



COMILLAS
UNIVERSIDAD PONTIFICIA

ICAI

GRADO EN INGENIERÍA EN TECNOLOGÍAS INDUSTRIALES

TRABAJO FIN DE GRADO

INTUITIVE SIGNALING SYSTEM FOR MICROMOBILITY

Autor: Contreras Porras, Paloma

Director: Fliflet, Arne

Madrid

July, 2019

AUTHORIZATION FOR DIGITALIZATION, STORAGE AND DISSEMINATION IN THE NETWORK OF END-OF-DEGREE PROJECTS, MASTER PROJECTS, DISSERTATIONS OR BACHILLERATO REPORTS

1. Declaration of authorship and accreditation thereof.

The author Mr. /Ms. _____ PALOMA CONTRERAS PORRAS

HEREBY DECLARES that he/she owns the intellectual property rights regarding the piece of work: INTUITIVE SIGNALING SYSTEM FOR MICROMOBILITY that this is an original piece of work, and that he/she holds the status of author, in the sense granted by the Intellectual Property Law.

2. Subject matter and purpose of this assignment.

With the aim of disseminating the aforementioned piece of work as widely as possible using the University's Institutional Repository the author hereby **GRANTS** Comillas Pontifical University, on a royalty-free and non-exclusive basis, for the maximum legal term and with universal scope, the digitization, archiving, reproduction, distribution and public communication rights, including the right to make it electronically available, as described in the Intellectual Property Law. Transformation rights are assigned solely for the purposes described in a) of the following section.

3. Transfer and access terms

Without prejudice to the ownership of the work, which remains with its author, the transfer of rights covered by this license enables:

- a) Transform it in order to adapt it to any technology suitable for sharing it online, as well as including metadata to register the piece of work and include "watermarks" or any other security or protection system.
- b) Reproduce it in any digital medium in order to be included on an electronic database, including the right to reproduce and store the work on servers for the purposes of guaranteeing its security, maintaining it and preserving its format.
- c) Communicate it, by default, by means of an institutional open archive, which has open and cost-free online access.
- d) Any other way of access (restricted, embargoed, closed) shall be explicitly requested and requires that good cause be demonstrated.
- e) Assign these pieces of work a Creative Commons license by default.
- f) Assign these pieces of work a HANDLE (*persistent URL*). by default.

4. Copyright.

The author, as the owner of a piece of work, has the right to:

- a) Have his/her name clearly identified by the University as the author
- b) Communicate and publish the work in the version assigned and in other subsequent versions using any medium.
- c) Request that the work be withdrawn from the repository for just cause.
- d) Receive reliable communication of any claims third parties may make in relation to the work and, in particular, any claims relating to its intellectual property rights.

5. Duties of the author.

The author agrees to:

- a) Guarantee that the commitment undertaken by means of this official document does not infringe any third party rights, regardless of whether they relate to industrial or intellectual property or any other type.

- b) Guarantee that the content of the work does not infringe any third party honor, privacy or image rights.
- c) Take responsibility for all claims and liability, including compensation for any damages, which may be brought against the University by third parties who believe that their rights and interests have been infringed by the assignment.
- d) Take responsibility in the event that the institutions are found guilty of a rights infringement regarding the work subject to assignment.

6. Institutional Repository purposes and functioning.

The work shall be made available to the users so that they may use it in a fair and respectful way with regards to the copyright, according to the allowances given in the relevant legislation, and for study or research purposes, or any other legal use. With this aim in mind, the University undertakes the following duties and reserves the following powers:

- a) The University shall inform the archive users of the permitted uses; however, it shall not guarantee or take any responsibility for any other subsequent ways the work may be used by users, which are non-compliant with the legislation in force. Any subsequent use, beyond private copying, shall require the source to be cited and authorship to be recognized, as well as the guarantee not to use it to gain commercial profit or carry out any derivative works.
- b) The University shall not review the content of the works, which shall at all times fall under the exclusive responsibility of the author and it shall not be obligated to take part in lawsuits on behalf of the author in the event of any infringement of intellectual property rights deriving from storing and archiving the works. The author hereby waives any claim against the University due to any way the users may use the works that is not in keeping with the legislation in force.
- c) The University shall adopt the necessary measures to safeguard the work in the future.
- d) The University reserves the right to withdraw the work, after notifying the author, in sufficiently justified cases, or in the event of third party claims.

Madrid, on ... 8th ... of July, 2019

HEREBY ACCEPTS



Signed... PALOMA CONTRERAS... PORRAS.....

Reasons for requesting the restricted, closed or embargoed access to the work in the Institution's Repository

I, hereby, declare that I am the only author of the project report with title:

INTUITIVE SIGNALING SYSTEM FOR MICROMOBILITY

which has been submitted to ICAI School of Engineering of Comillas Pontifical University in the academic year 2018/19. This project is original, has not been submitted before for any other purpose and has not been copied from any other source either fully or partially. All information sources used have been rightly acknowledged.

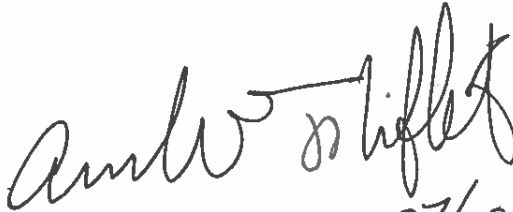


Fdo.: PALOMA CONTRERAS PORRAS

Date: 03/07/2019

I authorize the submission of this project

PROJECT SUPERVISOR



07/03/2019

Fdo.: ARNE FLIFLET

Date: DD/MM/2019



COMILLAS
UNIVERSIDAD PONTIFICIA

ICAI

GRADO EN INGENIERÍA EN TECNOLOGÍAS INDUSTRIALES

TRABAJO FIN DE GRADO

INTUITIVE SIGNALING SYSTEM FOR MICROMOBILITY

Autor: Contreras Porras, Paloma

Director: Fliflet, Arne

Madrid

July, 2019

RESUMEN DEL PROYECTO

1. Introducción

EL término micromovilidad se refiere a todo tipo de transporte personal ligero que puede transportar a uno o dos pasajeros. Algunos de los vehículos incluidos en esta definición son bicicletas, monopatines, scooters y patines, y predominantemente sus variantes eléctricas [noa19b].

Aunque la micromovilidad es cada vez más utilizada, las leyes para la seguridad y regulación de estos medios de transporte son actualmente obsoletas o incluso inexistentes.

1.1. Planteamiento del Problema

En el caso de las bicicletas, la normativa española exige una iluminación delantera en forma de luz blanca, y una iluminación trasera en forma de luz roja [BOE14]. En cuanto a la señalización, actualmente adopta la forma de señales manuales [DGT16a].

En cuanto a otras formas de transporte, como los patinetes eléctricos, sus reglamentos son aún más confusos. En España, los patinetes eléctricos se consideran VMP (Vehículos de Movilidad Personal) [RAC19], y sus límites deben ser establecidos por cada municipio [DGT16b]. Como tal, la señalización de la micromovilidad actual en su conjunto está fragmentada y no está normalizada en comparación con los sistemas equivalentes de los vehículos de motor.

El objetivo de este proyecto es resolverlo con un sistema de señalización universal para todo tipo de transporte de micromovilidad. Este sistema adopta la forma de un chaleco, con iluminación integrada y una caja de control que incluye, entre otros, un microprocesador y una unidad de medición inercial (IMU). A diferencia de otros sistemas disponibles en el mercado en la actualidad, este sistema proporciona una experiencia más intuitiva, ya que utiliza la detección de movimiento para realizar cambios de estado (como el apagado de los intermitentes una vez finalizado el giro o la señalización de frenado mediante la detección de desaceleración).

1.2. Estado del arte

La micromovilidad está en desarrollo, y cada día es más popular. Por esta razón, las regulaciones están siendo renovadas actualmente [DGT19b]. En este momento, la señalización de giro y parada viene en forma de señales manuales, y la iluminación sólo requiere algún tipo de iluminación delantera y trasera [DGT16a]. Para algunos medios de transporte, las regulaciones ni siquiera están centralizadas, dependiendo de cada ayuntamiento [DGT16b].

Sin embargo, aunque no es necesario tener más de un faro y una luz trasera, existen soluciones en el mercado que ofrecen seguridad adicional. Ejemplos de esto incluyen: faros, que están disponibles en una variedad de precios dependiendo de las diferentes características, como el ángulo del haz y el estilo de montaje; luces de freno, para las que rara vez se encuentran soluciones inteligentes, pero algunas se basan en un acelerómetro para detectar la desaceleración [Ama19a]; luces de giro, que se pueden encontrar desde 10 euros en Amazon [noa19a], y finalmente

dispositivos portátiles, en forma de chalecos y cascos, incluyendo luces delanteras y traseras, e incluso algunas luces de giro (que requieren ser apagadas manualmente) [Ama19b].

De toda la información anterior se puede concluir que las soluciones disponibles en el mercado de señalización de micromovilidad son poco inteligentes, y la mayoría de ellas se centran principalmente en las bicicletas. Además, hay muy pocas opciones que se puedan utilizar para más de un medio de transporte, ya que la mayoría de ellas están destinadas a ser colocadas en el vehículo y no en el pasajero. Además, para tener todas las características mencionadas anteriormente, el usuario necesita comprar más de un dispositivo. Por todo esto, la señalización de la micromovilidad tiene un margen de mejora considerable.

1.3. Objeto del proyecto

El objetivo de este proyecto es resolver el problema anteriormente planteado, mediante la creación de un chaleco de señalización universal, inteligente y utilizable para todas las formas de micromovilidad. El objetivo de este proyecto es proporcionar a los usuarios una señalización intuitiva y unificada con la facilidad de uso que ofrecen los sistemas de señalización para vehículos de motor.

Para lograr esto, algunos de los objetivos más específicos del proyecto son:

- Trabajar con sensores basados en movimiento para hacer la señalización más intuitiva para el usuario. Esto será en forma de luces de freno automáticas, que se activan al detectar la desaceleración del piloto, y luces de giro inteligentes, que se apagarán automáticamente cuando se haya completado el giro.
- Usar controles de botón explícitos cuando sea necesaria la entrada manual del usuario. Un ejemplo de ello es cuando se inicializan los intermitentes, de la misma manera que los intermitentes de los coches deben encenderse inicialmente de forma manual.
- La iluminación utilizada será muy visible y comprensible para los demás en la carretera. Esto se logrará mediante el uso de convenciones de señalización comunes en forma de faros en el pecho, luces de freno en la parte trasera e intermitentes similares a aquellos disponibles en los coches.

2. Metodología

Con el fin de alcanzar los objetivos anteriormente mencionados, el proyecto se ha dividido en tres subsistemas independientes, a saber, los subsistemas de control, iluminación y energía. La forma en que los tres están interconectados se puede ver en el diagrama de bloques en la Figura 1.

