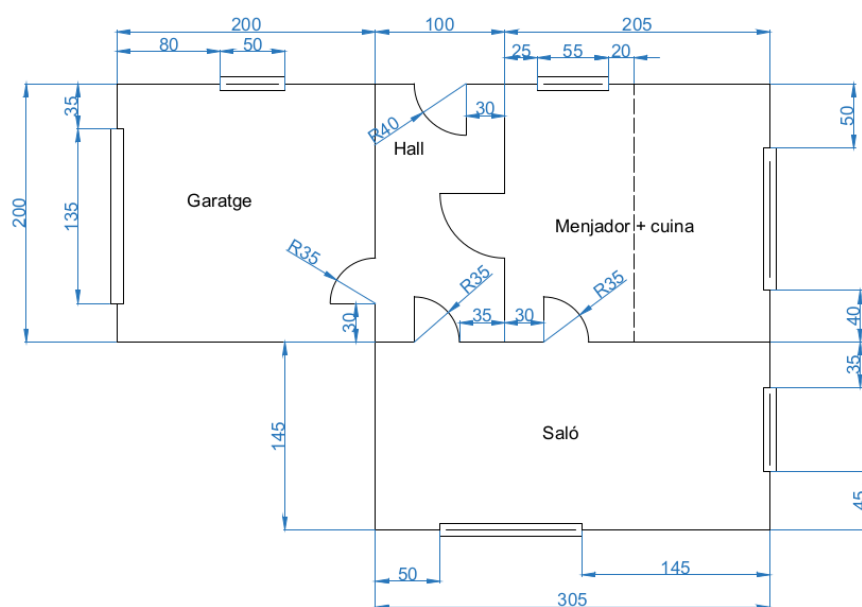
Plànols propis realitzats per AutoCAD i editat amb el GIMP.

## **Annexos**

### Annex 1: Plànols



	Nom	Data	Signatura	IES Antoni de Martí i Franquès	
Dibuixat	Arnau Llop	1/07/2018	Arnau Llop Iglesias		
Comprovat					
Escala 1:20	Treball de Recerca (2018): Casa domòtica per Raspberry Pi				Nº 4



	Nom	Data	Signatura	IES Antoni de Martí i Franquès
Dibuixat	Arnau Llop	1/07/2018	Arnau Llop Iglesias	
Comprovat				
Escala 1:20	Treball de Recerca (2018): Casa domòtica per Raspberry Pi			Nº 1

