



Departamento de (nome)  
Facultade de Informática  
**UNIVERSIDADE DA CORUÑA**

TRABALLO FIN DE GRAO  
GRAO EN ENXEÑERÍA INFORMÁTICA  
MENCIÓN EN ENXEÑARÍA DE COMPUTADORES

# **Dispositivo e aplicación Android para aumentar a seguridade nos desprazamentos en bicicleta**

**Estudiante:** Sergio Rodríguez Gayoso  
**Director/a/es/as:** Carlos Vázquez Regueiro

A Coruña, 24 de agosto de 2019.



*Dedicatoria*



## **Agradecimentos**

  Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales commodo, lectus velit ultrices augue, a dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in sapien. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Duis fringilla tristique neque. Sed interdum libero ut metus. Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit amet ante lobortis sollicitudin. Praesent blandit blandit mauris. Praesent lectus tellus, aliquet aliquam, luctus a, egestas a, turpis. Mauris lacinia lorem sit amet ipsum. Nunc quis urna dictum turpis accumsan semper.



## **Resumo**

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales commodo, lectus velit ultrices augue, a dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in sapien. Lorem ipsum dolor sit amet, consectetur adipiscing elit. Duis fringilla tristique neque. Sed interdum libero ut metus. Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit amet ante lobortis sollicitudin. Praesent blandit blandit mauris. Praesent lectus tellus, aliquet aliquam, luctus a, egestas a, turpis. Mauris lacinia lorem sit amet ipsum. Nunc quis urna dictum turpis accumsan semper.

### **Palabras chave:**

- First itemtext
- Second itemtext
- Last itemtext
- First itemtext
- Second itemtext
- Last itemtext
- First itemtext



# Índice Xeral

---

<b>1</b>	<b>Introdución</b>	<b>1</b>
1.1	Obxectivos . . . . .	2
1.2	Metodoloxía . . . . .	2
1.3	Proposta . . . . .	3
1.4	Traballo relacionado . . . . .	3
1.5	Estrutura da memoria . . . . .	3
<b>2</b>	<b>Analise e planificación</b>	<b>5</b>
2.1	Análise . . . . .	5
2.2	Planificación . . . . .	7
2.2.1	Fase 1: Dispositivo principal . . . . .	7
2.2.2	Fase 2: Aplicación do dispositivo móvil . . . . .	7
2.3	Custos do proxecto . . . . .	8
<b>3</b>	<b>Deseño e implementación do dispositivo</b>	<b>9</b>
3.1	Esquema xeral do dispositivo HW . . . . .	9
3.1.1	Placas de desenvolvemento . . . . .	9
3.1.2	Luces led . . . . .	12
3.1.3	Cámara . . . . .	13
3.1.4	Alimentación e baterías . . . . .	14
3.1.5	Software a utilizar . . . . .	15
3.1.6	Caixa e ancoraxes a bicicleta . . . . .	16
3.2	Leds . . . . .	16
3.2.1	Conexión coa Raspberry Pi . . . . .	16
3.2.2	Software de Control . . . . .	17
3.3	Camara . . . . .	18
3.3.1	Lentes . . . . .	19
3.3.2	Captura de vídeo . . . . .	19

3.4	Transmisión de vídeo . . . . .	20
3.5	Recepción de ordes . . . . .	20
3.6	Conexión co servidor . . . . .	21
3.7	Autoarranque do servidor . . . . .	21
3.8	Alimentación e enerxía . . . . .	22
3.9	Carcasa e anclaxe . . . . .	24
3.10	Custo do dispositivo . . . . .	25
3.11	Custo do dispositivo . . . . .	25
<b>4</b>	<b>Arquitectura e implementación da aplicación</b>	<b>31</b>
4.1	Esquema xeral da aplicación . . . . .	32
4.2	Actividade principal . . . . .	32
4.2.1	Layout . . . . .	33
4.2.2	Ciclo de vida da actividade . . . . .	33
4.2.3	Botones . . . . .	34
4.2.4	Sensores . . . . .	35
4.2.5	Superficie de vídeo . . . . .	35
4.3	Comunicación co dispositivo . . . . .	35
4.3.1	Broadcast . . . . .	35
4.3.2	Conexión . . . . .	36
4.3.3	Transmisión de ordes . . . . .	36
4.4	Recepción de vídeo . . . . .	36
4.5	Anclaxe a bicicleta . . . . .	36
<b>5</b>	<b>Evaluación experimental</b>	<b>39</b>
5.1	Consumo e autonomía . . . . .	39
5.2	Vídeo e lentes . . . . .	41
5.3	Visibilidade . . . . .	42
5.4	Estabilidade e consumo da aplicación . . . . .	42
<b>6</b>	<b>Conclusións</b>	<b>43</b>
6.1	Traballo futuro . . . . .	43
.1	Contido DVD . . . . .	45
.2	Requisitos e instalación . . . . .	45
<b>A</b>	<b>Glosario de acrónimos</b>	<b>47</b>
<b>B</b>	<b>Glosario de termos</b>	<b>49</b>

# Índice de Figuras

---

3.1	Diagrama de conexión dos leds.	18
3.2	Diagrama de funcionamento do circuíto de alimentación.	24
3.3	Diagrama do dispositivo.	26
3.4	Esquema completo do dispositivo.	27
3.5	Imaxes do circuíto por ambas caras.	28
3.6	Deseños do soporte.	28
3.7	Evolución dos deseños da carcasa.	29
3.8	Ultimo desño da carcasa.	29
4.1	Desño 3D do soporte para o dispositivo móvil.	37
5.1	Grafica do consumo en mA.	40



# Índice de Táboas

---

1.1	Lesividade en función da luminosidade da vía. (incluir cita) . . . . .	1
2.1	Custos monetarios do proxecto . . . . .	8
2.2	Tempo empregado no proxecto . . . . .	8



# Capítulo 1

## Introducción

---

Nos últimos anos os medios de transporte alternativos como son o caso de bicicletas, patinetes e similares están aumentando notablemente. As preocupacións medioambientais, os beneficios para a saúde e as vantaxes en termos de custo e eficiencia están a impulsar estes vehículos unipersonais. Porén, a falta de costume dos condutores xunto coas substancials diferencias entre os vehículos e a falta de proteccións en caso de accidente dificultan a convivencia con automóbiles nas mesmas vías e implican un risco engadido para a integridade física dos condutores destes vehículos. Unha das maneiras de reducir estes riscos é aumentar a visibilidade tanto da bicICLETA por parte dos coches coma viceversa co obxectivo de aumentar as distancias e os tempos de reacción dos condutores.

A maioría dos accidentes nos que a vítima e un ciclista implican un turismo na colisión, (referencia dgt 2016) . O estudio (referencia estudio valencia) con datos da Dirección Xeneral de Tráfico mostra que a pesar de que a maioría de accidentes ciclistas prodúcense de día e con boas condicións de visibilidade, momento no que hay maior numero desprazamentos en bicicleta, no caso de accidentes con pouca luminosidade as gravidade das lesión e superior, taboa

Varios estudos mostran que o uso de luces en bicicletas, especialmente as dinámicas permiten que estas sexan divisadas a maiores distancias tanto de día como de noite. A tese The Nighttime Conspicuity Benefits of Static and Dynamic Bicycle Taillights (incluir cita) estuda os beneficios

	Morto	Ferido grave	Ferido leve	Total
Pleno día	1.1%	13.7%	85.1%	100%
Crepúsculo	1.9%	13.0%	85.1%	100%
Iluminación suficiente (noite)	0.7%	9.3%	90.0%	100%
Iluminación insuficiente (noite)	2.8%	17.7%	79.5%	100%
Sen iluminación (noite)	10.3%	27.4%	62.4%	100%

Táboa 1.1: Lesividade en función da luminosidade da vía. (incluir cita)

de diferentes luces traseiras de bicicleta de noite. Conclúe que as luces que se moven co ciclista, como as colocadas nos nocollos, son as que mais visibilidade aportan pero cando o ciclista deixa de pedalear estes beneficios se perden polo que o uso de unha luz fixan cun patrón de palpadeo pode ser a mellor opción en tódalas circunstancias.

Por outra parte a falta de retrovisores na mayoría de bicicletas implica que o ciclista debe xirarse cada vez que quere saber o que está a acontecer tras el facendo que así perda momentaneamente a visión do que ocorre diante e incluso o equilibrio en ciclistas non experimentados.

Para paliar estos problemas exponse unha solución baseada nun ou varios dispositivos dotados de luces e cámara xunto a outro de control e visualización.

## **1.1 Obxectivos**

Os obxectivos principais de este proxecto serán dous: O primeiro de desenvolvemento de dous dispositivos diferenciados, un dotado de cámara e luces que se poderá colocar en diferentes lugares da bicicleta e do que se poderán utilizar unha ou varias unidades o mesmo tempo que disporá de alimentación propia ou compartida. Un segundo dispositivo de interacción co usuario para o manexo do primeiro e a visualización do vídeo capturado. Tamén será necesario o hardware que permita a comunicación dos dispositivos xa sexa por cable ou sen fíos.

O segundo consistirá en desenvolver o software que permita o funcionamento dos dispositivos. O control das luces para indicar posición aumentar a visibilidade ou indicar manobras coma a freada ou xiro. O control do vídeo permitindo activalo e desactivalo cando se deseñe. As posibles automatizacions como o accendido de luces cando haia pouca visibilidade ou a indicación automática da freada. As interfaces de iteración co usuario que permitan o control e a visualización sen distracciós da conducción. A integración e comunicación entre os dispositivos en tempo real e de forma transparente para o usuario.

Se comenzará coa análise das posíbeis soluciones contemplando as diferentes opcións de hardware dispoñibles tendo en conta o custo, o tamaño, as capacidades de funcionamento, as restriccions de compatibilidade, as restriccions no software a utilizar, a dispoñibilidade e a dificultade de uso polo usuario final.

Se realizará a implementación da solución elixida e se someterá a probas nun entorno real para comprobar o cumplimento dos requisitos establecidos.

## **1.2 Metodoloxía**

A forma de traballo consistirá en seguir as directrices dadas polo método da enxeñaría partindo de estudo de traballos relacionados anteriores e continuando co análise, deseño

implementación e avaliación do sistema implementado.

No apartado de deseño, debido a que o proxecto se estrutura en diferentes compoñentes, optarase por unha metodoloxía Top-Down que consiste en, partindo dos requisitos xerais, dividir o sistema en módulos independentes que se implementarán e probaran por separado. A cada un destes módulos se lle aplicara unha metodoloxía iterativa baseada en prototipos engadindo en cada incremento novas funcionalidades a o prototipo previamente desenvolvido, implementado e validado.

### 1.3 Proposta

Neste traballo propoñerase unha solución baseada nun microordenador colocado baixo a sela, a Raspberry Pi Zero, que contará cunha cámara é unhas serie de luces leds conectadas. Para o control e a visualización do vídeo en directo realizarase unha aplicación android que se executara nun teléfono situado no guiador da bicicleta.

### 1.4 Traballo relacionado

Existen no mercado dispositivos con funcións similares pero poucos integra tódalas funcións. É o caso da cámara Fly6 da compañía Cycliq, un combo de cámara traseira más luces que se poden controlar dende o móvil, pero non permite o streaming en directo do vídeo, solo a gravación. Conta con características interesantes como a estabilización da imaxe, resistencia a auga e un tamaño moi compacto ou seu prezo e de 179 euros. Outra opción existente é a cámara Hexagon que se financiou mediante crowdfunding no ano 2017 pero non chegou a producción. Este dispositivo si que contaba con streaming de vídeo en directo, xunto con acendido automático das luces en caso de freada, a aplicación tamén se encarga de gravar a posición gps no itinerario ou compartila en tempo real, conta cun segundo dispositivo con botóns para o control do dispositivo, e de chegar a producirse o seu prezo estaría entre os 100 e 200 euros.

Non te esquezas de incluir bibliografía e de referenciala no texto. É moi importante (non abuses de páxinas web) !!

### 1.5 Estrutura da memoria

Este documento estrutúrase en seis capítulos e un anexo. Neste primeiro capítulo expone a motivación os requisitos e as lineas xerais do proxecto. O capítulo 2 explora as posibles alternativas dispoñibles analizando as vantaxes e desvantaxes de cada unha delas fronte a solución elixida. O capítulo 3 relata o proceso e os detalles da implementación do dispositivo

a construir xunto co seu software. O capítulo 4 expón a implementación da aplicación android que se utilizará para o control e a visualización. O capítulo 5 describe as probas realizadas e a análise dos seus resultados. O sexto e derradeiro capítulo recolle as conclusión obtidas trala realización deste proxecto.