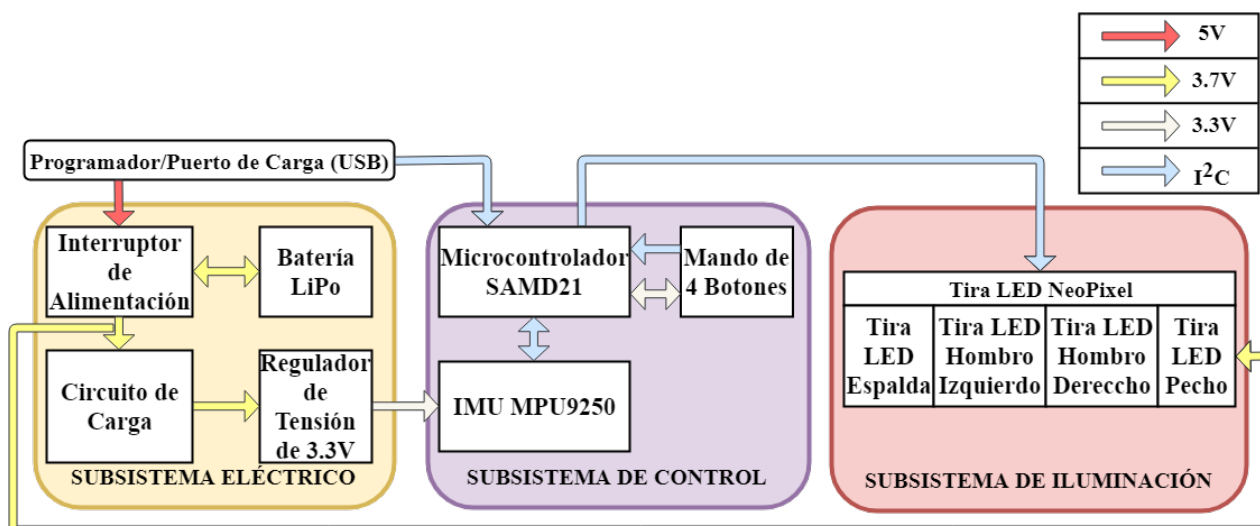


Figura 1. Diagrama de Bloques

La razón de esta partición es que sería posible trabajar por separado en los tres subsistemas diferentes, haciendo que su desarrollo sea más rápido y eficiente. En la etapa de desarrollo, las tres secciones podrían estudiarse y probarse por separado.

Para llegar a este sistema final tal y como está organizado en la Figura 1, se construyó un prototipo. Luego se estudió el primer modelo, hasta que se corrigieron los errores y se encontraron piezas innecesarias, lo que dio como resultado el producto propuesto en este proyecto.

Para el prototipo se utilizó la placa de de Sparkfun denominada SEN-14001, que incluye, entre otras, todas las características que este proyecto necesitaba. Además, se utilizaron pulsadores en lugar del mando a distancia que se pretende implementar en el modelo final. Por último, la batería y la tira de LEDs utilizada, que es de Adafruit [Ind19], no necesitan ser cambiadas para el producto real.

3. Resultados

Para el diseño del *hardware* de este proyecto, se eligió una tarjeta de circuito impreso (PCB). Este tipo de placa se puede conseguir a un precio razonable, si forma parte de una producción en masa, y también es más fácil diagnosticar errores en caso de que haya un componente defectuoso. Esto, junto con el hecho de que son muy ligeras y pueden ser diseñadas para ser muy pequeños, hacen de la PCB la mejor opción para un dispositivo portátil como el que se trata en este proyecto [Aga17].

El diseño de la tarjeta de circuito impreso para este proyecto puede verse en la Figura 2. Para una vista más detallada de este esquema, véase el apéndice B.

RESUMEN DEL PROYECTO

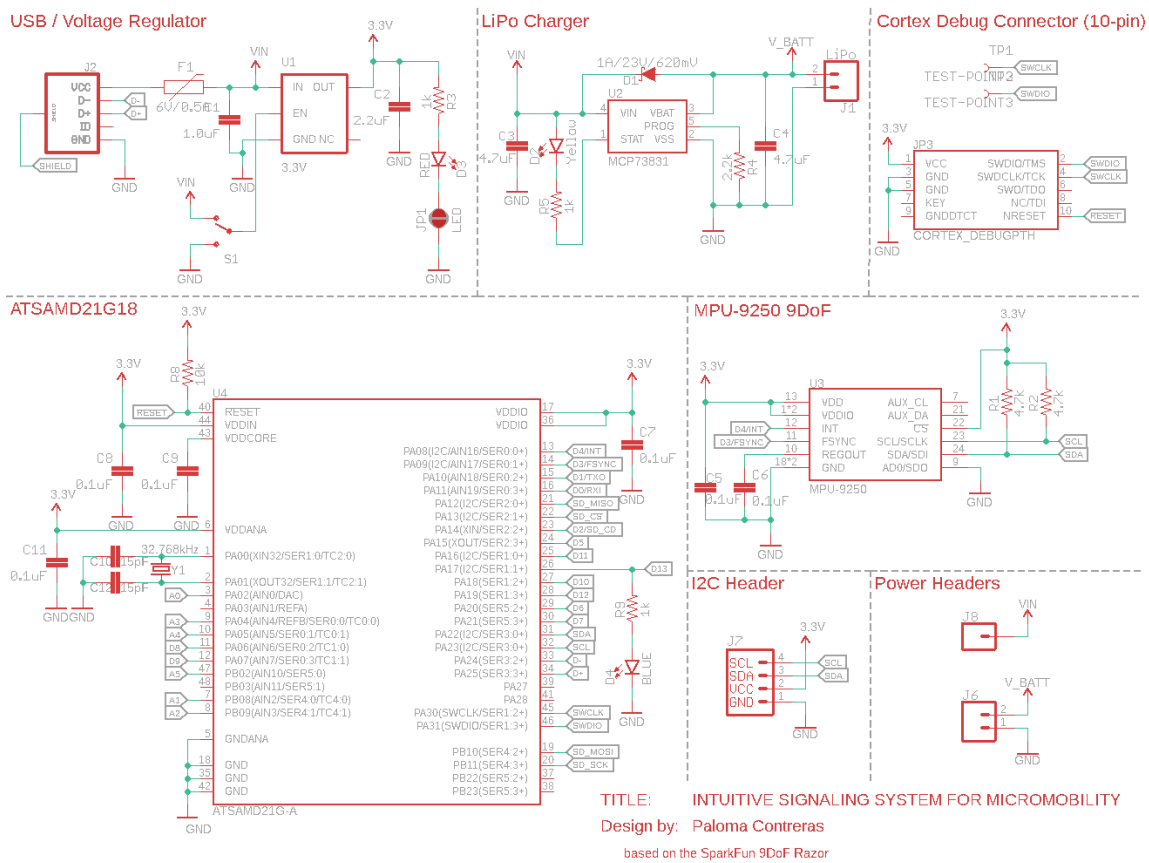


Figura 2. Esquema Eléctrico para la PCB

El sistema de señalización propuesto en este proyecto está integrado en un chaleco, que pretende ser de talla única para todos los usuarios. La distribución de los componentes del chaleco será la siguiente.

En primer lugar, para el subsistema de iluminación, el chaleco tendrá una tira LED NeoPixel única sobre los hombros y a lo largo de la parte superior de la espalda y el pecho [Ind19]. La tira de LEDs, a su vez, está conectada a una caja de control en el pecho del usuario. Esta caja de plástico duro, ligero y resistente al agua contiene tanto el subsistema de control como el de alimentación, ya que encapsula la placa de circuito impreso (o la placa SEN14001 en el caso del prototipo) y la batería. Para la entrada del usuario, un mando de 4 pulsadores es sostenido en la mano para activar los intermitentes, activar/desactivar las luces delanteras y las de emergencia y para salir de estado de colisión. En el diseño final, este mando está destinado a ser un mando a distancia.

Con respecto al *software*, para entender mejor el procedimiento que sigue el sistema, se puede ver una visión general del proceso en el diagrama de flujo en la Figura 3.

Cada proceso comienza y termina en el estado de reposo (idle). Además, a lo largo de todo el proceso, los sensores recogen información y la envían al microcontrolador. Estos datos se calibran (utilizando parámetros predefinidos, como se explica en el Capítulo 5) y se filtran. Dependiendo de cuáles sean las entradas, el microcontrolador enviará una orden diferente al subsistema de iluminación, para obtener una salida distinta en el sistema.

Este procedimiento se codifica mediante el uso de una máquina de estados, que se incluye en la Figura 4. Para una correcta comprensión de la máquina de estados, las claves utilizadas para representar las transiciones de estado se incluyen en la Tabla 1.

- **Idle** - Este es el estado principal en el que se encuentra el chaleco cuando no hace ninguna señalización. Aunque no se esté haciendo ninguna señalización, las luces no están apagadas, sino que tienen un patrón de luz colorido y tenue, para que el usuario sepa que el sistema está listo para recibir órdenes.

- **Giro a la Izquierda** - Este estado corresponde al intermitente izquierdo. Estas luces son de color naranja y dan la sensación de estar moviéndose del hombro derecho al izquierdo. Después de que este estado haya comenzado, hay dos maneras de hacer la transición al estado de reposo: una de ellas es automática, y ocurre cuando el giro se considera terminado; y la otra es presionando el mismo botón una vez más. Esta opción se ha incluido para que el usuario pueda rectificar su decisión.

- **Giro a la Derecha** - El estado de giro a la derecha es equivalente al giro a la izquierda, pero con dirección exactamente opuesta para la iluminación, y detección de espejo a lo largo del plano sagital [CLM19].

- **Luces de Emergencia** - Equivalentes a las luces de emergencia de un vehículo de motor [CLM19]; consisten en luces naranjas intermitentes. Este estado se introduce y sale manualmente pulsando el botón correspondiente.

- **Colisión** - Cuando se detecta un choque, el chaleco de seguridad detiene todos los demás procesos y convierte todas las luces en un color naranja parpadeante a máxima intensidad. Este estado sólo se abandona cuando se pulsa el botón de luces de emergencia y, a continuación, se vuelve al estado de reposo y se reanuda el comportamiento normal del sistema.

El frenado no se incluyó anteriormente porque no es un estado. La detección del frenado se realiza a lo largo de todo el proceso, y las luces de freno se pueden encender en cualquier momento (excepto desde el estado de colisión), sin interferir con el resto de las salidas. En cuanto a los faros delanteros, pueden estar encendidos al mismo tiempo que cualquier otra luz, excepto en el caso del estado de colisión. Estas luces se encienden y apagan utilizando el botón correspondiente del mando a distancia, pero esto no interfiere en la transición de estados.