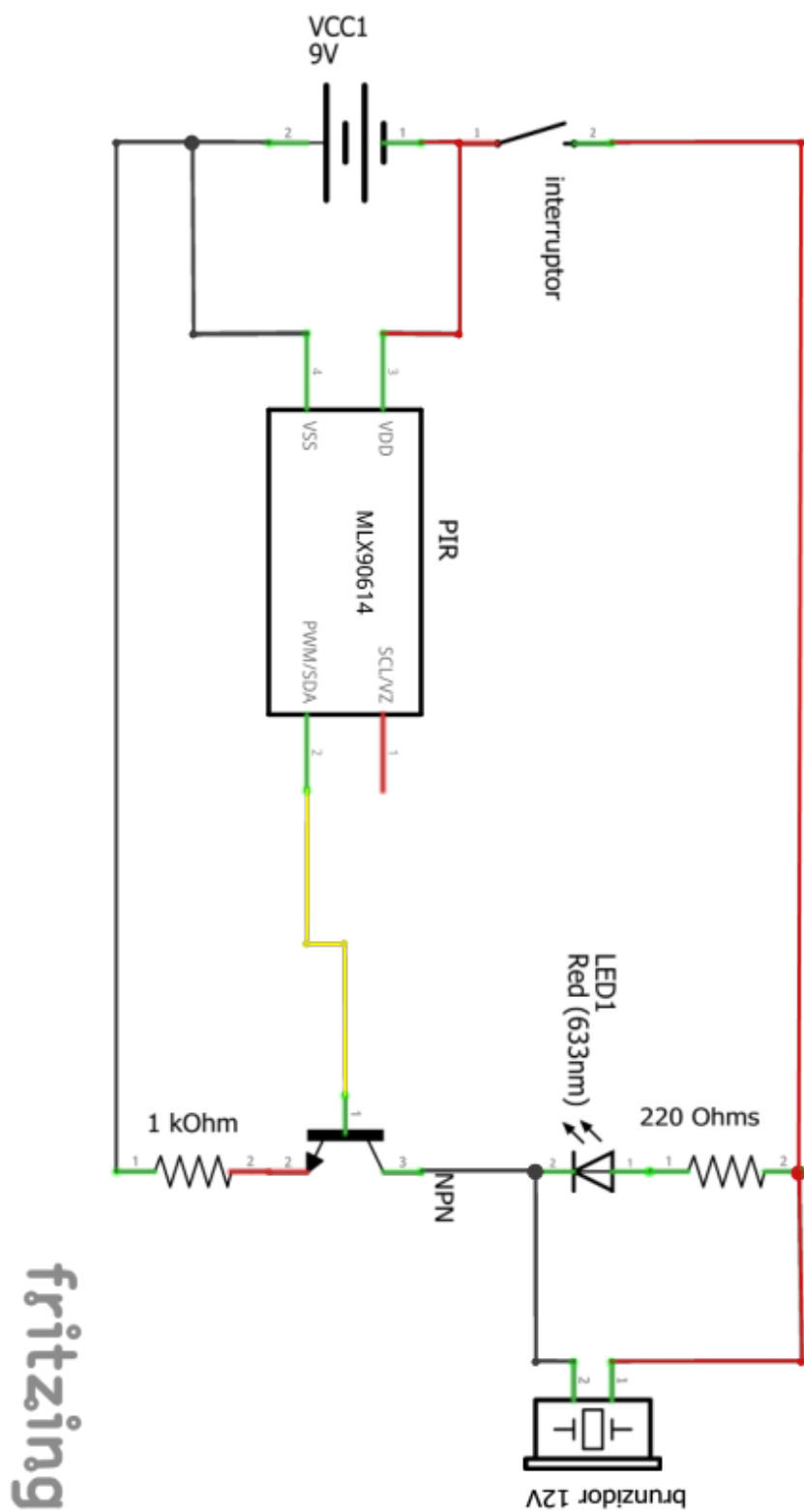


Annex 2: Seguiment de les despeses econòmiques del projecte.

Productes	Comerç on ho he adquirit	Import
Raspberry Pi 3 B pack	Amazon	60€
Dissipadors de Coure	comerç local	5€
Switch HDMI	Amazon	7€
Cables HDMI	Amazon	7€
Adaptador VGA HDMI	Amazon	11€
Cable Ethernet	Amazon	8€
Material de la maqueta	Leroy Merlin	53€
PIR	Comerç local	6€
Receptor IR	Comerç local	2€
Diverses Resistències	Comerç local	1€
Motora pas a pas (x5)	Amazon	14€
Placa PCB soldable	Comerç local	4€
Transistors NPN (x50)	Amazon	2€
Sensor de Gas MQ2	Amazon	10€
Brunzidor 12V	(el tenia per casa)	0€

LEDs	(els tenia per casa)	0€
Jumpers femella-femella	Amazon	4€
Jumpers mascle-masclle + 6 protoboards petites + 2 protoboards mitjanes + 1 protoboard gran	Amazon	9€
Capacitors electrolítics 1µF (x20)	Amazon	5€
Total aproximat		208,00 €

### Annex 3: Circuit del PIR



fritzing

#### Annex 4: Instruccions de la configuració LIRC

Notes to make IR shield (made by LinkSprite) work in Raspberry Pi 3 (bought from Amazon [1]).

The vendor has some documentation [2] but that is not complete and sufficient for Raspbian Stretch.

Following are the changes that I made to make it work.

```
$ sudo apt-get update
```

```
$ sudo apt-get install lirc
```

```
# Add the following lines to /etc/modules file
```

```
lirc_dev
```

```
lirc_rpi gpio_in_pin=18 gpio_out_pin=17
```

```
# Add the following lines to /etc/lirc/hardware.conf file
```

```
LIRCD_ARGS="--uinput --listen"
```

```
LOAD_MODULES=true
```

```
DRIVER="default"
```

```
DEVICE="/dev/lirc0"
```

```
MODULES="lirc_rpi"
```

```
# Update the following line in /boot/config.txt
```

```
dtoverlay=lirc-rpi,gpio_in_pin=18,gpio_out_pin=17
```