## Capítulo 2

# Analise e planificación

---

Neste capítulo partirse dos requisitos do proxecto para analizar as posibles funcionalidades planificar a implementación e elixir a solución a implantar.

## 2.1 Análise

O obxectivo do proxecto é a creación de dous dispositivos que realizarán as seguintes funcións:

- **Informar da posición da bicicleta mediante luces**

Esta función realizarase no dispositivo principal. Para elo terá que contar con luces leds e capacidade para controlalas, tamén dispor de capacidade de comunicación co segundo dispositivo e dispoñer dunha fonte de alimentación con capacidade suficiente para facer funcionar as luces.

- **Informar das manobras e estado do vehículo mediante luces**

Para realizar esta función os requisitos son os mesmos da función anterior o que engadiremos a necesidade de sensores para detectar cambios no movemento da bicicleta, para sinalizar freadas ou accidentes, e cambios na luz do ambiente para acender as luces cando as condicións lumínicas non sexan favorables.

- **Captura de vídeo do que sucede detrás do vehículo**

Esta función tamén se realizará no dispositivo principal e necesitará os requisitos de comunicación e alimentación enerxética xa citados. Contará cunha cámara para capturar as imaxes e necesitará a capacidade para procesalas e transmitilas en tempo real.

- **Entrada de ordes do usuario para o control de luces e vídeo**

Esta función se executa no segundo dispositivo, para realizala será necesario un mecanismo de entrada, como poden ser botóns, pulsadores ou pantalla táctil. Tamén se necesitará capacidades de procesamento, comunicación e alimentación enerxética.

- **Reproducción do vídeo en tempo real**

Tamén a realizar no dispositivo dous, esta función necesita, a maiores do anteriormente citado unha pantalla na que poder ver o vídeo, e capacidades abondo para reproducilo a tempo real.

A maiores destes requisitos funcionais contase cos seguintes obxectivos:

- **Pequeno tamaño e potabilidade**

Co motivo de poder dispoñer o dispositivo da bicicleta, xa sexa para cargalo ou por motivos de seguridade o deixar a bicicleta aparcada na rúa, priorizarase por un deseño portátil. O ideal e que os dispositivos poidan separarse da bicicleta con facilidade e que o seu tamaño permita gardalos no peto. Este, xunto cos requisitos funcionais, é o principal motivo polo que optarase por utilizar un teléfono móvil como segundo dispositivo, xa que a maioría de persoas dispoñen de un en todo momento, a así evitariase ter que levar un dispositivo a maiores.

- **Independencia entre dispositivos**

Buscarase que os dispositivos poidan conectarse sen fíos para obter unha maior independencia entre os dispositivos e evitar ter que colocar cables na bicicleta. Tamén se estudará a posibilidade de utilizar unha conexión cableada para diminuir o consumo enerxético dos dispositivos, neste caso optarase preferiblemente por unha conexión USB por compatibilidade co teléfono móvil.

- **Batería e alimentación**

Para poder alimentar o dispositivo principal necesitarase dunha batería con capacidade abonda para poder utilizalo polo menos un dia de uso sen ter que recargala. Tamén se estudará a posibilidade de incluír sistemas de aceso e apagado de ser preciso.

- **Sinxeleza e capacidade de actualización**

Pretendese desenvolver un sistema robusto e simple para facilitar o seu mantemento e poder actualizalo de forma sinxela. Especialmente o software ha de ser o mais simple posible para poder permitir incorporar novas funcionalidades no futuro.

## 2.2 Planificación

Segundo a metodoloxía Top-Dow o desenvolvemento do proxecto dividirase en dúas fases correspondentes a os dous dispositivos. Comezarase polo desenvolvemento do dispositivo principal seguido da aplicación no dispositivo móvil é a comunicación entre ambos. Unha vez desenvolvidas as funcións principais iterarase entre as fase para desenrolar a conexión entre os dispositivos.

### 2.2.1 Fase 1: Dispositivo principal

As tarefas a realizar son as seguintes:

- Conexión dos leds e probas de funcionamento.
- Funcións de control dos leds, secuencias e cores.
- Servidor de peticionés de ordes.
- Conexión da cámara e probas de funcionamento.
- Transmisión de vídeo.
- Autoarranque e apagado.
- Alimentación enerxética e batería.
- Deseño e construcción de carcasa e ancoraxes.

### 2.2.2 Fase 2: Aplicación do dispositivo móvil

Realizaranse a seguintes tarefas:

- Introdución de ordes.
- Deseño de interfaces.
- Transmisión de ordes.
- Xestión do estado.
- Probas con sensores.
- Funcións de automatización.
- Recepción de vídeo.
- Ancoraxe do dispositivo a bicicleta.

Compoñente	Cantidad	Custo por unidade	Subtotal
Raspberry Pi Zero W	2	11.00€	22.00€
Raspberry Pi Zero	2	5.00€	5.00€
Pi Camera Module V1	1	5.00€	5.00€
Pi Camera Module V2	2	20.00€	40.00€
Conxunto de lentes	1	2.00€	2.00€
Tira 8 leds w2812b	4	1.00€	4.00€
Anel 8 leds w2812b	2	1.00€	2.00€
Adafruit Powerboost 1000C	1	27.00€	27.00€
Batería Lipo 1600mA	1	7.00€	7.00€
Batería 18650 3400mA	2	2.00€	4.00€
Batería USB 5000mA	1	10.00€	10.00€
Material impresión 3D PLA	<1Kg	15.00€	15.00€
Cables			5.00€
Total			148.00€

Táboa 2.1: Custos monetarios do proxecto

Tarefa	Tempo
Documentación	80h
Desenvolvemento do dispositivo	100h
Desenvolvemento do software do dispositivo	100h
Desenvolvemento da aplicación	200h
Deseño das pezas 3D	70h
Probas e avaliación experimental	32h
Total	512h

Táboa 2.2: Tempo empregado no proxecto

### 2.3 Custos do proxecto

O software utilizado neste proxecto conta con licenzas de software libre polo que os custos restrinxiranse aos recursos humanos e recursos hardware. Nos recursos hardware da táboa 2.1 inclúense as pezas utilizadas finalmente para o dispositivo así como as que se utilizaron para realización de probas.

As estes custos engádense as ferramentas utilizadas incluíndo entre outros ordenador, impresora 3D, multímetro, soldador, dispositivo android e bicicleta.

Na táboa 2.2 móstranse os recursos humanos utilizados medidos en horas de traballo.

## Capítulo 3

# Deseño e implementación do dispositivo

---

Neste capítulo afondarase no desenvolvemento do dispositivo comezando polo aspecto físico, opcións dispoñibles e alternativas, seguido pola construcción do dispositivo e a implementación do software tendo en conta os problemas xurdidos e as solucións aplicadas.

### 3.1 Esquema xeral do dispositivo HW

O "dispositivo HW" debería ter un nome (estar bautizado) para que fose más sinxelo falar del ...

As esixencias principais destes dispositivo son a existencia de conexións para as luces leds, conexións para unha cámara e capacidade de procesamento e comunicación o que engadiremos unha quinta esixencia, xa que o obxectivo do proxecto e conseguir unha solución de hardware e software libre que posibilite que o usuario final poida adquirir os compoñentes e montalos de forma sinxela, é preciso que os compoñentes sexan fáceis de adquirir e traballar con eles.

A continuación analizaranse as diferentes opcións de hardware dispoñibles, as súas características e seus pros e contras para o proxecto proposto.

#### 3.1.1 Placas de desenvolvemento

Faltan referencias !! para que o lector poida consultar toda a información. Aquí só debes poñer a relevante para o proxecto (evita poñer palla)

Sendo o desenvolvemento dunha placa dedicada a este propósito a solución idílica, o custo de este proceso xunto coa dificultade final da construcción, sempre que non se optase por unha producción en serie, son as principais contras desta opción. Poren optarase por elixir unha placa cos requisitos requiridos entre as posibles opcións dispoñibles no mercado. Centrarémonos nos modelos co menor tamaño posible.

As placas contempladas son as seguintes:

- Arduino

Arduino é unha compañía dedicada o deseño e produción de placas de desenvolvemento con software e hardware, o que permite que terceiras compañías produzan as súas placas permitindo prezos finais de produto moi baixos.

Con un ide propio cunha linguaxe de programación baseada en C++, cunha ampla compatibilidade con diverso hardware, a diversidade de opcións e a sua facilidade de uso converte as súas placas nas mais populares do mercado.

Polo seu tamaño e forma as placas Arduino consideradas son as seguintes:

- Arduino Micro e Arduino Nano

O primeiro está baseado no microcontrolador de 8bits ATmega32U4 cunha frecuencia de 16MHz, 32KB de memoria flash e 2.5KB de SRAM. Conta con 20 pins de entradas/saídas dixitais con 7 canles PWM e 12 pins de entradas analóxicas.

O segundo baséase no microcontrolador de 8bits ATmega328 traballando a unha frecuencia de 16MHz, conta con 32KB de memoria flash e 2KB de SRAM. Conta con 22 pins de entradas/saídas dixitais dos cales 6 son PWM tamén conta con 8 pins de entradas analóxicas.

Estas dúas versión o contar con gran cantidade de pins tanto dixitais como analóxicos e un consumo enerxético e moi baixo xunto cun baixo prezo inferior os 5 euros nas versións de fabricantes de terceiros fainos perfectos para o propósito de control de leds, pero a sua baixa potencia computacional dificultaría o procesamento de vídeo. Tampouco conta co hardware necesario para as comunicación sen fios.

- Arduino MKR ZERO e Arduino MKR1000

Ambos baseado no procesador ARM M0+ de 32 bit de baixo consumo cunha frecuencia de funcionamento de 48MHz, 256KB de memoria flash e 32kB de SRAM, contan con 7 entradas analóxicas e 1 saída analólica e 12 pins poden funcionar como PWM. Inclúe conexións SPI, UART e I2C. Tamén inclúen unha conexión para alimentalos directamente cunha batería de 3.7v. A diferencia entre ambos e que o primeiro conta con 22 pins de entrada e saída dixital mentres que o segundo conta con 8 pero inclúe un chip Wi-Fi.

Ambos teñen as capacidades de procesamento necesarias para a xestión das luces, do vídeo e as conexións. O único punto negativo é que as cámaras compatibles a nivel de conexión e librerías con estas placas non dispoñen de moita calidade de vídeo.

Estas placas poden obterse por entre 20 e 50 euros

– Arduino MKR VIDOR 4000

Esta placa de desenvolvemento a parte do procesador ARM M0+ inclúe un chip *FPGA* que permite a sua configuración como diferentes hardware permitido que a placa poda dispoñer de diferentes compoñentes configurables como podería ser múltiples USB ou chips aceleradores de vídeo. Aparte conta con conexión micro HDMI mini PCI Express e un conector de cámara *MIP*I no que se poderían conectar diversas cámara con calidade mais que suficiente para este proxecto.

O seu prezo e superior os 60 euros.

• Raspberry Pi

As Raspberry Pi son unha serie de placas de desenvolvemento cun prezo moi axustado e unha potencia suficiente para para pode executar un sistema operativo completo. Grazas a sua popularidade dispón dun amplio soporte e compatibilidade con diversos software e outras plataformas hardware que a fan perfecta para diversos proxectos, como robótica, *Iot* ou centros multimedia. A versión dispoñible con menor tamaño e a Raspberry Pi Zero

– Raspberry Pi Zero e Raspberry PI Zero W

Esta placa conta cun microprocesador baseado na arquitectura ARM de 32bits que funciona a 1GHz, acompaña de un procesador de vídeo e unha memoria ram de 512MB. No apartado de conexións conta con un micro USB de carga e outro de datos, unha saída de vídeo HDMI e outra analóxica, unha rañura para unha tarxeta micro sd e un conector de cámara CSI. Tamen conta con 20 pins de conexión que a dotan de entradas e saídas dixitais, dúas canles *PWM*, conexiós *SPI* *I2C* e *UART* xunto a conexións de 5v, 3.3v e terra. A versión Zero W tamén dispón dun chip Wi-Fi e Bluetooth. A sua potencia e capacidade de conexión a fan más que capaz de para este proxecto, e o seu prezo, 5 e 10 euros respectivamente, é unha das súas principais vantaxes.