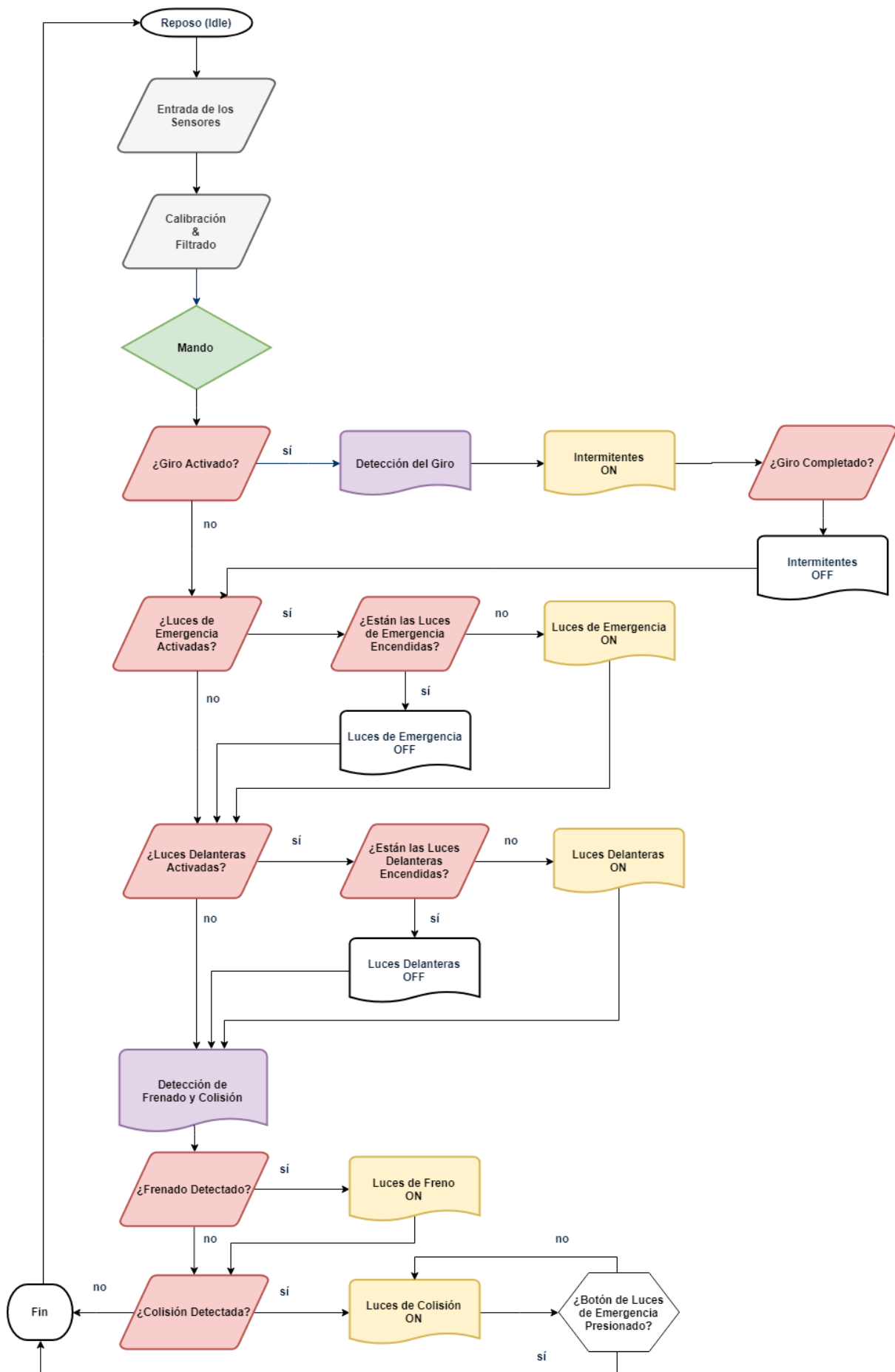


Figura 3. Diagrama de Flujo del Software

Abreviaturas para las Transiciones					
Abreviatura	E	D	I	Giro	Colisión
Acción	Botón de Luces de Emergencia Presionado	Botón de Intermitente Derecho Presionado	Botón de Intermitente Izquierdo Presionado	Giro Detectado	Colisión Detectada

Tabla 1. Abreviaturas de la Máquina de Estados

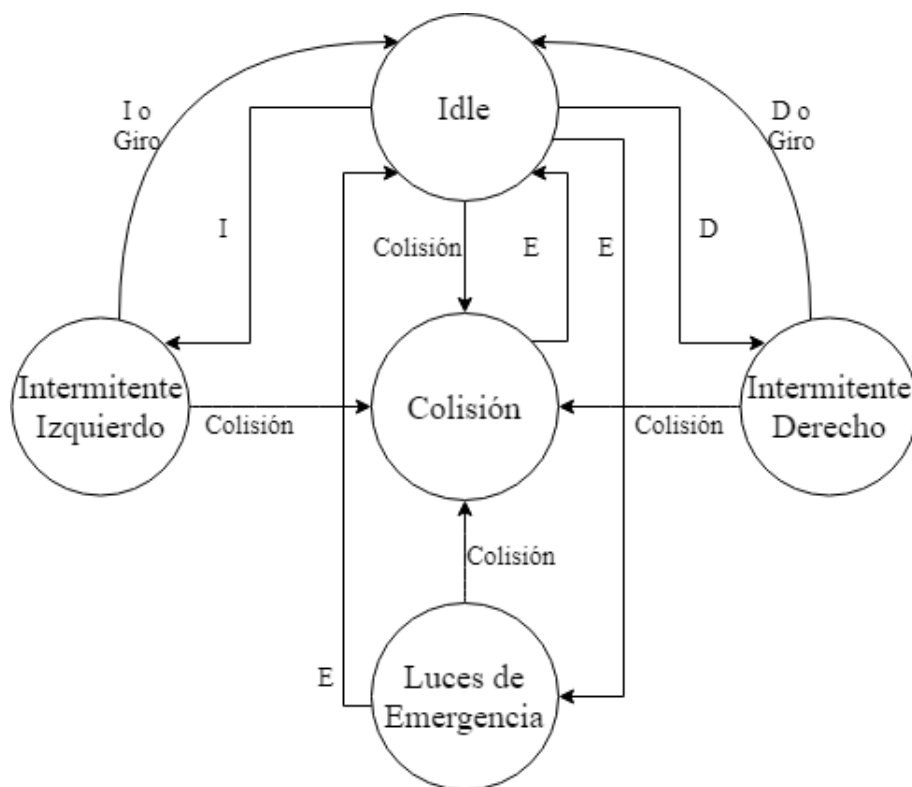


Figura 4. Máquina de Estados

Lo que hace que este producto sea más intuitivo para el usuario es que tiene la capacidad de cambiar automáticamente los estados detectando el movimiento del piloto. Esto se hace utilizando diferentes datos de los sensores. El proceso se explica a continuación.

- Detección del Giro** - Para detectar el giro se utilizaron dos variables: la aceleración en el eje Y (desplazamiento lateral para el usuario) y el ángulo alrededor del eje vertical del piloto. Estos dos valores son representativos para este propósito, ya que cambian significativamente durante un giro, pero tienen un comportamiento casi constante si el pasajero está siguiendo una trayectoria recta. Por ello, se puede detectar un giro cuando cada uno de estos valores ha superado un umbral determinado (es decir, que se está ejecutando el giro), y después ha tenido valores por debajo de otro umbral predeterminado (se ha retomado la trayectoria recta) [CLM19]. Estos umbrales se encontraron experimentalmente, durante la etapa de desarrollo, y se aseguró que cubrieran tanto un giro de noventa grados como un cambio de carril.
- Detección de Frenado** - La subrutina de frenado se ejecuta durante cada lazo principal independientemente del estado en que se encuentre el sistema, a menos que se haya

detectado una colisión. Para esta detección específica, se utiliza un control derivativo, con el fin de que la aceleración en el eje x (dirección delantera del piloto) sea más precisa y útil. Cuando la desaceleración del piloto alcanza un umbral predeterminado, las luces de freno se encienden [CLM19].

- **Detección de Colisión-** Al igual que la detección del freno, esta subrutina se ejecuta constantemente, utilizando datos de la IMU, para detectar un fallo en el instante en que este se produce. En este sistema, se considera que una colisión es una situación de caída vertical razonablemente grande, que se mide usando la aceleración en el eje z. Esta aceleración tiene poca variación durante el transporte, pero cambia repentinamente durante una colisión en la que se produzca la caída del piloto, lo que la hace adecuada para este propósito. Al igual que en secciones anteriores, la colisión se detecta cuando esta variable ha sobrepasado un umbral, que fue encontrado experimentalmente.

4. Conclusiones

Existen muchas opciones en el mercado para los sistemas de señalización, pero son arcaicos y no inteligentes. Así pues, esta idea supera a sus competidores en el mercado.

El chaleco propuesto en este proyecto tiene suficiente iluminación para hacer visible al piloto de noche, cumpliendo con la normativa vigente, y además añade visibilidad al usuario durante el día. Esto aumenta la seguridad para todas las personas que comparten un espacio con el piloto. Además, hace que la señalización sea más cómoda tanto para la persona que lleva el chaleco como para otras personas en la carretera, ya que incluye señales luminosas similares a las de los coches, que son fácilmente comprensibles.

Además, se ha incluido una nueva característica, que también mejora la seguridad. Este es el caso de las luces de caída, que pueden notificar a otros en la carretera de una colisión, o pueden ayudar a encontrar al ciclista si ha ocurrido en una zona de poco tráfico.

Este sistema es competitivo a lo que hay en el mercado en este momento, ya que proporciona características innovadoras pero no sube su precio, como se ha comprobado a partir de la investigación en el estado del arte y el análisis de costes en el Capítulo 7.

Por último, el hecho de que todas estas características estén incluidas en una sola pieza, que no necesita ser montada en el vehículo sino que sólo debe ser llevada por el usuario, la hace más versátil y accesible para todos. Además, su precio es aún más competitivo debido a esto, y a que no se necesitarían otras luces o señales.

5. Referencias

- [Aga17] T. Agarwal, “What Are The Advantages Of Using A Printed Circuit Board (PCB),” Aug. 2017, accessed 02/07/2019. [Online]. Available: <https://www.edgefx.in/advantages-using-printed-circuit-board-pcb/>
- [Ama19a] Amazon, “Cateye Kinetic X2 Rear: by Cateye: Amazon.es: Deportes y aire libre,” 2019, accessed 30/06/2019. [Online]. Available: https://www.amazon.es/Cateye-Kinetic-X2-Rear/dp/B01KXIX4CE/ref=as_li_ss_tl?ie=UTF8&linkCode=sl1&tag=prema0a-21&linkId=4240a9eec5008d01d93147fa988224d2&language=es_ES

- [Ama19b] —, “LED Light Vest,” 2019, accessed 26/06/2019. [Online]. Available: https://www.amazon.es/s?k=LED+Light+Vest&__mk_es_ES=%C3%85M%C3%85%C5%BD%C3%95%C3%91&ref=nb_sb_noss
- [BOE14] BOE, “Reglamento General de Vehículos - Actualizado 2014,” 2014, accessed 28/06/2019. [Online]. Available: <https://www.boe.es/buscar/act.php?id=BOE-A-1999-1826&b=53&tn=1&p=20140522#a22>
- [CLM19] P. Contreras, W. Lee, and D. Miller, “SignalMe,” Apr. 2019, accessed 02/07/2019. [Online]. Available: <https://courses.engr.illinois.edu/ece445/project.asp?id=7109>
- [DGT16a] DGT, “Guía para usuarios de la bicicleta,” 2016, accessed 12/06/2019. [Online]. Available: <http://www.dgt.es/Galerias/seguridad-vial/educacion-vial/recursos-didacticos/jovenes/Guia-Bicicleta-agosto-2016.pdf>
- [DGT16b] —, “Vehículos de Movilidad Personal (VMP),” 2016, accessed 30/06/2019. [Online]. Available: http://www.dgt.es/Galerias/seguridad-vial/normativa-legislacion/otras-normas/modificaciones/2016/Instr_16_V_124_Vehiculos_Movilidad_Personal.pdf
- [DGT19] —, “Real Decreto XXXX/2019 de medidas urbanas de tráfico.” 2019, accessed 30/06/2019. [Online]. Available: http://www.interior.gob.es/documents/642012/9796404/02_2019_Medidas_urbanas_de_trafico.pdf/dbde2652-6dce-4465-b260-767fae80be72
- [Ind19] A. Industries, “Adafruit NeoPixel LED Side Light Strip - Black 60 LED,” 2019, accessed 12/06/2019. [Online]. Available: <https://www.adafruit.com/product/3636>
- [noa19a] “10 Mejores Luces Intermitentes Bicicleta 2019,” 2019, accessed 30/06/2019. [Online]. Available: <https://bicicletaselectricas.xyz/luces-intermitentes-para-bicicleta/>
- [noa19b] “The Micromobility Conference, California 2019,” 2019, accessed 28/06/2019. [Online]. Available: <http://movmi.net/micromobility-conference-2019/>
- [RAC19] RACE, “¿Cómo circular con patinetes eléctricos? Normativa,” Mar. 2019, accessed 30/06/2019. [Online]. Available: <https://www.race.es/patinete-electrico-legislacion>

PROJECT SUMMARY

1. Introduction

MICROMOBILITY refers to all sorts of lightweight personal transportation that can carry one or two passengers [All18]. Some of the vehicles included in this definition are bicycles, skateboards, scooters, and roller-blades, and predominantly its electric variants [noa19b].