# Update the following lines in /etc/lirc/lirc\_options.conf

driver = default

device = /dev/lirc0

\$ sudo /etc/init.d/lircd stop

\$ sudo /etc/init.d/lircd start

# Check status to make lirc is running

\$ sudo /etc/init.d/lircd status

# Reboot before testing

\$ reboot

# To test if lirc driver is working

\$ sudo /etc/init.d/lircd stop

\$ mode2 -d /dev/lirc0

<press a key in remote and you should see multiple lines like below>

pulse 560

space 1706

pulse 535

# to record a custom remote/register a remote device

\$ sudo /etc/init.d/lircd stop

```
$ sudo irrecord -d /dev/lirc0 ~/lircd.conf
```

```
# follow the instruction prompted by the above command carefully
```

```
# at the end ~/lircd.conf file will be generated
```

```
# backup the original lircd.conf
```

```
$ sudo mv /etc/lirc/lircd.conf /etc/lirc/lircd_original.conf
```

```
$ sudo cp ~/lircd.conf /etc/lirc/lircd.conf
```

```
$ sudo /etc/init.d/lircd start
```

```
# you can test if the recorded remote works by
```

```
$ irsend SEND_ONCE <device-name> KEY_POWER
```

```
$ irsend SEND_ONCE <device-name> KEY_VOLUMEUP
```

[1] [https://www.amazon.com/Infrared-Shield-for-Raspberry-Pi/dp/B00K2IICKK/ref=pd\\_sbs\\_328\\_1?\\_encoding=UTF8&psc=1&refRID=1QPY33VFCGETBJ17K8QE](https://www.amazon.com/Infrared-Shield-for-Raspberry-Pi/dp/B00K2IICKK/ref=pd_sbs_328_1?_encoding=UTF8&psc=1&refRID=1QPY33VFCGETBJ17K8QE)

[2] <http://learn.linksprite.com/raspberry-pi/shield/infrared-transceiver-on-raspberry-pi-lirc-software-installation-and-configuration/>

## Annex 5: Arxiu de configuració LIRCD pel meu comandament a distància

```
# Please make this file available to others
# by sending it to <lirc@bartelmus.de>
#
# this config file was automatically generated
# using lirc-0.9.0-pre1(emulation) on Mon Sep 7 13:36:35 2015
#
# contributed by Colin Pamplin - <litten@live.co.uk>
#
# brand:                SONY
# model no. of remote control: RM-S100
# devices being controlled by this remote: SONY MHC-2500
# Mini HiFi System comprising: CD Player - CDP-H300, Tuner
# Amplifier and Twin Tape Decks all from early 1990's
# Note: The Higher spec (larger) RM-S300 has exactly the
# same IR codes for the corresponding keys
# Note: The frequency of 40000 works on my setup (RPi2)
# Alternative is 38000 or 38000 + 7% has been reported to work
# IR frequency I believe is 38Khz
#
# Type of device controlled: Compact HIFI System
# Devices controlled:    SONY MHC-2500
#
# Capture device: Raspberry Pi2 coupled to IR via GPIO
#
# Remote layout:
#
# +-----+
# | KEY_CD      KEY_SLEEP      KEY_POWER      |
# |           |               |               |
# | KEY_PLAYCD  KEY_PAUSECD    KEY_STOPCD      |
# |           |               |               |
# | KEY_EJECTCD KEY_PREVIOUS    KEY_NEXT       |
# |-----|
# |           |               |               |
# | KEY_TAPE    TAPE_A_FREV    KEY_FASTFORWARD |
# |           |               |               |
# | KEY_STOP    PLAY_REV       KEY_PLAY        |
# |           |               |               |
# |-----|
# |           |               |               |
# | TAPE_B_STOP TAPE_B_PLAY_REV TAPE_B_PLAY_FWD |
# |           |               |               |
```

```
# |
# | TAPE_B_FREV  TAPE_B_FFWD  TAPE_B_PAUSE  |
# |-----|
# |
# | KEY_TUNER  KEY_LEFTSHIFTSHIFT  TAPE_B_REC  |
# |
# | PHONO  KEY_CHANNELUP  KEY_VOLUMEUP  |
# |
# | AUX/  KEY_CHANNELDOWN  KEY_VOLUMEDOWN  |
# | VIDEO  |
# +-----+
```

begin remote

```
name    RM-S100
bits    12
flags   SPACE_ENC|CONST_LENGTH
eps     30
aeps    100
```

```
header  2400 600
one     1200 600
zero    600 600
gap     45000
min_repeat  2
toggle_bit  0
```

```
frequency 40000
duty_cycle 33
```