• ESP8266 e ESP32

A principal característica destas placas é que implementan chips Wi-Fi e Wi-Fi más Bluetooth respectivamente, contan cun procesador *RISC* de un ou dous núcleos con velocidades dispoñibles entre os 80MHz e 240MHz e memorias ram de entre 32KiB e 520KiB.

Os seus múltiples portos e interfaces, *SPI*, *I2C*, *UART*, *PWM* entre outros, o seu baixo consumo e a sua compatibilidade co entorno de programación de arduino fainos ideais para pequenos proxectos de IoT, robótica ou domótica. Segundo as súas características poden obterse dende o prezo de un euro.

O igual que pasaba coas placas Arduino os ESP son ideais para a parte do manexo das luces pero non para a xestión do vídeo.

- Outras placas baseadas en procesadores ARM

No mercado existen múltiples placas de desenvolvemento baseadas en procesadores ARM, non obstante o prezo e o soporte da Raspberry Pi faina a mellor opción para a maioría de proxectos mais xenéricos.

- Outras placas baseadas en *FPGA*

Os chip *FPGA* permiten un nivel de personalización hardware moi elevado, pero tamén contan cun alto prezo, e na maioría dos casos cun *toolchain* privativo que é necesario pagar para poder desenvolver en eles. O contrapunto a sua versatilidade é un maior custo de desenvolvemento en comparación con solucións de programación de alto nivel.

Tendo en conta o tamaño, a potencia, as conexións dispoñibles e o prezo, decidiuse optar pola Raspberry Pi Zero e Zero W como a placa encargada do control das luces e do vídeo.

### 3.1.2 Luces led

As dúas principais vantaxes das luces led fronte a outras formas de iluminación son o seu baixo consumo e o seu pequeno tamaño. Estas calidades fainas ideais para un dispositivo portátil coma o que pretendemos construír. Os requisitos principais son poder controlar a intensidade dos leds e dispor de polo menos un color vermello para indicar a posición e a freada, e un color amarelo ou ámbar para os intermitentes.

Coa intención de miniaturizar se optara por utilizar leds *RGB* que permiten xerar diversas combinacións de cores e así poder utilizar os mesmos leds para as diferentes funcións. Xa que a Raspberry Pi non conta con saídas analóxicas, utilizaranse as canles *PWM* que permite enviar sinais moduladas en pulsos. A modulación *PWM* permite acender e apagar os leds múltiples veces a unha alta frecuencia a unha velocidade, tan rápidas que o ollo humano percibe como diferentes intensidades lumínicas en función da anchura dos pulsos. En leds *RGB* compatibles o *PWM* tamén se pode utilizar para codificar a cor elixida, e en series de leds conectados e direccionables se pode elixir que led a iluminar e a súa cor e intensidade individualmente, permitindo así controlar un alto numero de leds cunha soa saída *PWM*.

Existen diferentes tipos de leds *RGB* direccionables no mercado, elixiremos o tipo de led en función do seu tipo de conexión, a dispoñibilidade de librerías de software compatibles e coa limitación de que deberán operar a 5v que a voltaxe constante que necesita a Raspberry Pi para funcionar.

Os tipos de led direccionables analizados son:

- WS2812B e WS2813

Estes leds inclúen un circuíto integrado en cada led que o conectalos en serie permite o control dunha secuencia teoricamente infinita de leds. Cada led conta con tres entradas e tres saídas: voltaxe, terra e datos. A información a pasar os leds se formara cun fluxo de datos a almacenar nun *buffer* en memoria, ocupando a información de cada un 3 bytes, e se pasarán o primeiro led que lerá os primeiros 24 bits coa información da intensidade de cada cor, vermella, verde e azul, e pasará o resto de datos o seguinte led.

Cada led tarda 30 microsegundos en recibir os datos e 50 microsegundos e actualizar a sua cor, o atraso de transmisión entre leds é de 0.5 microsegundos. O consumo máximo de cada led é de 60 mA a 5V.

Os WS2813 engaden unha segunda liña de datos para que se un led deixe de funcionar os seguintes poidan seguir recibindo a información.

- SK6812

Estes leds comparten a maioría de características dos WS2812B coa diferencia de que aumenta a sua taxa de refresco a 1.2KHz con respecto os 400Hz dos WS2812B. Tamén engaden unha cuarta cor branca en cada led.

O a nivel de software son compatibles con WS2812B pero as súas diferenzas non permiten a sua interconexión física.

- APA102 e APA102C

Estes leds contan cunha interface SPI que conta cunha sinal de datos é outra de reloxo. Isto é para solucionar o problema de sincronización que se poden producir cando os led son manexados dende placas con capacidade de multitarefa sen un *kernel* específico para entrada e saída a tempo real. Tamén aumenta sua taxa de refresco ata os 19.2kHz.

Sendo os leds APA102 superiores en características os outros dous, contan coa desvantaxe de que requiren máis cables de conexión. As súas vantaxes a nivel de velocidad e sincronismo son esencias para a xeración de imaxes ou vídeo pero non para simples animacións como as que utilizaremos neste proxecto. Porén optaremos por utilizar os leds WS2812B e WS2813 ou SK6812.

Este tipo de leds están dispoñibles en diferentes combinacións: leds individuais, tiras flexibles, tiras ríxidas, aneis e matrices. Faremos probas con tiras e aneis de diferentes tamaños.

### 3.1.3 Cámara

No caso da cámara plantexanse dúas opcións utilizar unha cámara usb ou unha das cámaras deseñadas para funcionar coa Raspberry Pi que utilizan a sua conexión CSI. Optaremos pola

segunda opción xa que unha cámara usb implica un tamaño demasiado grande para o proxecto e ademais non soen estar indicadas para a iluminación de exteriores.

No mercado existen diversos módulos de cámara para a Raspberry Pi pero a maioría están baseados nos Raspberry Pi Camera Module V1 e Raspberry Pi Camera Module V2 a principal diferencia entre ambos é que o primeiro conta con 5 megapixeles mentres o segundo conta con 8 megapixeles e unha notable mellora na calidade de imaxe.

As principais diferenzas nos módulos dispoñibles no mercado son:

- Tamaño

Existen versións específicas para a Raspberry Pi Zero más pequenas pero solo do Camera Module V1, as versións normais xa contan cun tamaño moi axustado.

- Presenza de filtro infravermello

As cámara soen contar cun filtro de luz infravermella para evitar o *aliasing* que se produce nas cámara xa que as pantallas que utilizamos non están destinadas para emitir infravermellos e os nosos ollos non son capaces de percibilos. As cámaras que non contan con este filtro dan como resultado imaxes mais luminosas e cunha tonalidade violácea. Estas cámara son útiles para entornos exteriores no solpor ou para visión nocturna se se conta con fontes de luz infravermellas.

- Tipo de lente

A uso dunha cámara cunha lente curva permite ampliar o campo de visión da cámara, se a lente é demasiado curva se producirá unha distorsión da imaxe nos bordes.

Xa que estas cámaras son todas compatíbeis a nivel de software probaremos diversos tipos con varios tipos de lente xa sexan incorporados ou engadindo unha lente externa.

#### 3.1.4 Alimentación e baterías

A Raspberry Pi Zero necesita unha fonte de alimentación que provea de 5V constantes. O seu consumo enerxético varía segundo a carga computacional, o uso do Wi-Fi e o uso da cámara podendo ascender a entorno 300mA. Os leds tamén funcionarán a 5V cun consumo máximo de 60mA por led, cando emiten luz branca a máxima intensidade.

Plantexamos dúas solución posibles:

- Bateria externa USB

Utilizar unha batería usb externa permite dispoñer de altas capacidades que prolongarían o tempo de uso pero implican o uso dun dispositivo a maiores. Outra de desvantaxes é que cando se esgota a batería a corrente interrompese de golpe sen que a Raspberry

poida realizar un apagado normal, como consecuencia tras varios apagados podería danarse o sistema de ficheiros se se estaba a escribir nel no momento do apagado.

- Batería, circuíto de alimentación e circuíto de encendido e apagado.

A maioría de baterías usadas en electrónica son baterías de *ions de litio* principalmente debido a sua alta capacidade enerxética e lonxevidade. A *voltaxe nominal* destas baterías é de 3.7V sendo 4.2V a voltaxe coa carga máxime e por debaixo de 3V deixan de proporcionar suficiente intensidade eléctrica para a maioría de aplicacións. Este tipo de baterías son as que atoparemos nos teléfonos móbiles, ordenadores portátiles e incluso en vehículos eléctricos. Poden colocarse en serie cando é necesario unha maior voltaxe ou en paralelo cando o que se necesita é unha maior capacidade.

Para poder utilizar estas baterías é necesario un circuíto de carga, un de protección, e un de conversión de voltaxe. Na maioría dos casos as baterías de consumo utilizan o estándar USB, que funciona a 5V, tanto para cargarse como para proporcionar enerxía. Polo que será necesario un chip de carga que acepte unha toma de 5V e que cargue a batería ata 4.2V. As baterías de litio poden ser perigosas danándose e chegando incluso a estoupar se se descargan demasiado ou se se sobrecargan polo que é necesario un circuíto de protección que evite a sobrecarga e a sobrecarga. Para proporcionar unha saída estable de 5V tamén é necesario un conversor de voltaxe. É habitual atopar o circuítos de carga más protección xuntos no mesmo chip áñada que tamén se atopan circuítos que integran as tres funcionalidades.

### 3.1.5 Software a utilizar

A Raspberry Pi conta cunha ampla gama de sistemas operativos, algúns con propósitos concretos como reproducción de multimedia, servidores locais ou nodos de rede. Tamén conta con versións das distribucións Linux máis popular como Ubuntu, Arch ou Kali entre outros. Raspbian é unha distribución baseada en Debian, é máis antiga e máis optimizada para a Raspberry Pi e a que dispón de máis soporte polo que será a elixida para o proxecto.

Para o control dos leds existen varias librarías dispoñibles para varias linguaxes de programación. As máis utilizadas son a de *Adafruit*, que só está dispoñible en Python, e a de *Jeremy Garff* que será a que utilicemeos, xa que conta con unha documentación detallada e esta dispoñible en varias linguaxes de programación, entre outras C, Python e Java.

Para a captura e transmisión do vídeo contamos con varias alternativas. A libraría *picamera* para Python permite configurar calquera parámetro e o *streaming* do vídeo na rede. Por outra parte o software para captura de vídeo *raspivideo*, escrito en Python, tamén permite moitos parámetros de configuración é a sua saída de vídeo pode enviarse a rede utilizando algún programa para redireccionar o *bitstream* dos datos como pode ser o software *socat*.

A comunicación entre o dispositivo de control e o de luces e captura de vídeo realizarase a través dunha conexión IP. Para manexar as peticións podemos emplegar unha das clases servidor Python, unha libraría de terceiros ou implementar o servidor a nivel de *sockets*.

Por motivos de compatibilidade entre todo o software necesario para o control de leds, vídeo, e conexións, decidiremos integralo todo nunha aplicación Python que se encargará das tres tarefas.

### 3.1.6 Caixa e ancoraxes a bicicleta

incluir esquemas e figuras para ilustrar conceptos

O lugar a colocar o dispositivo de iluminación e captura será na barra da sela da bicicleta, esta posición é a ideal tanto para capturar o vídeo como para que as luces sexan vistas polo tráfico que circula detrás do vehículo.

A Raspberry Pi conta cunha caixa oficial na que se pode instalar xunto coa cámara V2. Esta é a que utilizaremos no caso de alimentala cunha batería externa. Para suxeitar a placa á barra deseñaremos un soporte e o imprimiremos cunha impresora 3D. Para a versión con batería interna deseñaremos unha caixa protectora que albergue tódolos compoñentes e que se poida suxeitar á barra.

Os prototipos imprimiranse en *PLA* un material biodegradable e de fácil impresión, no é o mellor material para resistir a auga ou a humidade pero é ideal para o prototipado. Poderá utilizarse *ABS* para imprimir unha versión final, un material moito mais resistente as condicións atmosféricas.

## 3.2 Leds

Crearanse dous prototipos con dúas configuracións de leds diferentes, ambas contarán cun anel de 8 leds *RGB* direccionables WS2812B, que conta cun tamaño perfecto para colocar arredor da cámara, a segunda opción contara ademais con dúas tiras de 8 leds cada unha que se utilizaran como indicadores de xiro para aumentar a visibilidade.