Even though micromobility is increasingly being used, the laws for the safety and regulation of these modes of transportation are currently obsolete or even non-existent.

1.1. Problem Statement

In the case of bicycles, regulations in Spain require forward illumination in the form of a white light, and rear illumination, in the form of a red light [BOE14]. As for signaling, it currently takes the form of arm-based signals [DGT16a].

As for other forms of transportation, such as electric scooters, their regulations are even more unclear. In Spain, electric scooters are considered VMP (Personal Mobility Vehicles) [RAC19], and their limits need to be set by each municipality [DGT16b]. As such, current micromobility signaling as a whole is fragmented and non-standardized compared to equivalent systems on motor vehicles.

The purpose of this project is to solve this with a universal signaling system for all types of micromobility transportation. This system takes the form of a vest, with embedded lighting and a control box including, but not limited to, a microprocessor and an Inertial Measurement Unit (IMU). As opposed to other solutions available on the market at the moment, this system provides a more intuitive experience, as it uses motion sensing to make state changes (such as switching off the turning lights after the turn has finished or signaling brake by detecting deceleration).

1.2. State of the Art

Micromobility is under development, and becoming more popular everyday. For this reason, regulations are being renewed these days [DGT19b]. Right now, turn and stop signaling comes in the form of hand signals, and lighting only requires some kind of forward and rear illumination [DGT16a]. For some modes of transportation, regulations are not even centralized, depending on each municipality [DGT16b].

However, while it is not necessary to have more than one headlight and one rear light, there are solutions on the market that offer additional safety. Examples of this include: headlights, that are available in a variety of prices depending on different characteristics, such as beam angle and mounting style [Rem18]; brake lights, for which intelligent solutions are rarely found, but some rely on an accelerometer to detect deceleration [Ama19a]; turning lights, which can be found from 10 euros in Amazon [noa19a], and finally wearable devices, in the form of vests and helmets, including front and rear lights, and some even turning lights (which require to be turned off manually) [Ama19b].

Of all the information above it can be concluded that solutions available in the market for micromobility signaling are unintelligent, and the majority of them are mainly focused in

bicycles. Furthermore, there are very little options that can be used for more than one mode of transportation, as most of them are intended to be placed on the vehicle, rather than on the passenger. Moreover, to have all the features mentioned above, the user needs to buy more than one device. On the whole, micromobility signaling has a considerable margin for improvement.

1.3. Project Objective

The objective of this project is to solve the previously stated problem, by creating a universal signaling vest that is smart and usable for all forms of micromobility. The purpose of this project is to give users intuitive and unified signaling with the ease of use found in the signal systems for motor vehicles.

To achieve this, some of the more specific objectives of the project are:

- To work with motion-based sensors to make the signalization more intuitive to the user. This will be in the form of automatic brake lights, that get triggered by detecting the deceleration of the rider, and intelligent turning lights, which will automatically turn off when the turn has been completed.
- To use explicit button controls when manual input from the user is necessary. One such example is when turn signals are initialized, in the same way as cars' turns signals need to be initially turned on manually.
- The lighting used will be highly visible and understandable for others on the road. This will be achieved by using common signal conventions in the form of headlights on the chest, brake lights on the back, and left and right turn signals around the shoulders.
- As an additional feature, the vest will also provide a high visibility flashing hazard operation that activates when a user has an accident that causes a fall.

2. Methodology

In order to achieve the previously stated objectives, the project has been divided into three separate subsystems, namely control, lighting and power subsystems. The way in which the three of them are interconnected can be seen in the block diagram in Figure 5.

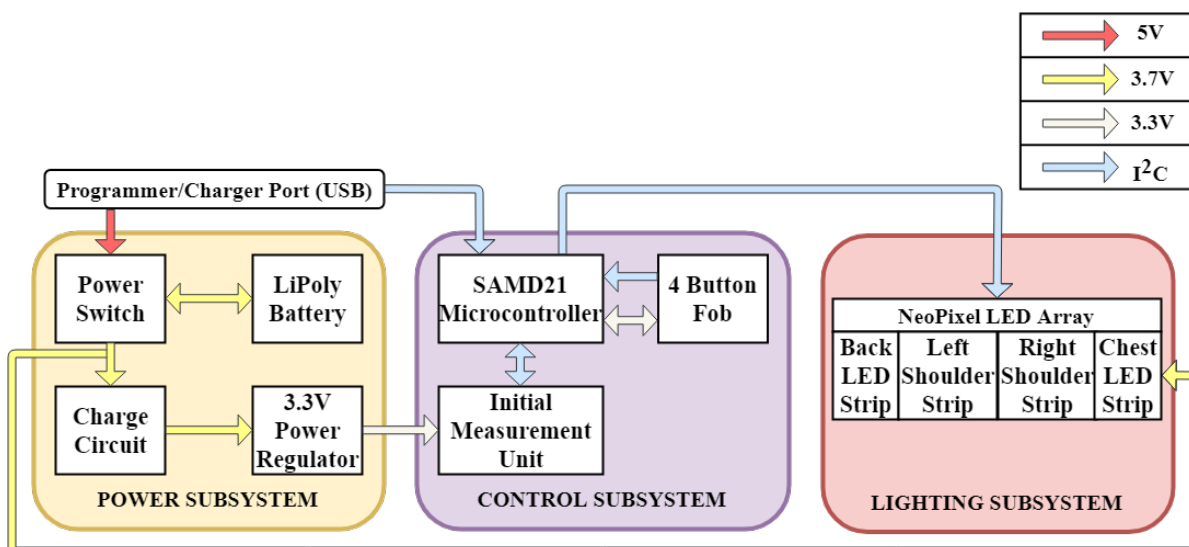


Figure 5. Block Diagram

The reason for this partition is that it would be possible to work separately on the three different subsystems, making their development faster and more efficient. In the development stage, all three sections could be studied and tested separately.

In order to reach this final system as it is organized in Figure 5, a prototype was built. The first model was then studied, until errors were corrected and unnecessary parts were found, and it gave as a result the product proposed in this project.

For the prototype, the breakout board from Sparkfun named SEN-14001 was used, which includes, among others, all the features that this project needed [Ele19]. Furthermore, push-buttons were used instead of the remote control that is intended to be implemented in the final model. Lastly, the battery and the LED strip used, which is from Adafruit [Ind19], do not need to be changed for the real product.

3. Results

For the hardware design of this project, a Printed Circuit Board (PCB) was the option chosen. This type of board can be achieved at a reasonable price, if part of a mass production, and is also easier to diagnose in case that there is a failed component. This, along with the fact that they are very light and can be designed to be very small, make the PCB the best option for a wearable like the one addressed in this project [Aga17].

The design for the Printed Circuit Board for this project can be seen in Figure 6. For a more detailed view of this schematic, see appendix B.

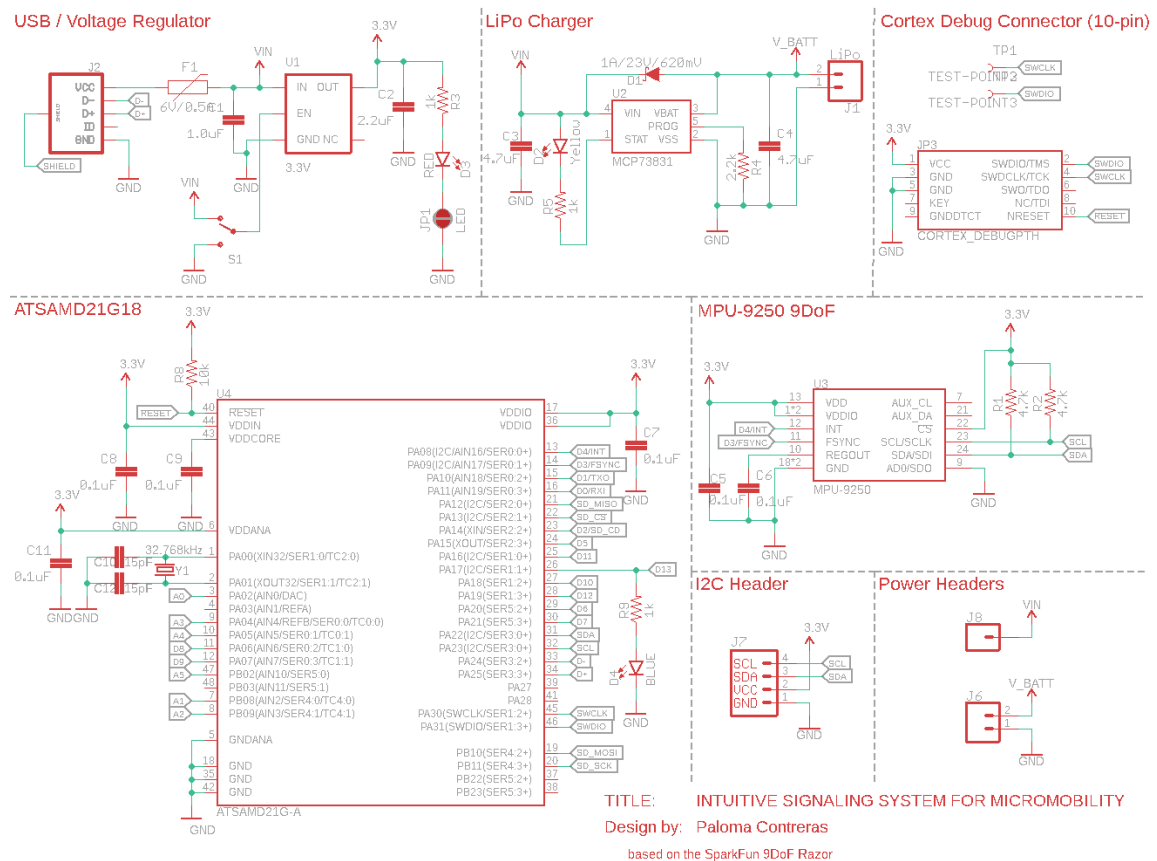


Figure 6. Board Schematic

The signaling system proposed in this project is integrated into a vest, which is intended to be a one size fits all wearable. The distribution of the components in the vest will be as follows.

First of all, for the lighting subsystem, the vest will have a unique NeoPixel LED strip over the shoulders and along the upper back and chest [Ind19]. The LED strip is, in its turn, wired to a control box on the chest of the user. This water resistant lightweight hard plastic box contains both the control and the power subsystems, as it encapsulates the PCB (or the breakout board in the case of the prototype), and the battery. For user input, a 4 button fob is held in the hand to initiate turns, toggle hazard mode, toggle the chest lighting and finish the crash state. In the final design, this fob is intended to be exchanged for a remote control.

With respect to the software, to best understand the procedure that the system follows, a general overview of the process can be seen in the flowchart in Figure 7.