begin codes

```
KEY_POWER      0xA81
KEY_SLEEP      0x061
KEY_CD         0xA41
KEY_PLAYCD     0x4D1
KEY_PAUSECD    0x9D1
KEY_STOPCD     0x1D1
KEY_EJECTCD    0x691
KEY_NEXT       0x8D1  # CD next
KEY_PREVIOUS   0x0D1  # CD previous track
KEY_TAPE       0xC41
TAPE_A_FREV    0xCC1  # Default keys maps to tape A
KEY_FASTFORWARD 0x2C1
KEY_STOP       0x1C1
PLAY_REV       0xEC1
```



```
KEY_PLAY          0x4C1
TAPE_B_STOP       0x18E
TAPE_B_PAUSE      0x98E
TAPE_B_FREV       0xD8E
TAPE_B_PLAY_REV   0x04E
TAPE_B_PLAY_FWD   0x58E
TAPE_B_FFWD       0x38E
TAPE_B_REC        0x78E
KEY_TUNER         0x841
KEY_LEFTSHIFT     0xCD6 # SHIFT
PHONO             0x041
AUX/VIDEO         0x441
KEY_CHANNELUP     0x096 # Tuner channel+
KEY_CHANNELDOWN   0x896 # Tuner channel-
KEY_VOLUMEUP      0x481
KEY_VOLUMEDOWN    0xC81
end codes
```

```
end remote
```

## Annex 6: Codi en Python facilitat per NotifyMyDevice

### Python Example

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-
import requests
import json
api_key = 'user_api_key'
notification_title = 'Some title to send'
notification_message = 'Some message to send'
url = "https://www.notifymydevice.com/push"
data = {"ApiKey": api_key, "PushTitle": notification_title, "PushText":
notification_message}
headers = {'Content-Type': 'application/json'}
r = requests.post(url, data=json.dumps(data), headers=headers)
if r.status_code == 200:
    print 'Notification sent!'
else:
    print 'Error while sending notificaton!'
coding UTF-8 is for users that want to send latin character like ŠĐČĆŽšđčćž
```

## Annex 7: Codi final del programa del projecte en Python

```
#!/usr/bin/env python
# coding=utf-8

# sha dimportar les següents llibreries pel
#correcte funcionament de l'estepper

import sys # importa unes system specific parameters and functions
import time
import RPi.GPIO as GPIO
import threading
import socket
import smtplib
import requests
import json
import os

GPIO.setmode(GPIO.BCM) #anomena els pins pel nom GPIO, no pel nombre

SOCKPATH = "/var/run/lirc/lircd"
sock = None

pin_gas = 13
GPIO.setup(pin_gas, GPIO.IN, pull_up_down = GPIO.PUD_DOWN)

StepPins = [17, 19, 27, 22] #array amb la menció dels pins

for pin in StepPins:
    print ("Setup pins")
    GPIO.setup(pin,GPIO.OUT)
    GPIO.output(pin, False)

GPIO.setup(5,GPIO.OUT)

global USUARI
global PASS
global DESTINATARI
global ASSUMPTE
global COS

USUARI = 'provestdrilop@gmail.com'
PASS = '*****'
DESTINATARI = 'provesTDRilop@gmail.com'
ASSUMPTE = 'RPI3 des de la casa domòtica'
COS = "Alarma desactivada, has estat tu?"
```

```
# definir la seqüència descrita al full tècnic del fabricant
```

```
Seq = [[1,0,0,1],  
        [1,0,0,0],  
        [1,1,0,0],  
        [0,1,0,0],  
        [0,1,1,0],  
        [0,0,1,0],  
        [0,0,1,1],  
        [0,0,0,1]]
```

```
StepCount = len(Seq)
```

```
#print ("Introdueix el sentit de gir")
```

```
#StepDir = int(input(""))
```

```
StepDir = 1
```

```
if len(sys.argv)>1:
```

```
    WaitTime = int(sys.argv[1])/float(1000)
```

```
else:
```

```
    WaitTime = 10/float(1000)
```

```
#StepCounter = 0
```

```
#timeout = time.time() + 5 #5 segons de bucle
```

```
def correu_alarma():
```

```
    try:
```

```
        s = smtplib.SMTP('smtp.gmail.com',587)
```

```
        s.ehlo()
```

```
        s.starttls()
```

```
        s.ehlo()
```

```
        s.login(USUARI, PASS)
```

```
        s.sendmail(USUARI, DESTINATARI, COS)
```

```
        s.quit()
```

```
        print 'Mail enviat!'
```

```
    except:
```

```
        print 'Quelcom ha fallat'
```

```
def notificacio_alarma():
```

```
global api_key, notification_title, notification_message, url, data
api_key = 'MSOHFBOD9Q83D8B5J8CWDAMND'
notification_title = "UIII!!!"
notification_message = "Alarma desactivada"
url = "https://www.notifymydevice.com/push"
data = {"ApiKey": api_key, "PushTitle": notification_title
, "PushText": notification_message}