### 3.2.1 Conexión coa Raspberry Pi

Estes leds contan con catro puntos de conexión entrada de voltaxe, terra, entrada de datos é saída de datos.

A voltaxe necesaria para alimentalos e de 5V, aínda que na maioría dos caso o fabricante indica un soporte a voltaxes de entrada de entre 4V e 7V. A Raspberry Pi conta con pins de saída a 5V conectada directamente a entrada, sen contar coa limitación dun fusible como noutras versións da placa, polo que pode alimentar os leds directamente pero xa que cada led pode chegar a consumir ata 60mA e facer pasar polo *power rail* unha corrente excesiva podería

provocar danos ou unha aumento da temperatura da placa implicando menor velocidade e maior consumo. Tendo en conta que o fabricante non recomenda que a placa consuma mais de 1A será conveniente alimentar os leds directamente dende a fonte de alimentación, especialmente na versión na que utilizaremos 24 leds.

Para a conexión de datos terase que usar unha das saídas da placa conectadas a un das dúas canles *PWM* da que dispón. Estas saídas lóxicas contan cunha voltaxe de 3.3V, pero as tiras leds requieren que a voltaxe na entrada de datos, para ser interpretada como un valor lóxico HIGH, sexa polo menos un 70% da voltaxe de alimentación, neste caso 3.5V. Para solucionalo proponese dúas opcións: Reducir a voltaxe de entrada dos leds, o que implicaría unha menor luminosidade, ou aumentar a voltaxe do valor lóxico, optarse por esta solución utilizando un conversor lóxico de nivel. Na practica comprobaremos que os leds utilizados seguen interpretando como valor lóxico positivo os 3.3V polo que a utilización ou non do conversor de nivel a valoraremos máis adiante en función do espacio dispoñible.

Os pins da placa dispoñibles con conexión *PWM* serán o GPIO18 e GPIO12 para a canle PWM0 e GPIO13 para a canle PWM1. A libraría que utilizaremos tamén permite controlar o led mediante a conexión SPI e a PCM, utilizaremos a *PWM* por que é a única que nos permite controlar dúas tiras led independentes simultaneamente, coa contraindicación de que o utilizar o *PWM* a Raspberry non poderá xerar audio analóxico, algo que non necesitarase neste proxecto.

Utilizaremos o GPIO18 para controlar o anel led e o GPIO13 no caso que utilicemos os intermitentes que irán conectados un o outro como se indica na figura 3.1.

### 3.2.2 Software de Control

Para manexar os leds utilizaremos a libraría *rpi\_ws281x* de Jeremy Garff na sua versión para Python. O seu funcionamento é moi simple, primeiro teremos que configurar os parámetros da tira led, coma o numero de leds, o pin a que esta conectado, o tipo de tira ou a canle *PWM* entre outros. No programa principal deberemos inicializar os leds con estes parámetros e executala coa función *begin*. Cada vez que queiramos que os les cambien os seu estado chamaremos a función *show*.

Escribiranse funcións encargadas dos padróns de iluminación. Estes padróns serán os seguintes:

- **Luz vermella fixa**, é a encargada de indicar a posición da bicicleta.
- **Luz vermella intermitente**, a sua función é a mima que a anterior, pero o padrón de palpadeo aumentara a visibilidade. Crearanse distintos padróns combinando distintas frecuencias en intensidades lumínicas.

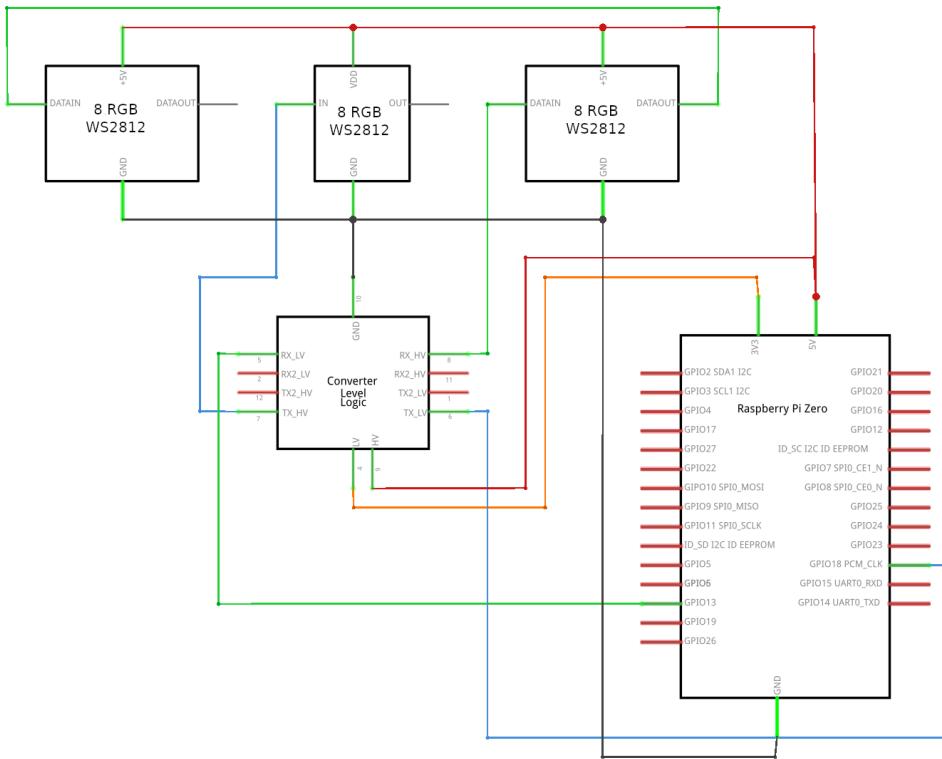


Figura 3.1: Diagrama de conexión dos leds.

- **Luz vermella incremental**, é a encargada de indicar a freada, a sua intensidade aumentará ata o valor máximo para emular as luces de freada dos coches.
- **Luz amarela de xiro a esquerda ou dereita**, indica o xiro iluminando progresivamente os leds do anel dende os situados no centro ata os do extremo esquerdo ou derecho, de dispoñer das tiras extra de leds de xiro estas iluminaranse a continuación. Unha vez iluminados tódolos leds estes se apagarán e o padrón repetirase de novo.

Tamén se escribirá unha función para controlar a intensidade dos leds en función dun valor numérico recibido, 0 será mínimo e 100 a máxima intensidade.

### 3.3 Camara

A cámara a utilizar é a Raspberry Pi Camera, probarase a versión 1 e a versión 2, ambas conectarase a Rasberry Pi Zero co mesmo cable, o conector da placa é delicado polo que deberase conectar con coidado. Para habilitala executarase o comando *raspiconfig* na terminal e no apartado de interfaces activarase a opción cámara.

A versión 1 da cámara conta cunha resolución de 5 megapixeles mentres que a versión 2 conta con 8. **As seguintes imaxes ilustran a diferenza de calidade de imaxe entre as dúas versión.**

### 3.3.1 Lentes

Para poder capturar o completo da estrada a cámara necesitará de algúns tipos de lente que permita un maior campo de visión. Existen versións da cámara que xa inclúen unha lente, pero tamén podemos atopar lentes externas como as destinadas para os dispositivos móbiles, que contan co tamaño necesario para a cámara da Raspberry. **Nas seguintes fotos ilustraemos as diferencias según o tipo de lente.**

### 3.3.2 Captura de vídeo

*Raspivideo* é o software que utilizaremos para capturar as imaxes, o programa executase dende terminal proporcionándolle diferentes parámetros. No noso caso os parámetros a utilizar serán:

- **-t** Tempo de captura de vídeo, no noso caso será 0 indicando que a captura será continua.
- **-w e -h** Son os parámetros de anchura e altura de píxeles, probaremos diferentes resolucións para conseguir a máxima calidade posible sempre que o tamaño da imaxe non repercuta na latencia de transmisión.
- **-fps** *Frames per second*, é o numero de imaxes a capturar cada segundo, variaremos con este valor para minimizar a latencia.
- **-b** *Bitrate*, o numero de bits por segundo, buscaremos o valor máis alto posible sen que produza retardos na transmisión.
- **-n** Con este parámetro deshabilitaremos a previsualización do vídeo.
- **-pf** Parámetro para elixir o perfil do codificador de vídeo H264, as opcións dispoñibles son, *baseline*, *main* and *high*. Utilizaremos a opción *baseline* xa que é a que menor custo computacional ten.
- **-o** Con este parámetro indicamos a saída de vídeo, como por exemplo a un arquivo, no noso caso utilizaremos a saída estándar que indicaremos con “-”, e que redirecionaremos más tarde.

## 3.4 Transmisión de vídeo

Para transmitir o vídeo ao dispositivo móvil a través da rede preséntanse varias posibilidades. Para comparalas transmitiremos vídeo dende a Raspberry Pi cunha mesma resolución, 720p, e recibiremos e reproduciremos o nun pc mediante *vlc*.

- A primeira opción a analizar é o software de vídeo *vlc*, unha completa ferramenta de reproducción que tamén permite a transmisión e a recepción de vídeo na rede mediante diferentes protocolos. Faremos unha proba utilizando o *vlc* na Raspberry Pi para transmitir o vídeo da cámara e recibilo nun pc con *vlc*. Como resultado obtemos unha transmisión cunha latencia superior a un segundo, que imposibilita o seu uso para controlar o tráfico en tempo real.
- A segunda opción que probaremos consistirá en capturar o vídeo coa ferramenta de captura de vídeo da Raspberry Pi, *raspivid*, e redireccionar a sua saída a rede utilizando *netcat* unha utilidade para transmitir e recibir na rede mediante *tcp* ou *udp*. Transmitiremos mediante *udp* para conseguir unha menor latencia a custo de perder algúns fotograma. A recepción de vídeo a realizaremos nun pc mediante *vlc* coma no caso anterior. A latencia obtida neste caso é mellor que no anterior.
- Buscando reducir ainda máis a latencia probaremos a utilizar o software *socat*, que funciona de forma similar a *netcat* e conta tamén con moitas opcións de configuración. O procedemento será igual que no caso anterior, faremos a captura con *raspivid* e redireccionaremos o vídeo a un porto nunha dirección *ip* mediante *udp* neste caso utilizando *socat*. Como resultado obtemos unha latencia ainda menor que con *netcat* polo que utilizaremos este software para a transmisión de vídeo.

## 3.5 Recepción de ordes

Para recibir os comandos enviados dende o dispositivo móvil e executalos mediante Python implementaremos un servidor tamén Python para poder integrar a recepción e a execución de ordes no mesmo programa.

A primeira opción será utilizar peticionés *http* e manexalas mediante a clase de Python *BaseHTTPRequestHandler*, o problema é que esta clase bloquea o programa mentres se executan as ordes correspondentes a petición recibida. Para solucionar este problema poderíase implementar un servidor *multithread* ou utilizar unha libraría para Python que implemente o servidor *multithread* de forma transparente. Plantexase utilizar as librarias *multithread Tornado*, *Twisted* ou *lighttpd* pero debido os recursos limitados da Raspberry Pi Zero o uso dunha destas librarias podería implicar maiores latencias e consumo enerxético.

Finalmente optase por manexar a conexión directamente mediante *sockets* non bloqueantes é así poder utilizar un solo *thread*. Para elo introduciremos as peticións de conexión nunha lista, unha vez aceptada introducirase nunha segunda lista as mensaxes recibidas e se executara a orde correspondente para cada mensaxe.

As mensaxes serán as seguintes:

- **r** para o xiro a dereita.
- **l** para o xiro a esquerda.
- **n** para a luz de noite.
- **b** para a luz de freo.
- **k** para o padrón de palpadeo.
- **o** para apagar as luces.
- **v** para iniciar a captura e transmisión de vídeo.
- **vs** para deter a captura e a transmisión de vídeo.
- **c** para comprobar que a conexión segue aberta.
- **valor numérico** para establecer a intensidade das luces.

### 3.6 Conexión co servidor

Os dispositivos conectaránse mediante unha rede local, xa sexa a través dun cable usb ou dunha rede wi-fi, en ámbolos dous casos o dispositivo Android será o encargado de aloxar a rede xa sexa compartindo por usb ou creando un punto de acceso wi-fi.