Every process starts and ends in the idle state. Besides, throughout the whole process, the sensors are gathering information and sending it to the microcontroller. This data is then calibrated (using predefined parameters, as explained in chapter 5) and filtered. The fob also sends the user's button inputs to the microcontroller. Depending on what the inputs are, the microcontroller will send a different order to the lighting subsystem, to get a different output in the system.

This procedure is coded by the use of a state machine, which is included in Figure 8. For the correct understanding of the state machine, the keys used to represent state transitions are included in Table 2.

- **Idle** - This is the main state the vest is in when not doing any signaling. Even though no signaling is being done, the lights are not turned off; on the contrary, they have a colorful and dim light pattern, so that to let the user know that the system is ready to receive orders.
- **Left Turn** - This state corresponds to a left turn signal. These lights are orange and give the sensation of being moving from the right to the left shoulder. After this state has started, there are two ways to transition to idle state: one of them is automatic, and happens when the turn has been considered finished; and the other is by pressing the same button one more time. This option has been included so that the user can rectify their decision.
- **Right Turn** - The Right Turn state is equivalent to the Left Turn, but with exact opposite direction for the lighting, and mirroring detection along the sagittal plane [CLM19].
- **Hazards** - Equivalent to a motor vehicle's hazard lights; they consist on blinking orange lights. This state is entered and exited manually by pressing the corresponding button.
- **Crash** - When a crash is detected, the vest will stop all other processes and turn all lights to a blinking orange at full brightness. This state is only exited when hazard button is pressed, and then the normal behavior of the system is resumed.

Braking was not included above because it is not a state. Braking detection is done throughout the whole process, and brake lights can be turned on at any moment (except from crash state), not interfering with the rest of the outputs. As for headlights, they can be on at the same time as any other light, except for the case of the crash state. These lights are turned on and off by using the corresponding button from the fob, but this does not interfere in the state transition.

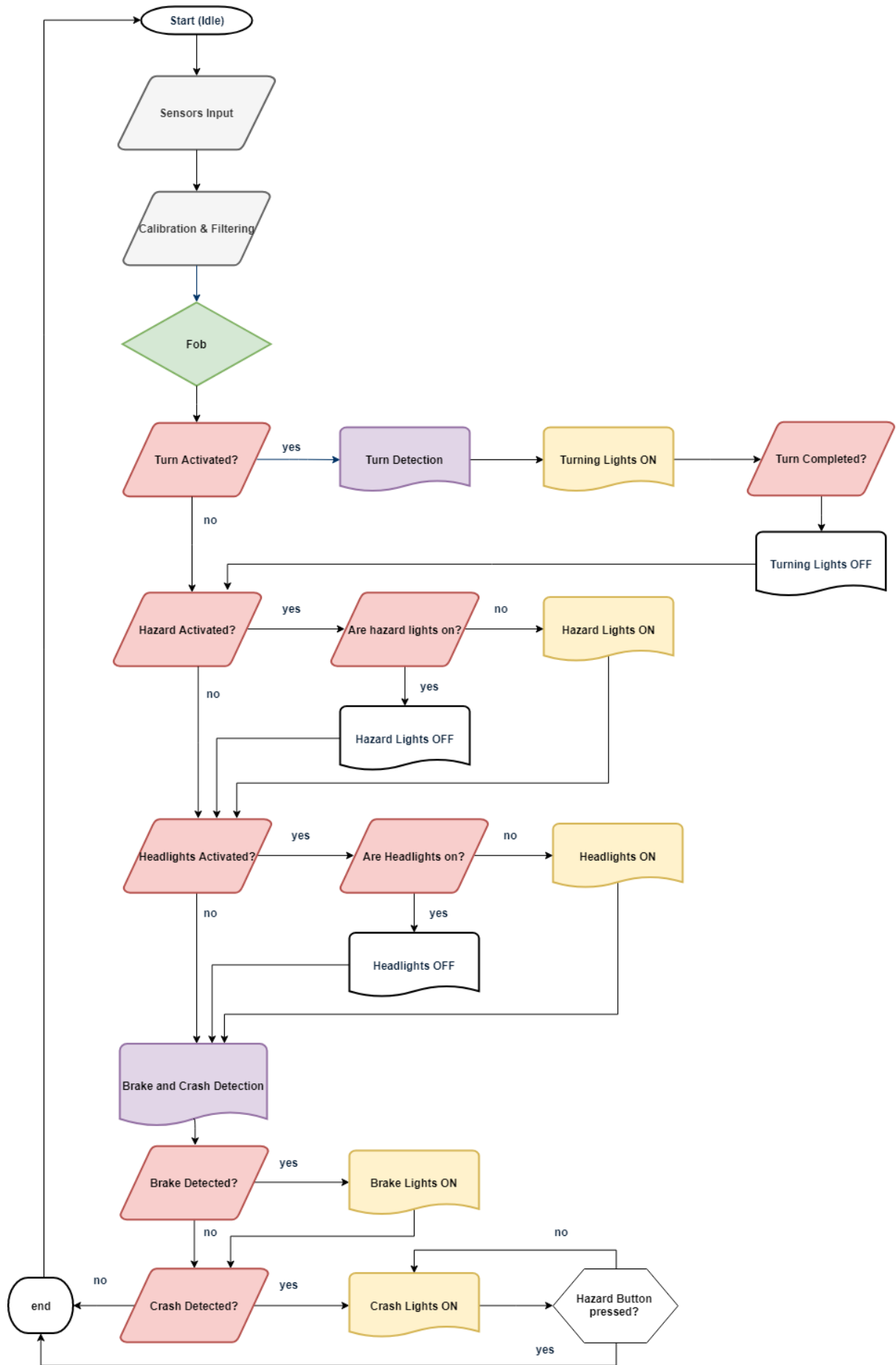


Figure 7. Software Flowchart

Transition Keys					
Key	H	R	L	Turn	Crash
Event	Hazard Button Pressed	Right Button Pressed	Left Button Pressed	Turn Detected	Crash Detected

Table 2. State Transition Keys

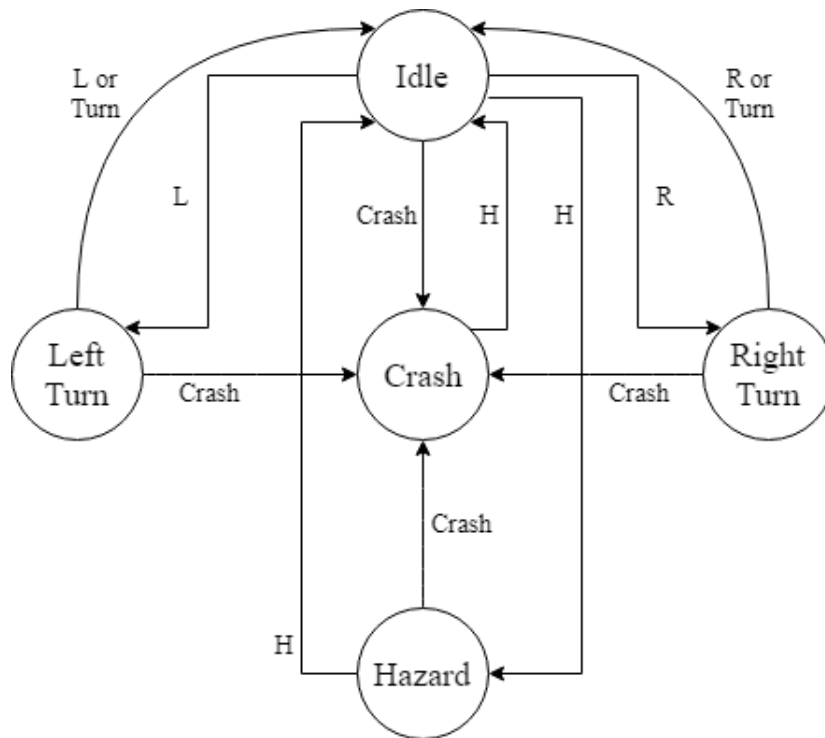


Figure 8. State Machine

What makes this product more intuitive for the user is that it has the ability to automatically change states by detecting the movement of the rider. This is done by using different data from the sensors. The process is explained below.

- Turn Detection** - In order to detect the turn, two variables were used: the acceleration in the y-axis (lateral displacement for the user) and the yaw (angle around the vertical axis of the rider). These two values are representative for this purpose, as they change significantly during a turn, but have an almost constant behavior if the passenger is following a straight path. For that reason, a turn can be detected when each of these values have gone over a specified threshold (meaning the turn is being executed), and after that have had values below another predetermined threshold (the straight path has been retaken) [CLM19]. These thresholds were found experimentally, during the development stage, and it was ensured that they covered both a ninety-degree turn and a lane change.
- Brake Detection** - The braking subroutine runs during every main loop regardless of the state the system is in, unless a crash has been detected. For this specific detection, a derivative control is used, in order to make the acceleration in the x-axis (front direction of the rider) to be more precise and useful. When the deceleration of the rider reaches a predetermined threshold, the brake lights turn on [CLM19].

- **Crash Detection** - In the same way as the brake detection, this subroutine is constantly being executed, using sampled IMU data to detect a crash the instant it happens. In this system, a crash is considered to be a situation of a reasonably large vertical drop, which is measured using the acceleration in the z-axis. This acceleration has little variance during the ride, but changes suddenly during a fall, which makes it suitable for this purpose. As in previous sections, the crash is detected when this variable has overstepped a threshold, which was found experimentally.

4. Conclusions

There are many options in the market for signaling systems, but they are archaic and non-intelligent. Thus, this idea outperforms its competitors on the market.

The vest proposed in this project has sufficient lighting to make a rider visible at night, complying with current regulations, and also adds visibility to the user during daylight. This increases safety for all the people who share a space with the rider. Furthermore, it makes the signage more convenient for both the person wearing the vest and others on the road, as it includes lighting signs similar to those of cars, which are easily understandable.

Moreover, a new feature has been included, which also improves safety. This is the case of the crash lights, that can notify others on the road of a fall, or can help find the rider if it has happened on a poorly trafficked area.

This system is competitive to what there is on the market at the moment, as it provides innovative features but does not add to the price, as it has been proven from the research in chapter 2 and the cost analysis in chapter 7.

Lastly, the fact that all these features are included in one piece, which does not need to be mounted on the vehicle but only be worn by the user, makes it more versatile and accessible for everyone. Furthermore, its price is even more competitive because of this, due to the fact that no other lights or signals would be necessary.