headers = {'Content-Type': 'application/json'}

global r

r = requests.post(url, data=json.dumps(data), headers = headers)

if r.status_code == 200:
    print 'Notification sent'
else:
    print 'Something went wrong'

def pujada():
    timeout = time.time() + 5 #5 segons de bucle
    StepDir = -1
    test = 0
    StepCounter = 0
    trenca = False
    while not trenca:

        print StepCounter
        print Seq[StepCounter]

        for pin in range (0,4):
            xpin = StepPins[pin]
            if Seq[StepCounter][pin]!=0:
                GPIO.output (xpin,True)
            else:
                GPIO.output(xpin,False)

        StepCounter += StepDir

        if StepCounter>=StepCount:
            StepCounter = 0

        if StepCounter<0:
            StepCounter = StepCount + StepDir

        if test == 5 or time.time() > timeout:
```

```

        trenca = True
        test = test - 1
        time.sleep(WaitTime)

```

def baixada():

```

    timeout = time.time() + 5 #5 segons de bucle
    StepDir = 1
    test = 0
    StepCounter = 0
    trenca = False
    while not trenca:

        print StepCounter
        print Seq[StepCounter]

        for pin in range (0,4):
            xpin = StepPins[pin]
            if Seq[StepCounter][pin]!=0:
                GPIO.output (xpin,True)
            else:
                GPIO.output(xpin,False)

        StepCounter += StepDir

        if StepCounter>=StepCount:
            StepCounter = 0

        if StepCounter<0:
            StepCounter = StepCount + StepDir

        if test == 5 or time.time() > timeout:
            trenca = True
            test = test - 1
            time.sleep(WaitTime)

```

def init\_irw():

```

    global sock
    sock = socket.socket(socket.AF_UNIX, socket.SOCK_STREAM)
    print ('Starting up on %s' % SOCKPATH)
    sock.connect(SOCKPATH)

```

init\_irw()

def next\_key():

```

    """Get the next key pressed. Return keyname,updown.

```

```
'''
while True:
    data = sock.recv (128)
    data = data.strip()

    if data:
        break

words = data.split()
return words[2], words[1]

def infiniteloopIR():
    global val
    val = 1
    GPIO.output(5,True)

    while True:
        keyname, updown = next_key()
        print next_key()
        print('%s (%s)' % (keyname, updown))
        time.sleep(1)

        if keyname == 'KEY_POWER' and val != 1:
            val = 1
            GPIO.output(5,True)
            correu_alarma()
            notificacio_alarma()

        elif keyname != 'KEY_POWER' and val != 0:
            val = 0
            GPIO.output(5,False)

def BucleInfinit():
    mpin=26
    tpin=25
    cap=0.000001
    adj=2.130620985
    i=0
    t=0
    while True:

        if val == 1:

            GPIO.setup(mpin, GPIO.OUT)
```

```
GPIO.setup(tpin, GPIO.OUT)
GPIO.output(mpin, False)
GPIO.output(tpin, False)
time.sleep(0.2)
GPIO.setup(mpin, GPIO.IN)
time.sleep(0.2)
GPIO.output(tpin, True)
starttime=time.time()
endtime=time.time()

while (GPIO.input(mpin) == GPIO.LOW):
    endtime=time.time()

measureresistance=endtime-starttime
res=(measureresistance/cap)*adj
i=i+1
t=t+res
if i==10:

    t=t/i
    print(t)
    if t > 4000:
        pujada()
        time.sleep(5)
        baixada()
        time.sleep(5)

    #i=0
    #t=0
    i=0
    t=0

def infinit_gas():
    GPIO.setup(pin_gas, GPIO.IN, pull_up_down = GPIO.PUD_DOWN)

    while True:

        #print GPIO.input(pin_gas)

        if GPIO.input (pin_gas) == 0:
            os.system("omxplayer -b
/home/pi/Desktop/TDR/TDR_proves/OS_sound/so_alarma.wav")

        #else:
            #print("No gas")
```



```
thread1 = threading.Thread(target=infinitemloopIR)  
thread1.start()
```

```
thread2 = threading.Thread(target=BucleInfinit)  
thread2.start()
```

```
thread3 = threading.Thread(target=infinitem_gas)  
thread3.start()
```