Para permitir que o dispositivo móvil se conecte o servidor sen ter que coñecer a dirección *ip* de este procederáse da seguinte maneira. Dende o servidor crearase un *thread* no que se abrirá un *socket* encargado de enviar unha mensaxe a dirección de *broadcast* para que poda ser recibido por tódolos dispositivos da rede.

### 3.7 Autoarranque do servidor

O servidor deberá arrancar automaticamente o acender o dispositivo, e arrancar de novo se por algún motivo detense a súa execución. Para elo crearemos un servizo en *systemd* que se encargará de arrancar o programa e reinicialo se é necesario. A estrutura do servizo e a seguinte, indica a localización do programa Python a executar, a orde de reiniciar sempre, os

*logs* a utilizar para rexistrar as execucións e os fallos, e o usuario e grupo que o executará. Situaremos o ficheiro do servizo na ruta `/etc/systemd/system/` e unha vez ali o habilitaremos coa orde `sudo systemctl enable bikeview.service` agora o servizo executarase cando se arranque o dispositivo e reiniciarase se se para.

## 3.8 Alimentación e enerxía

O consumo de amperios da Raspberry Pi Zero sen carga de traballo é duns 120 mA, gravando vídeo a 1080p o consumo é de 230mA, gravarase vídeo a 720p polo que o consumo será algo menor pero engadirase o consumo do chip wifi funcionando. Os leds ws2812 teñen un consumo máximo de 60 mA cada un 20 mA como máximo por cada un dos tres leds *RGB* a máxima intensidade, o noso máximo consumo realizarse coa luz vermella acesa de forma continua, xa que no resto de modos os padróns de palpadeo reducen o consumo. Contamos con 24 destes leds polo que o consumo máximo será de 20mA por 24 leds, un total de 480mA que sumados o consumo da Raspberry Pi nos da un consumo máximo teórico de 710mA na versión sen leds intermitentes o consumo seria de 160mA máis 230mA, en total 390mA.

- Unha primeira versión más sinxela contará solo cunha batería usb para alimentar a Raspberry. Os requisitos de esta batería serán a amperaxe e a capacidade.

Partirase do valor do consumo máximo aproximado de 400mA para calcular o tempo de funcionamento. Con esta amperaxe aos 5V que funcionan a Raspberry e os leds a potencia utilizada sería de 2W. Neste suposto unha batería de 5000mAh cunha voltaxe nominal de 3.7V pode proporcionar 18.5Wh polo que duraría ata 9 horas e 15 minutos, no caso dunha batería de 1000mAh o tempo mínimo teórico de funcionamento sería de algo menos de dúas horas. Comprobaremos se estes supostos se cumplen facendo medicións do tempo de funcionamento.

- Realizaremos unha segunda versión más avanzada que apagará o dispositivo cando a batería baixe de certo límite de voltaxe para evitar que o dispositivo se desconecte e contará tamén cun pulsador para poder acendela e apagala.

Para isto utilizaremos o chip de carga Adafruit Powerboost 1000 que conta cunhas características moi interesantes a maiores da protección de sobrecarga conta cunha led e un pin que se activaran cando a voltaxe da batería baixe dos 3.2v, unha voltaxe operacional de 5.2v para evitar perdidas de voltaxe en cables e conectores, un pin habilitador que permite conectar ou desconectar a batería, e proporciona 1 amperio de intensidade sen baixar a voltaxe dos 5v. Non conta con protección de sobrecarga polo que as baterías que utilicemos deben incluír un circuito de protección, este é o caso da maioría de baterías, de utilizar unha sin protección, como unha cela 18650, deberemos engadir

o circuíto de protección ou asegurarnos de implementar o apagado por voltaxe baixa correctamente.

A realización o circuíto basearase no guía *lipopi* de Daniel Bull que utiliza o Adafruit Powerboost, nas súas dúas versións a de 500 mA e 1 A, para programar o apagado automático da Raspberry Pi cando a batería baixe de 3.2v e un pulsador para o encendido, tamén conta con dúas versións máis unha que tamén permite o apagado, e outra que monitoriza a voltaxe da batería. Realizarase a versión con pulsador para encendido e apagado.

O funcionamento é o seguinte, o premer o pulsador conéctase o positivo da batería co pin habilitador, acendendo o adafruit powerboost e por conseguinte acendendo a Raspberry Pi. O acender a Raspberry Pi un pin conectado o pin habilitador acenderase para seguir mantendo un valor positivo. Para evitar que o voltaxe no pin habilitador caia no tempo entre que pulsamos o pulsador e a Raspberry Pi arranca e encárgase de manter o valor positivo, situaremos un circuíto RC formado por un condensador cunha resistencia en paralelo entre o pulsador e o pin habilitador. O condensador cargarase cando o pulsador cerre o circuíto e descargarase a continuación mantendo a voltaxe o tempo suficiente para que a Raspberry arranque e acenda o pin. Utilizarase un condensador de  $100\mu F$  xunto cunha resistencia  $100k\Omega$  que proporcionan un tempo suficiente de 10 segundos. O pin da Raspberry que utilizaremos para este propósito pode ser o 14 correspondente a conexión *uart*, que se acenderá coa Raspberry e se desconectará cando se apague, engadiremos unha resistencia de  $10k\Omega$  para protexer este pin. Tamén poderíase utilizar calquera outro pin de propósito xeral indicando no arquivo de configuración *config.txt* na partición *boot* da Raspberry, que o pin arranque cun valor positivo e cun valor negativo cando o dispositivo se apague, no caso de utilizar o pin *GPIO 5* as ordes serían as seguintes: *gpio = 5 = op, dh* para que o pin arranque con valor positivo, *dtoverlay = gpio-poweroff, gpiopin = 5, active\_low = "y"* para deixar o pin apagado cando se apaga a Raspberry.

Para o apagado utilizarase un segundo pin conectado o pulsador, cando este se pulse, estando a Rasberry acesa, se conectará a voltaxe da batería, cando este valor positivo chegue o pin un script Python encargarase de apagar o dispositivo. Engadiranse un divisor de voltaxe para reducir a voltaxe da batería xa que cando está completamente cargada a sua voltaxe de 4.2v e superior o máximo valor lóxico tolerado pola Raspberry Pi de 3.3V. Utilizaremos unha resistencia de  $33k\Omega$  conectada entre o pin e a batería e unha resistencia de  $100k\Omega$  entre o pin e terra. Para evitar que o pin de encendido dispare o apagado situarase un diodo entre o pin de encendido e apagado evitando que a voltaxe circule nesa dirección.

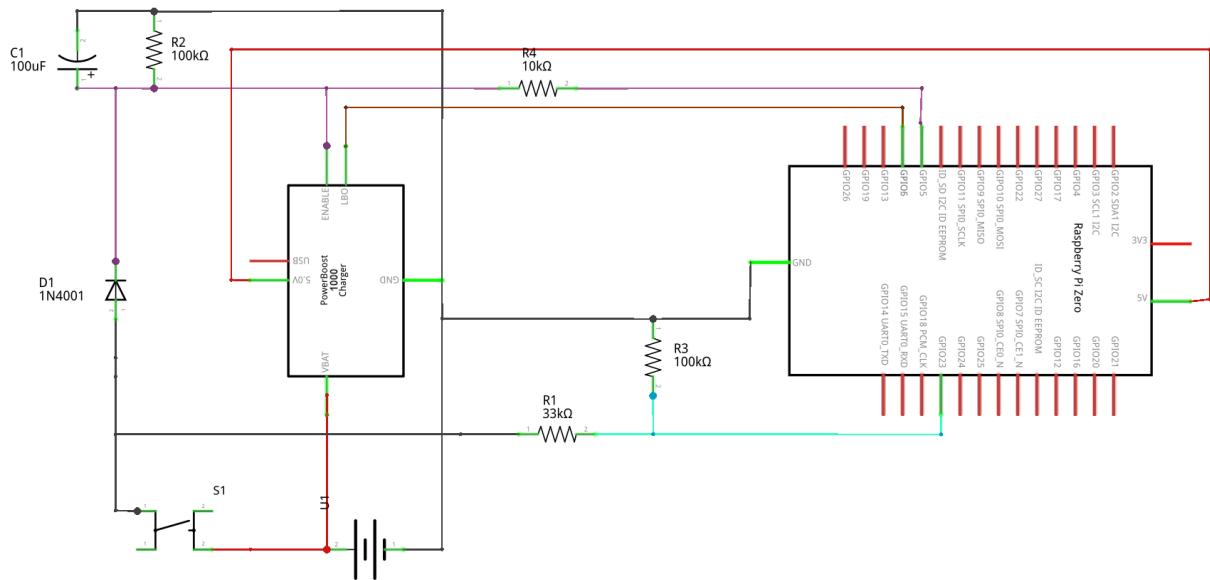


Figura 3.2: Diagrama de funcionamento do circuíto de alimentación.

Finalmente conectarase o pin indicador de batería baixa a outro pin de entrada da Raspberry que mediante o script Python apagará o dispositivo. O diagrama de funcionamento mostrase na figura 3.2.

Na figura 3.3 podemos ver o diagrama final do dispositivo e na figura 3.4 o esquema completo integrando os leds o circuíto de carga e a cámara.

Para arrancar automaticamente o script Python instalarase un novo servizo en *systemd* e de igual maneira que co servidor executarase no arranque e cada vez que se pare.

Na implementación física colocaremos o chip de carga xunto cos compoñentes electrónicos nunha placa cun conector para poder conectalo directamente os conectores da Raspberry Pi Zero. Tamén colocarase nesta placa o conversor lóxico de nivel como se ve na figura

Na elección da batería teremos en conta a maiores da sua capacidade o seu tamaño e forma, sendo o ideal que sexa similar o da Raspberry para poder integrala no dispositivo con facilidade. Por eso elixiremos unha batería de 1600mA e 5.92Wh con circuíto de protección e unhas dimensíons de 9 x 34 x 50mm que para este suposto, xa que engadindo os leds extra calcúlase un consumo máximo de 3.55W, debería proporcionar un tempo mínimo de funcionamento de 2 horas aproximadamente.

## 3.9 Carcasa e anclaxe

Para protexer o dispositivo e suxeitalo baixo a sela da bicicleta plantexaranse dúas opcións.

- A primeira realizarase para a versión do proxecto alimentada cunha batería usb. Consistirá en utilizar a carcasa oficial da Raspberry Pi Zero que inclúe un oco para a cámara e espazo para as conexión na parte de atrás, a carcasa so permite a uso de cámaras sen lentes polo que incorporarase unha lente externa.

Para suxeitar a carcasa a bicicleta deseñaremos un soporte en 3d co software Blender. Partiremos das medición da carcasa e deseñarase un soporte que suxeite a carcasa firmemente e permita atala a barra da sela mediante unha correa.

Unha vez deseñado e tras comprobar que o deseño é imprimible o exportarase no formato *STL* que abriremos cun software encargado de dividir o deseño en capas e traducilo a ordes de desprazamento interpretables pola impresora, como resultado obterase un arquivo *GCODE* que pasaremos a impresora. O prototipo imprimirse o prototipo en 3d probarase e aplicaranse correccións no modelo. Para este deseño realizáronse dúas iteracións móstranse na figura 3.6.

- A segunda versión terá que albergar a Raspberry Pi Zero xunto coa cámara, o chip de carga e alimentación, o conversor lóxico de voltaxe e a batería. Utilizaremos tamén neste caso o software de edición 3d Blender para deseñar os prototipos.

O deseño contara co anel led situado no exterior o redor da lente da cámara, no interior colocaranse tódolos compoñentes electrónicas e a batería. Na parte superior contará cun oco para o conector micro usb de carga e acceso a tarxeta micro sd da Raspberry Pi. No exterior contará tamén con dous brazos articulados nos que situaremos as dúas tiras leds para indicar o xiro.

Da mesma forma que no caso anterior o deseño imprimirse modificarase e volverase a imprimir ata que se obteña un resultado aceptable. Na figura 3.7 pódese observar a evolución dos diferentes deseños e na figura 3.8 unha imaxe renderizada do deseño final.