5. References

- [Aga17] T. Agarwal, “What Are The Advantages Of Using A Printed Circuit Board (PCB),” Aug. 2017, accessed 02/07/2019. [Online]. Available: <https://www.edgex.in/advantages-using-printed-circuit-board-pcb/>
- [All18] M. Alliance, “How Micro Mobility Solves Multiple Problems in Congested Cities,” Jul. 2018, accessed 12/06/2019. [Online]. Available: <https://maas-alliance.eu/how-micro-mobility-solves-multiple-problems-in-congested-cities/>
- [Ama19a] Amazon, “Cateye Kinetic X2 Rear: by Cateye: Amazon.es: Deportes y aire libre,” 2019, accessed 30/06/2019. [Online]. Available: https://www.amazon.es/Cateye-Kinetic-X2-Rear/dp/B01KXIX4CE/ref=as_li_ss_tl?ie=UTF8&linkCode=sll1&tag=prema0a-21&linkId=4240a9eec5008d01d93147fa988224d2&language=es_ES

- [Ama19b] —, “LED Light Vest,” 2019, accessed 26/06/2019. [Online]. Available: https://www.amazon.es/s?k=LED+Light+Vest&__mk_es_ES=%C3%85M%C3%85%C5%BD%C3%95%C3%91&ref=nb_sb_noss
- [BOE14] BOE, “Reglamento General de Vehículos - Actualizado 2014,” 2014, accessed 28/06/2019. [Online]. Available: <https://www.boe.es/buscar/act.php?id=BOE-A-1999-1826&b=53&tn=1&p=20140522#a22>
- [CLM19] P. Contreras, W. Lee, and D. Miller, “SignalMe,” Apr. 2019, accessed 02/07/2019. [Online]. Available: <https://courses.engr.illinois.edu/ece445/project.asp?id=7109>
- [DGT16a] DGT, “Guía para usuarios de la bicicleta,” 2016, accessed 12/06/2019. [Online]. Available: <http://www.dgt.es/Galerias/seguridad-vial/educacion-vial/recursos-didacticos/jovenes/Guia-Bicicleta-agosto-2016.pdf>
- [DGT16b] —, “Vehículos de Movilidad Personal (VMP),” 2016, accessed 30/06/2019. [Online]. Available: http://www.dgt.es/Galerias/seguridad-vial/normativa-legislacion/otras-normas/modificaciones/2016/Instr_16_V_124_Vehiculos_Movilidad_Personal.pdf
- [DGT19] —, “Real Decreto XXXX/2019 de medidas urbanas de tráfico,” 2019, accessed 30/06/2019. [Online]. Available: http://www.interior.gob.es/documents/642012/9796404/02_2019_Medidas_urbanas_de_trafico.pdf/dbde2652-6dce-4465-b260-767fae80be72
- [Ele19] S. Electronics, “SparkFun 9dof Razor IMU M0 - SEN-14001,” 2019, accessed 12/06/2019. [Online]. Available: <https://www.sparkfun.com/products/14001>
- [Ind19] A. Industries, “Adafruit NeoPixel LED Side Light Strip - Black 60 LED,” 2019, accessed 12/06/2019. [Online]. Available: <https://www.adafruit.com/product/3636>
- [noa19a] “10 Mejores Luces Intermitentes Bicicleta 2019,” 2019, accessed 30/06/2019. [Online]. Available: <https://bicicletaselectricas.xyz/luces-intermitentes-para-bicicleta/>
- [noa19b] “The Micromobility Conference, California 2019,” 2019, accessed 28/06/2019. [Online]. Available: <http://movmi.net/micromobility-conference-2019/>
- [RAC19] RACE, “¿Cómo circular con patinetes eléctricos? Normativa,” Mar. 2019, accessed 30/06/2019. [Online]. Available: <https://www.race.es/patinete-electrico-legislacion>
- [Rem18] R. Remick, “The 7 Best Bike Lights Reviewed & Compared For 2019,” Apr. 2018, accessed 30/06/2019. [Online]. Available: <https://www.outsidepursuits.com/best-bike-headlight-reviews/>

PART I



PROJECT REPORT



Table of Contents

1. Introduction	13
1.1. Problem Statement	13
2. State of the Art	15
3. Description of the Developed Model	17
3.1. Project Objectives	17
3.2. System Overview	17
3.3. Methodology	19
4. Hardware Design	21
4.1. Main Components	21
4.1.1. Microcontroller	21
4.1.2. Inertial Measurement Unit	21
4.1.3. LED Strip	22
4.1.4. LiPo Battery	22
4.2. Board Schematic	22
4.3. Physical Design	23
5. Sensor Noise Reduction	25
5.1. Filtering	25
5.2. Calibration	25
6. Software Design	27
6.1. Software Flowchart	27
6.2. State Machine	29
6.3. State Transitions	30
6.3.1. Turn Detection	30
6.3.2. Brake Detection	32
6.3.3. Crash Detection	32
6.4. Lighting States	33
6.4.1. Idle	33
6.4.2. Right and Left Turn	34
6.4.3. Hazard	34
6.4.4. Crash	35
6.4.5. Brake	35
6.4.6. Headlights	36
7. Project Budget	37

8. Conclusions	41
8.1. Conclusions on Results	41
8.2. Conclusions on Methodology	41
8.3. Future Improvements	42
Bibliography	43
II. Appendices	47
A. Complete Code	49
A.1. Main Program - State Machine	51
A.1.1. main.cpp	51
A.2. Turn, Crash and Brake Detection	53
A.2.1. detection.cpp	53
A.2.2. detection.hpp	55
A.3. Libraries for Lighting Control	55
A.3.1. lighting.cpp	55
A.3.2. lighting.hpp	57
B. Board Schematic	59

List of Figures

1. Regulated Bicycle Signals - Right Turn	13
2. Regulated Bicycle Signals - Left Turn	14
3. Regulated Bicycle Signals - Stop	14
4. Block Diagram	18
5. Board Schematic	23
6. Physical Design	24
7. Software Flowchart	28
8. State Machine	30
9. Turn Detection Framework	31
10. Turn Detection Flowchart	31
11. Brake Detection Flowchart	32
12. Crash Detection Flowchart	33
13. Detailed Board Schematic	61

List of Tables

3. State Transition Keys	29
4. Bill of Materials	38

List of Code Extracts

1. rainbowCycle function	34
2. RunningLights function	34
3. Hazards function	35
4. Crash state code	35
5. Braking lights code	36
6. Headlights code	36

Acronyms

<i>ICAI</i>	Instituto Católico de Artes e Industrias
<i>UIUC</i>	University of Illinois at Urbana-Champaign
<i>IMU</i>	Inertial Measurement Unit
<i>VMP</i>	Vehículos de Movilidad Personal
<i>9DoF</i>	9 Degrees of Freedom
<i>PCB</i>	Printed Circuit Board
<i>DCM</i>	Direction Cosine Matrix

Chapter 1

Introduction

MICROMOBILITY refers to all sorts of lightweight personal transportation that can carry one or two passengers [All18]. Some of the vehicles included in this definition are bicycles, skateboards, scooters, and roller-blades, and predominantly its electric variants, as mentioned in this year's micromobility conference [noa19b].

1.1. Problem Statement

Even though micromobility is increasingly recurrent, the laws for the safety and regulation of these modes of transportation are currently obsolete or even non-existent.

In the case of bicycles, regulations in Spain require forward illumination in the form of a white light, and rear illumination, in the form of a red light [BOE14]. As for signaling, it currently takes the form of arm-based signals, as shown in Figure 1, Figure 2 and Figure 3 [DGT16a].

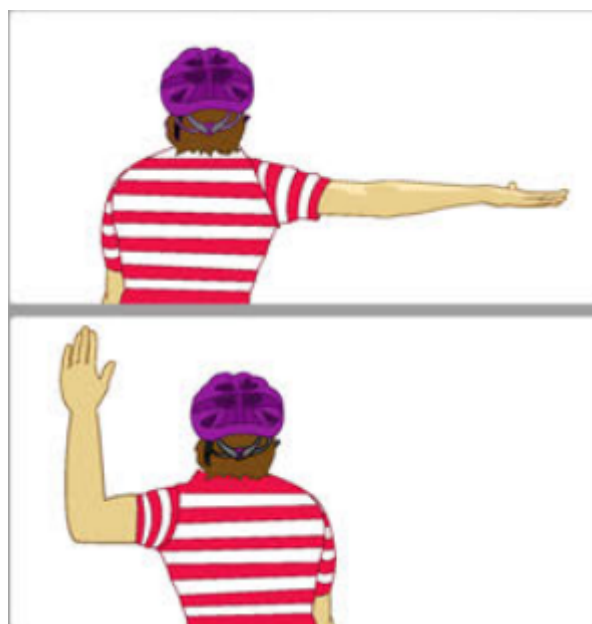


Figure 1. Regulated Bicycle Signals - Right Turn

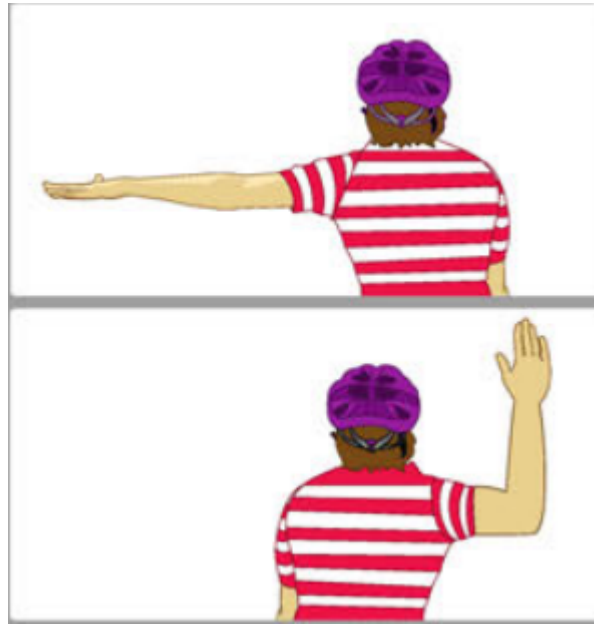


Figure 2. Regulated Bicycle Signals - Left Turn



Figure 3. Regulated Bicycle Signals - Stop

As for other forms of transportation, such as electric scooters, their regulations are even more unclear. In Spain, electric scooters are considered VMP (Personal Mobility Vehicles) [RAC19], and their limits need to be set by each municipality [DGT16b]. As such, current micromobility signaling as a whole is fragmented and non-standardized compared to equivalent systems on motor vehicles.

The purpose of this project is to solve this with a universal signaling system for all types of micromobility transportation. This system takes the form of a vest, with embedded lighting and a control box including, but not limited to, a microprocessor and an Inertial Measurement Unit (IMU). As opposed to other solutions available in the market at the moment, this system provides a more intuitive experience, as it uses motion sensing to make state changes (such as switching off the turning lights after the turn has finished or signaling brake by detecting deceleration).

This document is original, but the development of the idea behind it was mostly completed in collaboration with Derrick Miller and Wonyong Lee.

Chapter 2

State of the Art

MICROMOBILITY transportation is under development, and becoming more popular everyday. However, 72% of Spanish citizens think of it as "a risk to road safety" [DGT19a]. And this is for a reason: 7.5% of the accidents in Spain have a cyclist involved [ABC18], and there were also 273 accidents involving an electric scooter in 2018 [Pas18].

Regulations are also being renewed these days [DGT19b]. Right now, turn and stop signaling comes in the form of hand signals, and lighting only requires some kind of forward and rear illumination [DGT16a]. For some modes of transportation, regulations are not even centralized, depending on each municipality [DGT16b].

However, while it is not necessary to have more than one headlight and one rear light, there are solutions on the market that offer additional safety. Examples of this include:

- **Headlights** - This type of lights are compulsory, and for that reason there is a wider range of solutions available. Some of the aspects to take into account when choosing headlights are impermeability, beam angle, battery type and mounting style [Rem18]. This last feature is the one determining if the lights can be used for one or more micromobility vehicles. Headlights are available in a wide range of prices.
- **Brake Lights** - There are some solutions in the market for this feature. Most of the rear lights available are static [Tri19], which do not serve as brake lights. There are also some solutions which are exclusive for bicycles, as they are mechanically attached to the back wheel to detect the brake [Vic16]. Finally, there are also some more intelligent features, involving an accelerometer to detect the brake [Arr17]. This solution is available for over 30 euros in Amazon [Ama19a].
- **Turning Lights** - Even though these lights are not compulsory nowadays, being able to warn the other vehicles on the street about your intentions can significantly improve safety. There are solutions that can be attached to the vehicle and can be bought from 10 euros in Amazon [noa19a].
- **Wearables** - The main advantage of this kind of solutions is that they can be used in several modes of transport, as they are not attached to the vehicle, but to the passenger. Currently, there exist vests including front and rear LED lights (some even include turning lights) that can be bought from 20 euros [Ama19b]. Helmets with music and turning lights are also available from 90 euros in Amazon [noa19a]. However, these solutions need the user input to turn off the lights when the turn has been completed.

Of all the information above it can be concluded that solutions available in the market for micromobility signaling are unintelligent, and the majority of them are mainly focused in bicycles. Furthermore, there are very little options that can be used for more than one mode of transportation, as most of them are intended to be placed on the vehicle, rather than on the passenger. Moreover, to have all the features mentioned above, the user needs to buy more than one device. On the whole, micromobility signaling has a considerable margin for improvement.

Chapter 3

Description of the Developed Model

THE model proposed in project consists of the development of a vest with a built-in lighting system, which receives orders from a microcontroller [Atm15]. This, on its turn, provides the correct commands by processing 4-button inputs, as well as the sensor information provided by an IMU [Inv16]. This chapter presents the problem that needs to be addressed as well as an overview of the proposed solution.

3.1. Project Objectives

The main objective of this project is to solve the scourge found in today's signaling systems for micromobility, by creating a universal signaling vest that is smart and usable for all these modes of transport. The purpose of this project is to give users intuitive and unified signaling with the ease of use found in the signal systems for motor vehicles [CLM19]. In order to achieve this, the specific objectives of the project are as follows:

- To work with motion-based sensors to make the signalization more intuitive to the user. This is achieved by the utilization of an Inertial Measurement Unit, whose functioning will be explained in subsection 4.1.2.
- To use explicit button controls when manual input from the user is necessary. One such example is the initialization of the turn signals or hazard lights.
- The lighting used will be highly visible and understandable for others on the road. This will be achieved by using common signal conventions in the form of forward illumination on the chest, brake lighting on the back, hazard lighting similar to what is available on cars, and left and right turn signals around the shoulders.
- As an additional feature, the vest will also provide a high visibility flashing hazard operation that activates when a user has an accident and consequently falls to the ground [CLM19].

3.2. System Overview

In order to achieve the previously stated objectives, the project has been divided into three separate subsystems, namely control, lighting and power subsystems. The way in which the three of them are interconnected can be seen in the block diagram in Figure 4.

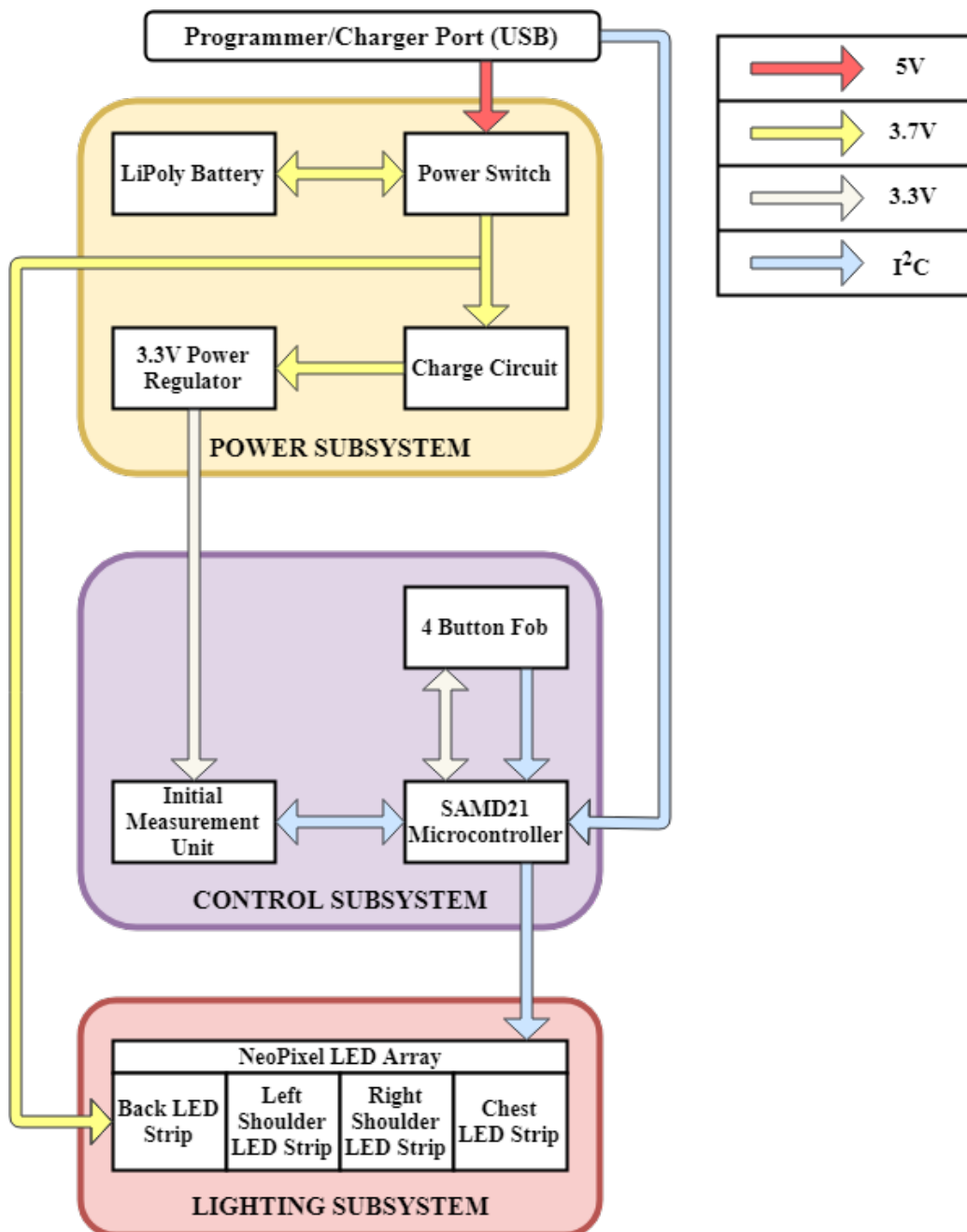


Figure 4. Block Diagram

The reason for this partition is that it would be possible to work separately on the three different subsystems, making their development faster and more efficient. In the development stage, all three sections could be studied and tested separately.

The functions performed by each of the subsystems consist of the following:

- Control Subsystem** - The control subsystem processes the inputs from both buttons and sensors, in order to produce control messages for the lighting subsystem. A microcontroller and inertial measurement unit (IMU) are used to determine the passenger's acceleration in different directions, as well as the yaw, pitch and roll. This information is then processed to determine the state transitions, which is finally sent to the lighting subsystem .

- **Lighting Subsystem** - The lighting subsystem takes commands from the control subsystem explained above, over NeoPixel serial one wire [Ind19] and produces light patterns that are consistent with the signals that the user intends to give.
- **Power Subsystem** - The power subsystem allows for battery operation of the vest, charging of the on-board battery, and distribution of power to the appropriate subsystems in the form of direct battery potential for lighting and regulated 3.3V for the micro-controllers and sensors [CLM19].

3.3. Methodology

The way in which this project was approached is outlined in this section. First of all, a design was made as to find what was thought to be the necessary parts to build the system. At this time, research was done in order to find how to refine sensor data to be able to gather useful information for the purpose of this project. This research is explained in chapter 5.

After all the research had been done, and the parts were available, both the development for the lighting and control subsystems were initiated.

For the control subsystem, experiments were run as to find the data that was representative for each of the detection processes. This will be further explained in section 6.3. During this experimental phase, the SD card that was included in the prototype was very useful. Nevertheless, this feature has been deleted from the final product, as it is of no use to the user.

At the same time, the lighting subsystem was developed, by studying the NeoPixel library and how it could be used to create the desired signals.

Lastly, the complete prototype was put together, joining the control and lighting subsystem, and powering them properly. Having done this, it was possible to carry new experiments, in order to find the thresholds that would trigger each detection (this is further explained in section 6.3). When this was achieved, the model for the system was accomplished.

Chapter 4

Hardware Design

THIS chapter is focused on the explanation of the components that have been used, as well as their layout in the resulting model. The sections included below include a list of the main components and their role within the project, a board schematic, and the physical design, which explains the distribution of the devices on the vest.

4.1. Main Components

For the proper understanding of the design, it is indispensable to know the main components that allow its behavior. In the following subsections, there will be an explanation on the devices chosen, as well as the reason why they were selected among similar ones.

Furthermore, to better understand how all these components are connected to one another, both the block diagram in Figure 4 and the board schematic in section 4.2 can be consulted.

4.1.1. Microcontroller

The microcontroller chosen for this project is the SAMD21 [Atm15]. This device is powerful enough for our purpose and future developments, while entering within the budget necessary for the final product to be competitive, as it will be seen in chapter 7. Additionally, it is also convenient because of its compatibility with the Arduino library.

The primary function of this component is to take all user and sensor inputs, process the information and send to the lighting subsystem the necessary orders for the correct output.

In addition, to ensure that the microcontroller is easy to repair in the event of a fault being found, a debug connector has been included in the board schematic, as seen in Figure 5 and Figure 13.

4.1.2. Inertial Measurement Unit

In order to make the vest an intuitive signaling method, it is essential to collect information from sensors built into the vest. For this reason, an Inertial Measurement Unit (IMU) has been implemented.

The device chosen for this project is the MPU9250 [Inv16], which is a 9 degrees of freedom (hereon 9DoF) IMU. This component has an accelerometer, a gyroscope and a magnetometer.

By using these three types of sensors, it is possible to collect enough information to program different outputs that depend on the user's motion.

As it will be explained in further detail later in section 6.3, of all the information provided by the IMU, only some of it is directly used for the scope of this project. Thus, the x-axis acceleration is used for brake detection, while the z-axis acceleration is needed for crash detection. For the turn detection, both the y-axis acceleration and the yaw are used.

While all accelerations mentioned above are provided by the accelerometer, all the angle information originates in the gyroscope. This includes the yaw, mentioned above, as well as pitch and roll angles, but these two have not been necessary for this project. As for the magnetometer, its purpose is not to give sensor output that is directly being used, but to improve the information that is being perceived by the microcontroller.

4.1.3. LED Strip

The device explained in this subsection is the LED strip used for the lighting of the vest. This is the visual output of the signaling system and, therefore, an essential part of this project.

For this important part, the device chosen is the NeoPixel single-wire LED strip by Adafruit [Ind19]. The reason that these LEDs were selected is that, although they come in a single strip, they are individually addressable. This, along with the fact that there is an Arduino library provided by Adafruit to control this type of strips, called NeoPixel [Ada12], provides the best environment to make the output more visually appealing. Another reason for this decision is that the strips can be cut to the desired size, which makes them very useful to adapt to all parts of the vest. This feature made it possible to have only one strip on the vest, which makes the implementation easier.

4.1.4. LiPo Battery

The last component to be explained in this section is the battery. For this project, a LiPo Battery was used, as it is light and small and gives sufficient power for this project. In fact, this is a solution that is popular for wearables like this one [Dav17].

The usual voltage for this type of batteries is 3.7V, which is enough to power the lighting strips in this project, and can be stepped down to the 3.3V needed for the microcontroller and the IMU. This makes it possible to only have one battery, as well as a lineal voltage regulator, reducing both the complexity and the cost of the final product.

The only aspect that needed to be taken into account with this battery is that it needs to be handled with care, as there is risk of puncture, and some safety guidelines need to be followed [Rea18]. For this reason, the final product has a hard plastic box containing both the control subsystem and the battery.