### 3.10 Custo do dispositivo

### 3.11 Custo do dispositivo

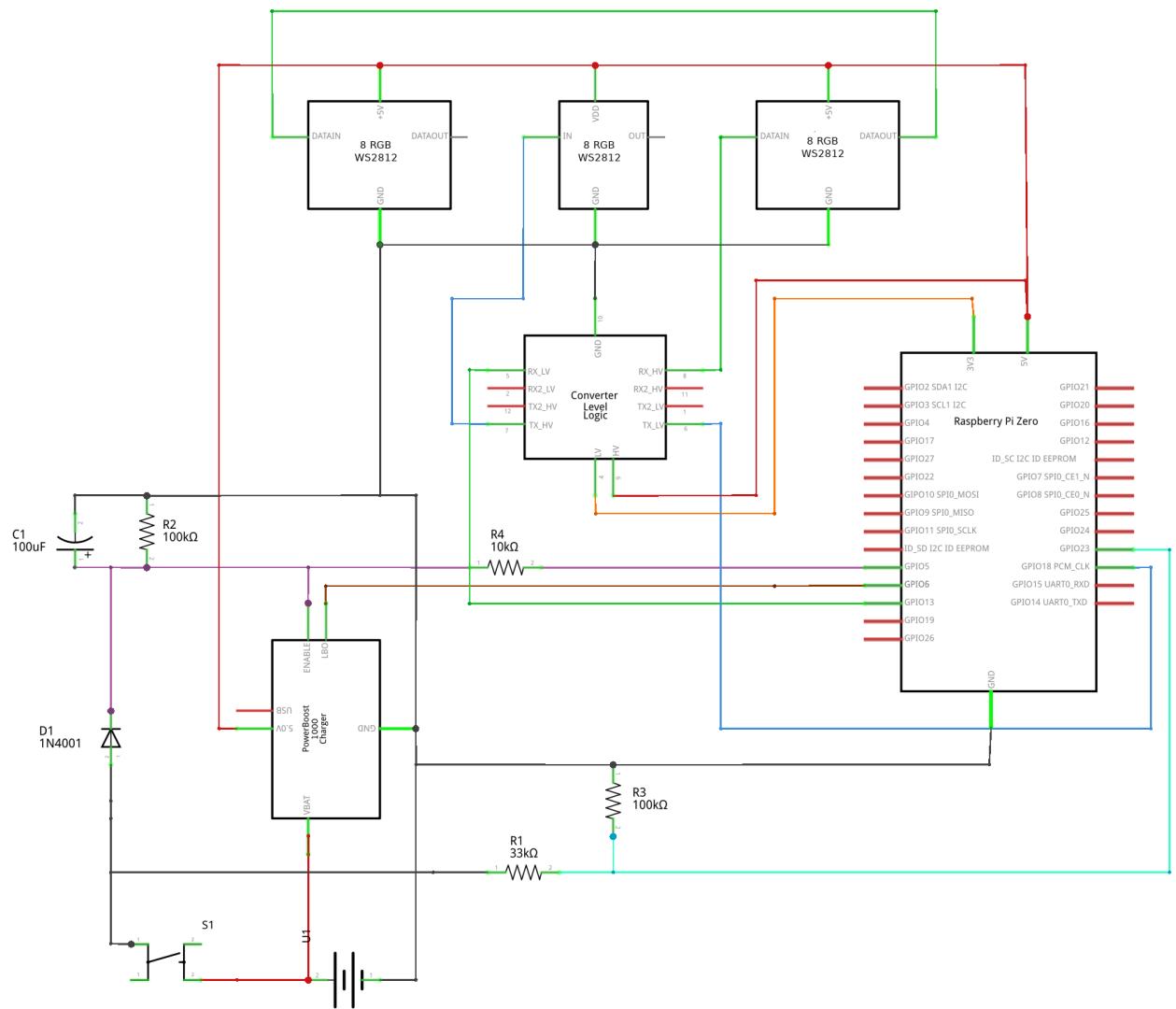


Figura 3.3: Diagrama do dispositivo.

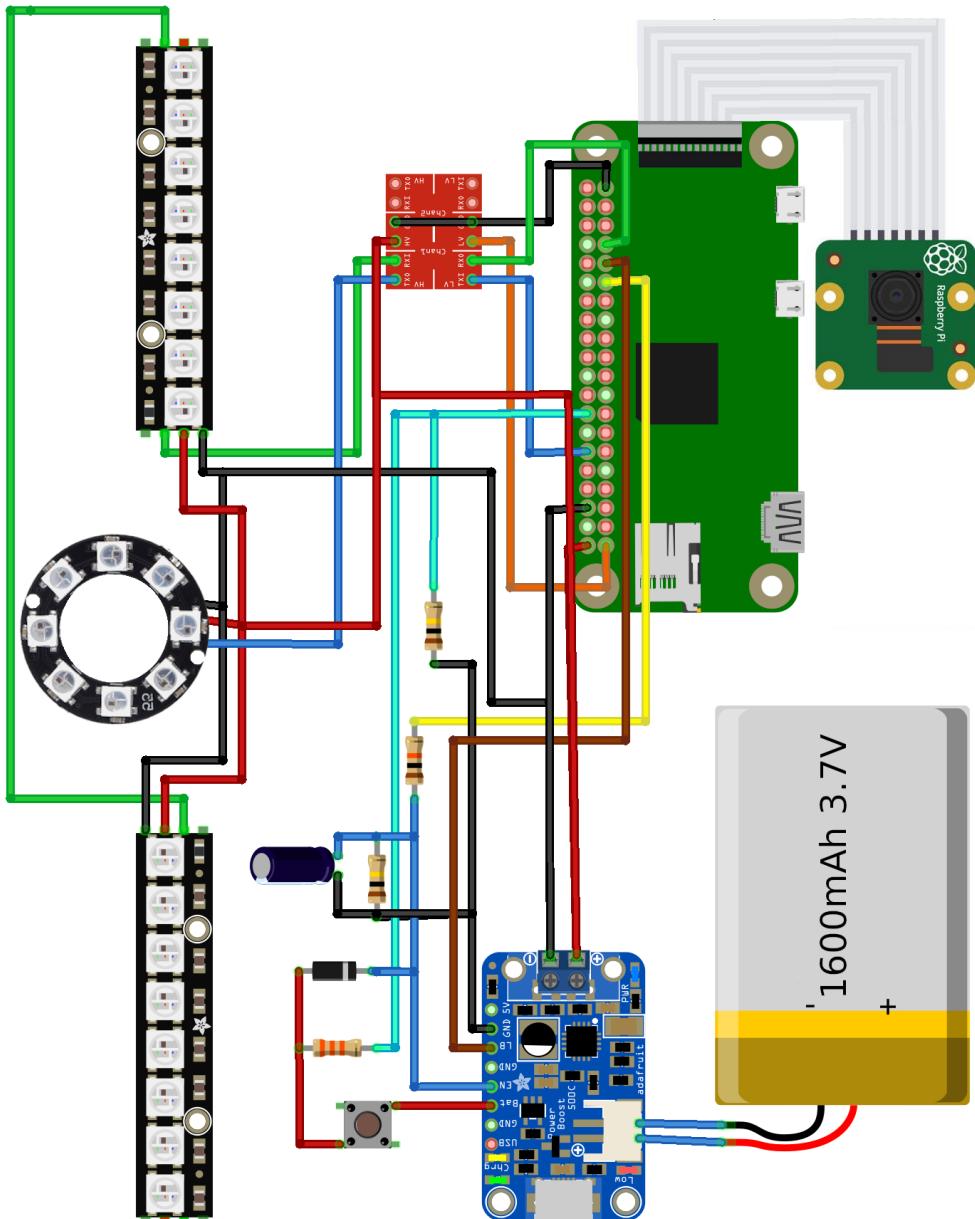


Figura 3.4: Esquema completo do dispositivo.

*3.11. Custo do dispositivo*

---

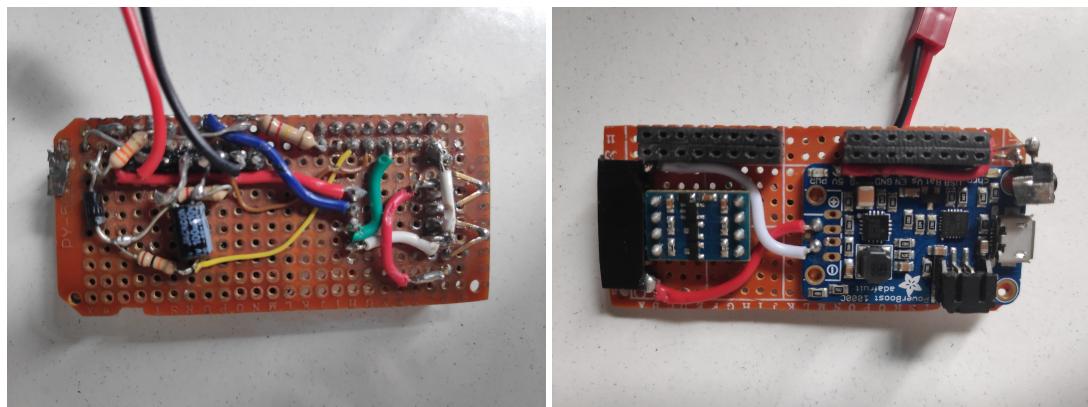


Figura 3.5: Imaxes do circuíto por ambas caras.

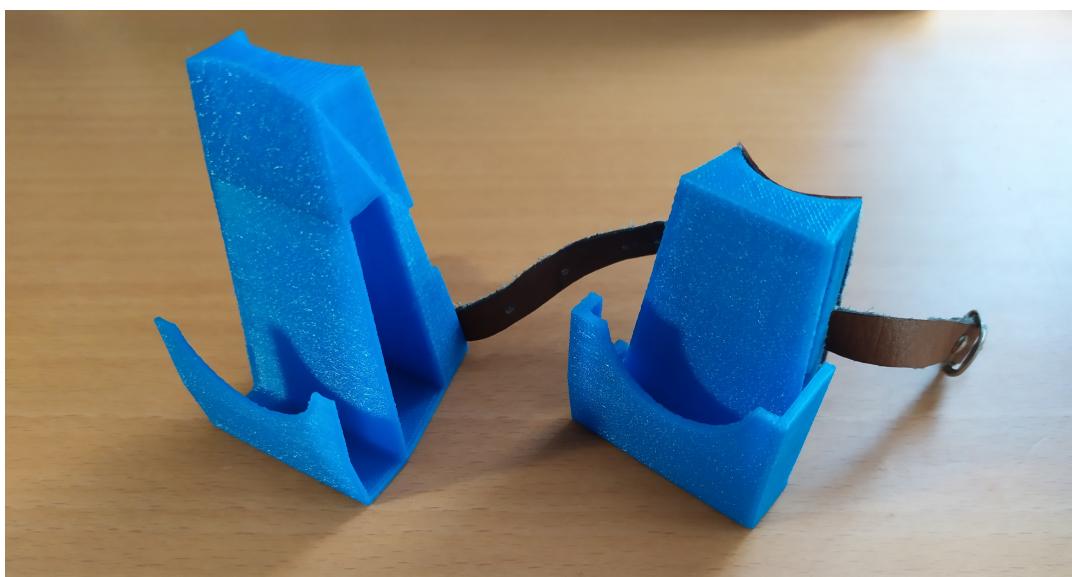


Figura 3.6: Deseños do soporte.



Figura 3.7: Evolución dos deseños da carcasa.

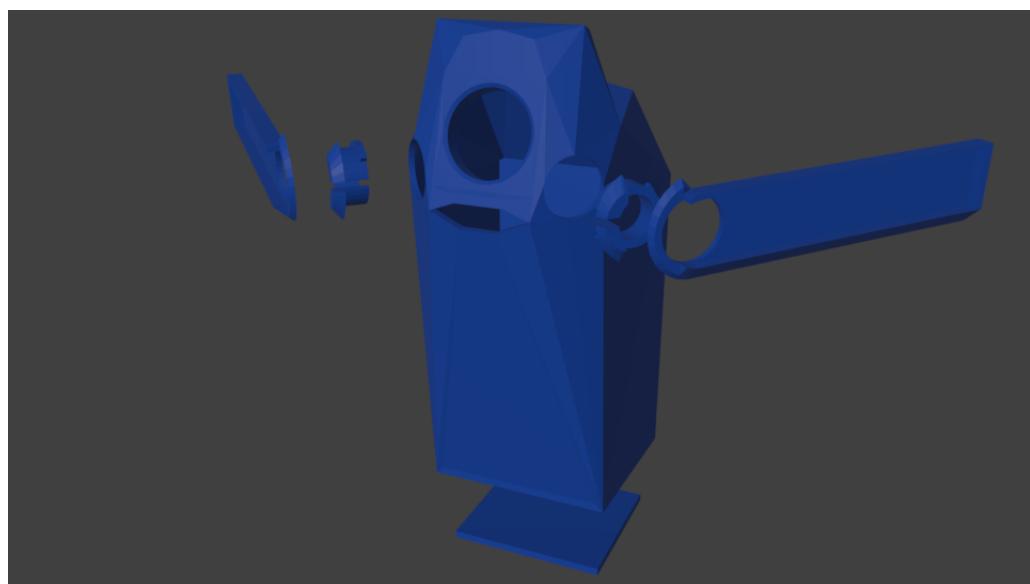


Figura 3.8: Ultimo desño da carcasa.

---

*3.11. Custo do dispositivo*

## Capítulo 4

# Arquitectura e implementación da aplicación

---

Detalles da implementación do software, problemas xurdidos e solucions aplicadas posibles subsección: servidor python- control das luces, control da cámara, comunicación, sockets e broadcasting aplicación Android- layouts e botones, comunicacions ordes e recepcion de vídeo, xestion de varios dispositivos

**Estou algo preocupado, porque agora mesmo o SW é moi sinxelo (literalmente 4 páxinas ...)**

Son moitas as posibilidades de implementación deste dispositivo, os requisitos principais son:

- Dispoñer dunha pantalla para visualizar o vídeo e o estado das luces.
- Contar con algunha interface de entrada de datos como botóns ou pantalla táctil.
- Dispoñer dun hardware para comunicarse co dispositivo principal. Como por exemplo: Wi-Fi, Bluetooth ou USB.
- Contar cunha batería ou fonte de alimentación

Realizar unha implementación física do dispositivo contaría coas vantaxes de poder contar cunha alta personalización dos seus componentes e robustez ao contar cun único dispositivo obxectivo. Sen embargo se nos presenta outra opción moito más atractiva: utilizar un teléfono móvil e crear unha aplicación dende onde poder visualizar o vídeo e controlar as luces, aparte de aforrar a construcción do dispositivo. Reduciremos así o número de componentes que o usuario ten que levar, xa que é habitual dispoñer dun móvil en todo momento.

Desarrollaremos a aplicación para o sistema operativo Android xa que este é o sistema operativo más empregado globalmente e nos permitirá chegar a un maior numero de usuarios.

Para suxeitar un teléfono móvil ao guiador da bicicleta existen múltiples ancoraxes que se adaptan a varios tamaños de teléfono. Tamén existen modelos configurables segundo o tamaño do teléfono que despois se poden imprimir en 3D.

## 4.1 Esquema xeral da aplicación

Para crear esta aplicación Android optarase por utilizar a linguaxe de programación Kotlin e a o entorno de programación Android Studio. Kotlin é unha linguaxe de programación creada por JetBrains cun obxectivo inicial de executarse na maquina virtual de Java, dende 2017 Kotlin é unha linguaxe oficial para desenvolver aplicacións Android.

Algunhas das súas vantaxes fronte a Java son:

- Unha maior expresividade: Podes escribir mais con menos código.
- Maior seguridade: En Kotlin é obligatorio especificar a nulabilidade dos obxectos e esta se comproba en tempo de compilación.
- É funcional: Kotlin é unha linguaxe orientada a obxectos pero inclúe conceptos da programación funcional como as expresións lambda.
- Fai uso de extensión de funcións: Permite estender clases con novas funcionalidades se necesita de ter acceso o código da clase.
- É altamente interoperable: Pódese utilizar librerías e clases de Java no mesmo proxecto.

incluir referencia a libro Kotlin

A aplicación contará con catro compoñentes. O principal será a *MainActivity*, a encargada de iniciar as compoñentes a visualizar na pantalla, executar as ordes e mostrar a información. O segundo será o *layout* onde se definiran as compoñentes a visualizar e a sua posición en pantalla. O terceiro a clase *request* a encargada da conexión co servidor e de transmitirlle as ordes. O cuarto elemento é a clase encargada de recibir o vídeo e descodificalo.

## 4.2 Actividade principal

A actividade principal ou *MainActivity* dunha aplicación Android é a primeira pantalla que aparece cando executamos a aplicación e a encargada de controlar o seu funcionamento e a sua interface de usuario. Neste caso a actividade será a encargada de mostrar os botóns e encargarse do seu funcionamento, amosar o estado da conexión, encargarse da xestión dos sensores e reproducir o vídeo. O mesmo tempo encargarase de instanciar e comunicarse coas clases encargadas da conexión co dispositivo e da recepción de vídeo.

### 4.2.1 Layout

Un *layout* é a definición da estrutura da interface de usuario. Esta actividade contará con dous *layouts* un para a posición vertical e outro para a posición horizontal da pantalla.

#### Layout vertical

Este *layout* contará cunha superficie reservada para o vídeo na metade superior da pantalla. Na metade inferior situaranse os botóns encargados de executar as ordes. Na parte superior situase a barra de estado que na sua parte dereita contará cun botón para conectar que servirá o tempo de indicador de conexión, a sua esquerda mostrarase unha barra para controlar a intensidade dos leds.

imaxes layout vertical

#### Layout horizontal

Neste *layout* o vídeo mostrarase como o fondo e os botóns sobre el. Os de xiro a esquerda e xiro dereita situados a ámbolos dous lados e o resto incluíndo o de control de intensidade lumínica e o de conexión situaranse na parte de inferior da pantalla.

imaxes layout horizontal

### 4.2.2 Ciclo de vida da actividade

En Android cada actividade conta cun ciclo de vida, pasa por varios estados dende antes de iniciarse ata despois de finalizar. O estado principal dunha actividade é activa, no momento que a actividade está en primeiro plano e interactuando co usuario.

imaxe ciclo actividad

Para xestionar o que sucede no resto de estados utilizanse unhas funcións de *callbacks*, no noso caso realizaremos as seguintes accións en cada fase:

#### onCreate

Este método e o que se chama ao executar a actividade. Nel iniciaremos os componentes a mostrar en pantalla e definiremos o seu comportamento.

Aquí xestionaremos o estado da conexión, iniciando a clase *request* se non esta en funcionamento e enviando mensaxes o dispositivo para comprobar que segue conectado. Tamén xestionaremos o estado da transmisión de vídeo xa que será necesario iniciar a recepción antes de iniciar a transmisión.

### onResume

Este método executase despois de *onCreate* cando a pantalla xa é visible para o usuario.

Rexistralo aquí un *sensor listener* para obter a información do sensor de luz.

### onPause

Este método executase cando a aplicación deixa de estar en primeiro plano, se o usuario a minimiza u outra aplicación executase a actividade pasará a *onStop* se despois de acceder a aplicacións recentes a aplicación volve a primeiro plano pásese o método *onResume*.

No noso caso neste método cancelaremos o rexistro do sensor xa que non se seguirá a utilizar e deteremos a transmisión de vídeo se se esta a executar.

### onDestroy

Neste método a actividade e detida completamente e devénse liberar os recursos.

Aquí enviaremos a orde para deter a conexión co dispositivo.

### 4.2.3 Botones

A actividade é a encargada de iniciar os botóns e manexar o seu funcionamento. Estes botón son os seguintes:

- **Esquerda:** O pulsar este botón enviarase unha orde de acender a luz de xiro a esquerda. Se hai algunha luz acesa se enviará primeiro unha orde para apagalas. O pulsalo por segunda vez se enviara a orde de apagar luz de xiro e no caso de que a luz vermella estivese acesa antes de indicar o xiro esta luz se acenderá de novo. O botón palpebrará en amarelo mentres estea aceso.
- **Dereita:** O seu funcionamento é o mesmo que no botón esquerda.
- **Vermello:** O pulsalo enviarase unha orde para acender a luz de vermella, o pulsador non funcionara se algunha das luces de xiro está acesa.
- **Noite:** O pulsalo activarase o modo noite, no que o sensor lumínico do móvil encargarase de acender a luz vermella cando a luz ambiente baixe de certo umbral.
- **Freio:** Activa o modo de freada no que acenderase a luz vermella progresivamente cando o acelerómetro do dispositivo móbil detecte unha freada.
- **Brillo:** Este botón despregará unha barra cun indicador que poderemos deslizar para elixir a intensidade das luces.

- **Conexión:** Este botón enviará unha orde de conexión, unha vez conectado cambiará a súa apariencia para indicar que existe conexión co dispositivo.

#### 4.2.4 Sensores

Unhas das vantaxes de utilizar un dispositivo móvil é que estes contan con diversos sensores para moitos fins. No noso caso utilizarase o acelerómetro e o xiroscopio para rexistrar cambios na posición e aceleracións no dispositivo e o sensor de luz para medir a intensidade de luz no ambiente. En Android accederemos o sensor instanciando a clase *SensorManager* e definindo unha instancia do sensor da que obteremos os datos mediante a función *on sensor change*.

##### Acelerometro

##### Sensor de Luz

O sensor de luz é un fotorreceptor que xera unha sinal eléctrica dependendo da incidencia de fotóns. Para utilizar o sensor rexistraremos un *sensor listener* no método *onResume* e o método de callback *onSensorChanged* executarase cada vez que se detecte un cambio, neste método definirase o comportamento do sensor: Cando o botón de Noite esta activado acenderase o botón Vermello se o sensor rexistra un valor inferior a 400 lúmines. Este valor correspondece coa intensidade lumínica o comenzo do solpor. incluir referencia

#### 4.2.5 Superficie de vídeo

Iniciarase a superficie de vídeo na fase de creación da aplicación, procederase a reflectir a superficie para facer un efecto de espello para facer mais natural a visualización do vídeo. O video asignarase a superficie iniciando a clase *VideoReceiver* mediante as funcións de *callback* que se executarán cando se produza un cambio na superficie de vídeo.

### 4.3 Comunicación co dispositivo

Crearase unha clase *Request* encargada de tódalas comunicacóns co dispositivo. Esta clase, unha vez instanciada, encargarase de establecer a conexión co dispositivo, reconectar se se perde a conexión e transmitirlle as ordes.

#### 4.3.1 Broadcast

Para coñecer a dirección IP do servidor, a clase *request* conta cunha función que abrirá un socket no porto 5555 para esperar a recepción dun datagrama broadcasteado a tódalas direccións. O recibir o datagrama comprobase que contén a mensaxe "BikeView" se é así a función devolverá a dirección IP emisora do paquete.

### 4.3.2 Conexión

Unha vez obtida a dirección IP a clase *request* utilizará unha función para establecer a conexión que devolverá un socket conectado ao socket remoto do servidor.

### 4.3.3 Transmisión de ordes

Para transmitir as ordes a clase *request* contara cunha función que recibe a mensaxe a transmitir e a envía a través do socket. Espera a recibir resposta do servidor e se é positiva devolve o valor booleano verdadeiro, en caso de non recibila devolve o valor falso.

## 4.4 Recepcción de vídeo

Para este apartado crearase a clase *VideoReceiver* que executarase no seu propio *thread*. O vídeo é codificado no formato H.264 e transmitido por UDP, para recibilo ábrese un *socket* no porto 5000 extraese a mensaxe do datagrama e, para acelerar o proceso, en vez de utilizar a función *MediaExtractor* pásase a unha función de parseo encargada de separar os datos recibidos en *NAL units*, o formato que utiliza H.264 para o transporte. Para elo cando se identifica a cabeceira da seguinte *NAL units* a anterior se envía a outra función encargada de decodificar o vídeo mediante un *MediaCodec*, finalmente o vídeo se mostra sa superficie.

## 4.5 Anclaxe a bicicleta

Existen diversas opcións para suxeitar o móvil o guiador dunha bicicleta, no mercado hai soportes para dispositivos concretos e outros que se adaptan ao tamaño e forma de diferentes modelos. No noso caso utilizaremos unha código *SCAD* que executaremos co software de deseño 3D paramétrico OpenSCAD no que introduciremos as dimensíons do dispositivo e como resultado obteremos un arquivo *STL* co deseño 3D do soporte. Vendo que os soportes impresos non constaban coa resistencia esperada optouse por utilizar un soporte de aluminio nas probas para preservar a integridade do dispositivo móvil.

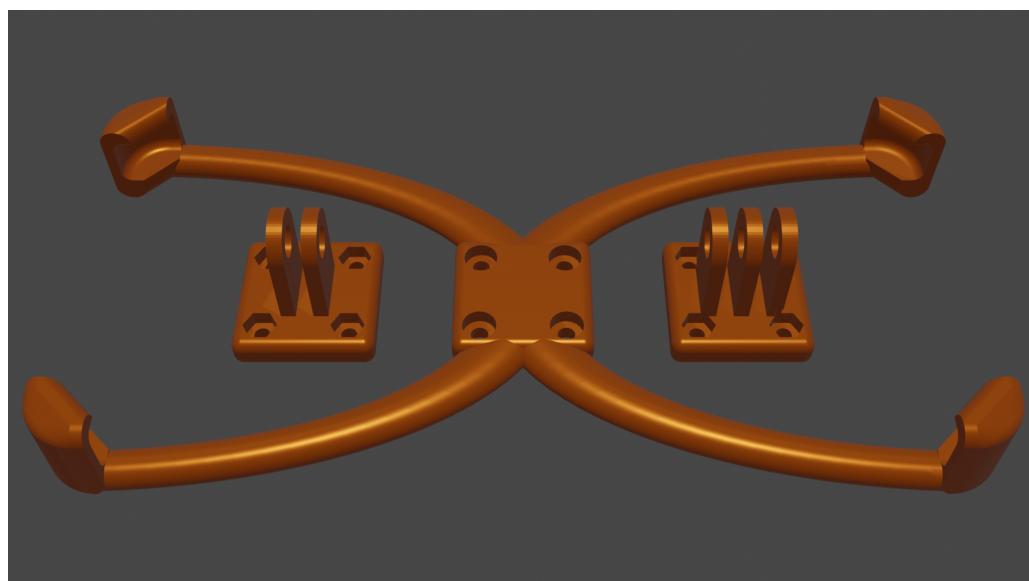


Figura 4.1: Desño 3D do soporte para o dispositivo móbil.



## Capítulo 5

# Evaluación experimental

---

Para comprobar que os resultados obtidos correspóndense co plantexado probaremos o dispositivo e a aplicación. Realizaremos probas en contornos controlados e probas no medio real o que esta dirixido.

### 5.1 Consumo e autonomía

Comenzarase as probas medindo o consumo de amperios do sistema. Para elo colocarase un amperímetro usb entre unha fonte de alimentación e a conexión usb coa que alimentarase o sistema nestas probas. Repetiremos as probas con diferentes fontes de alimentación e distintos cables para descartar fallos e conseguir unha maior consistencia nos resultados. A continuación realizaranse as mesmas probas no dispositivo medindo o consumo alimentándoo coa batería.

Analizaranxe os seguintes supostos para o dispositivo con un anel led e para o que conta a maiores coas dúas tiras led.

- **Sistema en reposo:** O sistema esta aceso pero só se estean a executar as funcións do sistema operativo incluíndo o servidor ssh para o control remoto.
- **Servidor funcionando:** Executamos o servidor.
- **Cliente conectado:** Conectamos o dispositivo móvil o servidor.
- **Vídeo transmitindo:** Transmitimos vídeo en directo o dispositivo móvil.
- **Vídeo parado:** Paramos a transmisión de vídeo.
- **Desconexión do cliente:** Pechamos a aplicación no dispositivo móvil.
- **Luces intermitentes:** Iniciamos as luces intermitentes a máxima intensidade nunha das direccións, consumo varia no proceso, rexistremos o valor máximo.

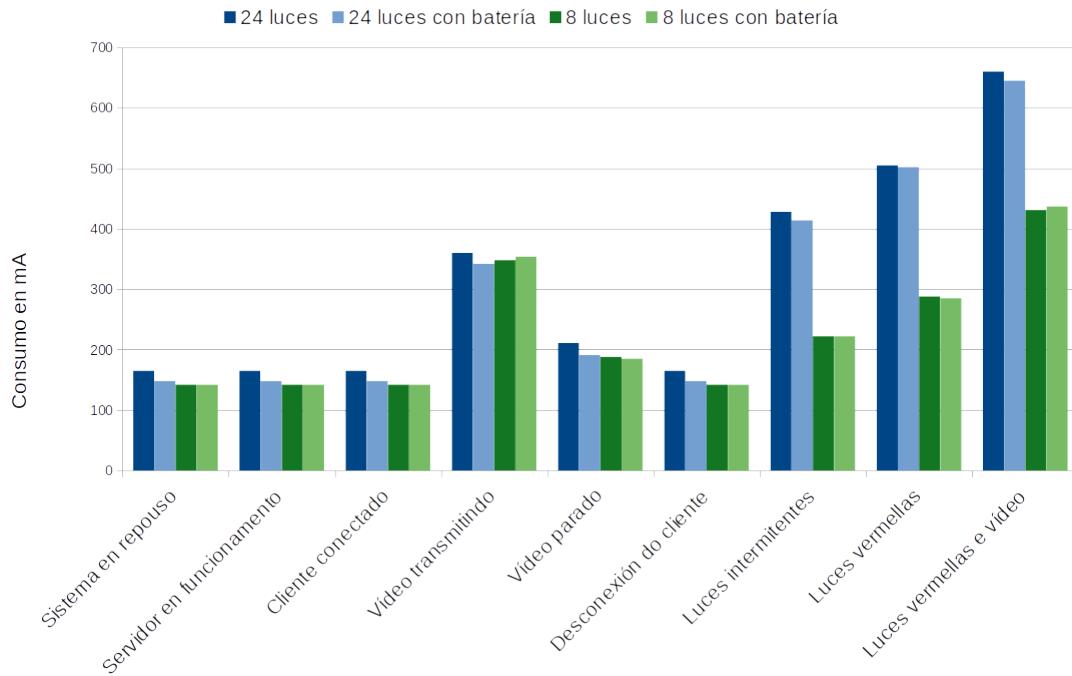


Figura 5.1: Grafica do consumo en mA.

- **Luces vermelhas:** Acendemos as luces vermelhas a intensidade máxima.
- **Luces vermelhas e transmisión de vídeo:** Consumo coas luces vermelhas a máxima intensidade e co vídeo transmitindo en directo.

Como se pode observar na gráfica da figura 5.1 o uso ou non de batería non afecta en gran medida ao consumo de amperios, pero ha de terse en conta que estas probas realizarónse coas batería coa carga completa. Se comparamos a previsión de consumos que fixemos coas medición obtidas

As fontes de alimentación utilizadas poden proporcionar un máximo de 3A e 2.1A respectivamente, tamén utilizáronse dous cables un de maior calidad e outro cunha calidad inferior. En ningún dos casos obtívérонse diferencias apreciables sendo a maior diferencia de 4mA. Na versión con batería necesitouse utilizar cableado a maiores para realizar as probas, o que pode que incrementara lixeiramente o consumo.

A seguinte proba a realizar será a de autonomía do dispositivo, para elo buscarase o consumo máximo acendendo as luces vermella a máxima intensidade o tempo que se transmite vídeo.

Comezaránse as probas co dispositivo dotado de batería interna, cando a voltaxe da batería

esta entorno os 3.7V detectase un problema o sistema apagase, indicando batería baixa. O *script* Python encargado de monitorizar o pin conectado o indicador de voltaxe baixo do Adafruit Powerboost detecta unha caída de voltaxe que confunde co franco de caída esperado cando o pin ponse a 0. O pin de baixo voltaxe do Adafruit Powerboost está conectado a voltaxe da batería cando a caga e superior a 3.2V e conectase a un valor de 0V cando baixa de este límite e debido a que cando batería esta a comezar a descargarse para poder proporcionar a amperaxe adecuada a súa voltaxe baixa a un ritmo rápido que confunde o *script*. O podemos solucionar incluíndo unha segunda comprobación no *script* despois de detectar o franco de caída comprobarase que o valor lido no pin é 0 antes de apagar a Raspberry.

A continuación detectase un segundo problema, despois de min volvese a apagar por batería baixa e unha vez apagado a batería recuperase ata unha voltaxe de 3.5V. Esto debese a que o estar consumindo o máximo de amperios necesita baixar a súa voltaxe para poder seguir entregando estes amperios. Que o sistema se apague neste punto e conveniente para protexer a batería e prolongar a súa vida útil pero reduce o tempo de uso da batería, que poderíamos seguir usando sempre que non utilicemos o consumo máximo do dispositivo. Este problema pódese solucionar de dúas formas: Utilizando un batería de maior capacidade xa que poderá seguir subministrando a amperaxe necesaria a menor voltaxe. Utilizar unha batería cunha constante de descarga maior, isto é a capacidade máxima de amperios que pode subministrar en función da capacidade en amperios hora. A batería utilizada ten unha constante de descarga de 1C isto quere dicir que coa sua caga completa o fabricante garante que proporcione 1.6A funcionando con normalidade. Ámbalas dúas solucións implican na práctica unha batería de maior tamaño.

Procedemos a repetir a proba pero esta vez deshabitando o apagado automático, esperando a que o sistema se apague cando a voltaxe non sexa suficiente para manter a Raspberry acesa ou no peor caso cando o circuito de protección que inclúe a batería a desconecte a 2.5V. De non usar unha batería con circuito de protección poderíamos danar a batería podendo incluso arder ou estoupar durante próximas cargas. Neste caso obtemos minutos de funcionamento.

Comezarase as probas para a versión do dispositivo sen luces intermitentes, faremos a proba cunha batería de 6000mA, obtense , Neste caso observase que cando o dispositivo non pode subministrar suficiente voltaxe para alimentar a Raspberry esta se apaga, e como consecuencia tamén os leds alimentados dende o seu *power rail*, o diminuir o consumo a voltaxe volve a ser suficiente para alimentar a Raspberry polo que esta se reinicia, entrando desta forma nun ciclo de reinicios ata que a voltaxe non e suficiente para arrancar a Raspberry.

## 5.2 Vídeo e lentes

Nesta sección procederase a analizar a calidade do vídeo obtida a latencia

### **5.3 Visibilidade**

### **5.4 Estabilidade e consumo da aplicación**

# **Capítulo 6**

# **Conclusións**

---

**D**ERRADEIRO capítulo da memoria, onde se presentará a situación final do traballo, as leccións aprendidas, a relación coas competencias da titulación en xeral e a mención en particular, posibles liñas futuras,...

## **6.1 Traballo futuro**

Mellorar interfaces da app, subila a tenda de aplicacións de google Engadir mecanismos de control, control por voz, mando no guiador, pulsadores intermitentes desear circuito impreso Detector de caídas Soporte a multiples dispositivos, visualizar diferentes camaras, utilidade en outros ámbitos, camións etc. Mellorar a estructura do dispositivo, facelo resistente a auga e golpes, protexer os leds, Estudar a unclision de difusores de luz e ou lentes nos leds para aumentar a distancia de visualización e oou reducir o Consumo Luces frontais e laterais Aumentar a bateria e o mecanismo de protección desta para aumentar o tempo de uso Apagar o dispositivo dende a aplicación Engadir menú de configuración



# Apéndices

---

## .1 Contido DVD

## .2 Requisitos e instalación

revisar e redactar tutorial descargar ultima version de raspbian lite flashear so en targeta micro sd montar particiones creadas en la targeta navegar a la particion boot de la sd crear un archivo vacio llamado ssh crear un archivo llamdo wpa\_supplicant.conf editar el archivo a diendolasiguiente  
COUNTRYctrl\_interface = DIR = /var/run/wpa\_supplicant GROUP = netdev updateconfig = 1

network= ssid="SSID" psk="PASSWORD" key\_mgmt = WPA-PSK contemplar realizar configuracion //medium.com/@aallan/setting-up-a-headless-raspberry-pi-zero-3ded0b83f274espulsamosl config en la partida de Interfacacing Options activar el modulo de la camara en la partida de advanced options

instalacion de la libreia de jgarff para el control de los leds sudo apt install scons descargamos la libreria con wget [https://github.com/jgarff/rpi\\_ws281x/archive/master.zip](https://github.com/jgarff/rpi_ws281x/archive/master.zip) descomprimimos con unzip ./master.zip entra en la carpeta python y ejecutar sudo python ./setup.py build ejecutar sudo python ./setup.py install



## Apéndice A

# Glosario de acrónimos

---

**ERLANG/OTP** *Erlang Open Telecom Platform.*



## Apéndice B

# Glosario de termos

---

**Bytecode** Código independente da máquina que xeran compiladores de determinadas linguaxes (Java, Erlang,...) e que é executado polo correspondente intérprete.

---