4.2. Board Schematic

For the hardware design of this project, a Printed Circuit Board (PCB) was the option chosen. This type of board can be achieved at a reasonable price, if part of a mass production, and is also easier to diagnose in case that there is a failed component. This, along with the fact that they are very light and can be designed to be very small, make the PCB the best option for a wearable like the one addressed in this project [Aga17].

Even though the final product is aimed to include a PCB, a breakout board that includes all the components listed in section 4.1 (excluding the LED strip but including others) was used in the development stage. This made it possible to prove that the components chosen are, indeed, usable for the scope of this project. The breakout board utilized is the SEN-14001 [Ele19].

The design for the Printed Circuit Board for this project can be seen in Figure 5. For a more detailed view of this schematic, see appendix B.

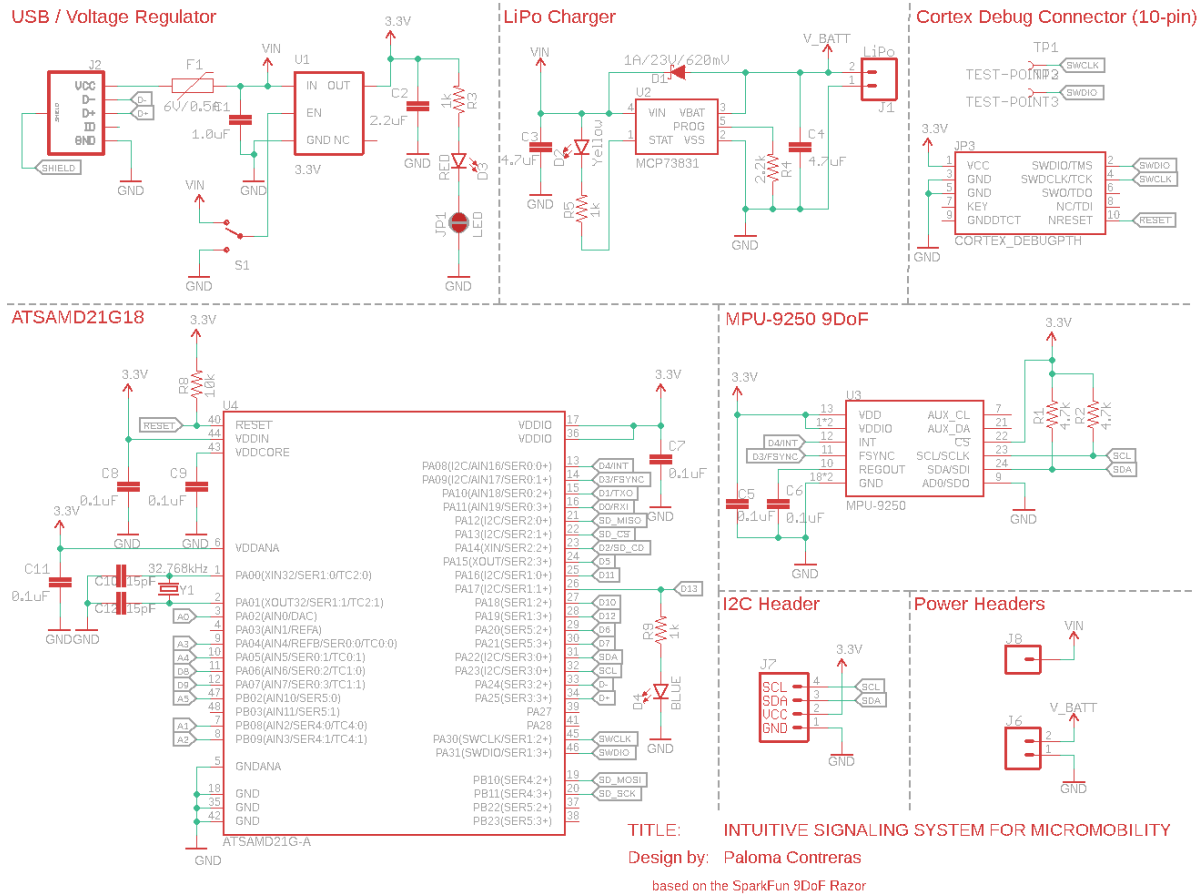


Figure 5. Board Schematic

4.3. Physical Design

In this section, system’s overall layout will be explained. As mentioned before in this document, the device is integrated into vest. This vest is intended to be a one size fits all wearable. However, if developed, adding more sizes could be an improvement, especially to gain in appearance.

The distribution of the components in the vest will be as follows.

First of all, let us consider the lighting subsystem. For this part, over the shoulders and chest, and along the upper back, is one unique NeoPixel LED strip [Ind19]. The way in which it gives all the necessary outputs is next described.

- The upper back part is used for brake signals, for which it will be turned red when necessary.

- For the turning lights, the current design makes use of the back part, as well as the upper part of the chest LEDs, which will become orange when a turn is being signaled. In order to show the direction of the turn, the lights will be repeatedly turning on and off in a sequence, as to give the sensation of being moving from one side to the other (if seen from the back), starting from the left shoulder and ending on the right shoulder when the turn is aimed to the right or vice versa.
- For the headlights, the lower part of the chest is used, which is turned bright white when the user makes the necessary input with the fob.
- As for hazard lights, the same LEDs that those of the turning lights are used, which blink repeatedly to resemble those of a car.
- Lastly, for the crash lights all the LEDs are used, as they turn a vivid orange-red, at full brightness, which blinks quickly so as to provide high visibility.

To better understand the output explained above, visit <https://courses.engr.illinois.edu/ece445/project.asp?id=7109> for videos of the prototype in the different situations.

The LED strip is, in its turn, wired to a control box on the chest of the user. This water resistant lightweight hard plastic box contains both the control and the power subsystems, as it encapsulates the PCB (or the breakout board in the case of the prototype), and the battery.

For user input, a 4 button fob is held in the hand to initiate turns, activate/deactivate hazard mode and chest lighting, and finish the crash state.

Even though the layout explained corresponds to that used in the development stage, it is subject to change if the project were to be further studied. As such, the sketch in Figure 6 could be more aesthetically pleasing than that of the prototype.

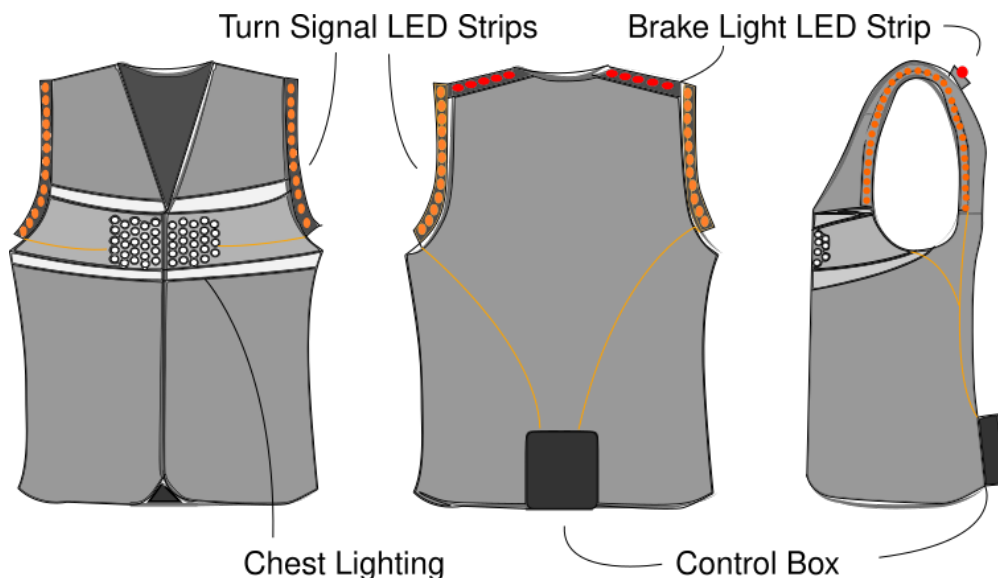


Figure 6. Physical Design

Chapter 5

Sensor Noise Reduction

THE main concern with this project is that the sensor's information needs to be accurate enough in order to ensure the correct functioning of our device. Given that we are going to use a 9DoF IMU [Inv16] (see subsection 4.1.2), consisting of an accelerometer, gyroscope, and magnetometer, we have the tools to make a fairly accurate position estimation [CLM19].

5.1. Filtering

Even though the IMU used is fairly accurate, an algorithm has been used, as to improve the functioning of the system. The purpose of this is to reduce the noise and get a better estimation on the position data.

There are many solutions available that could have been used, such as the Mahony algorithm, the Kalman filtering or the Madgwick sensor fusion algorithm. Among these three methods, this last one gets fewer error than the Kalman filter, and is also better than the Mahony algorithm for a more complex Inertial Measurement Unit (9DoF) like ours [CLM19, Mad10]. However, its implementation would add a higher level of complexity to the system, so simpler options were contemplated.

Finally, the DCM algorithm has been used. This consists of an extended Kalman filter, which is used to estimate attitude in the Direction Cosine Matrix (DCM) formation and gyroscope biases [HV15, WGP17, CLM19].

5.2. Calibration

As for calibration, an open source code [WGP17] has been used to calibrate the IMU so that it gives more accurate output data. In order to do that, the maximum and minimum values of all the angles from the gyroscope were found, as well as the different accelerations from the accelerometer. These parameters, specific to each board, were then inserted into the code, as a scale factor for the data provided from the sensors. This improves considerably the accuracy of this information, as it considers the errors that the board itself may have.

Furthermore, this calibration only needs to be made once for each board; therefore, if the vest were to be manufactured in a large scale, calibration could be made by the company, taking that problem away from the customer.

Chapter 6

Software Design

THE objective of the following sections is to explain how all the software responsible for the functioning of the system has been developed and programmed. In order to do this, a software flowchart to visually see the procedure that the software follows is included, as well as a definition of the state machine involved, with both the states definitions and transitions. To read the full code, see appendix A. This appendix includes the code that was created for this system, excluding the open source libraries that were used, which include the Razor AHRS [WGP17] for the calibration and filtering and the Adafruit Neopixel library [Ada12], among others.

6.1. Software Flowchart

First of all, to best understand the code that is explained in the following sections, an overall view of the process followed by the microcontroller can be seen in the flowchart in Figure 7. This procedure is described below.

Every process starts and ends in the idle state. Besides, throughout the whole process, the sensors are gathering information and sending it to the microcontroller. This data is then calibrated (using predefined parameters, as explained in chapter 5) and filtered. The fob also sends the user's button inputs to the microcontroller.

After gathering all this information, it is processed to follow different orders, depending on what the input is. Thus, if the user has pressed the button to signalize a turn, the turning lights are turned on and the turn detection is activated. After the software has detected that the turn has finished, the lights automatically turn off.

If the button that has been pressed is the hazard button and the lights were not activated beforehand, then they are turned on, and vice versa. This is also the process followed for the headlights.

Moreover, brake and crash detection are always being carried, and its lights are turned on automatically when either one of them is detected.

Another important factor to take into account is that whenever crash is detected, all other processes come to a halt, until hazard button is pressed and the normal behavior of the system is resumed.

These decisions explained above can be followed in detail in Figure 7. It is important to take into account that this procedure is very fast, and that the chart is meant to be representative. For this reason, some details of the process might not be exact. One such example is that the turn does not need to be finished to receive new orders from the sensors or the fob.

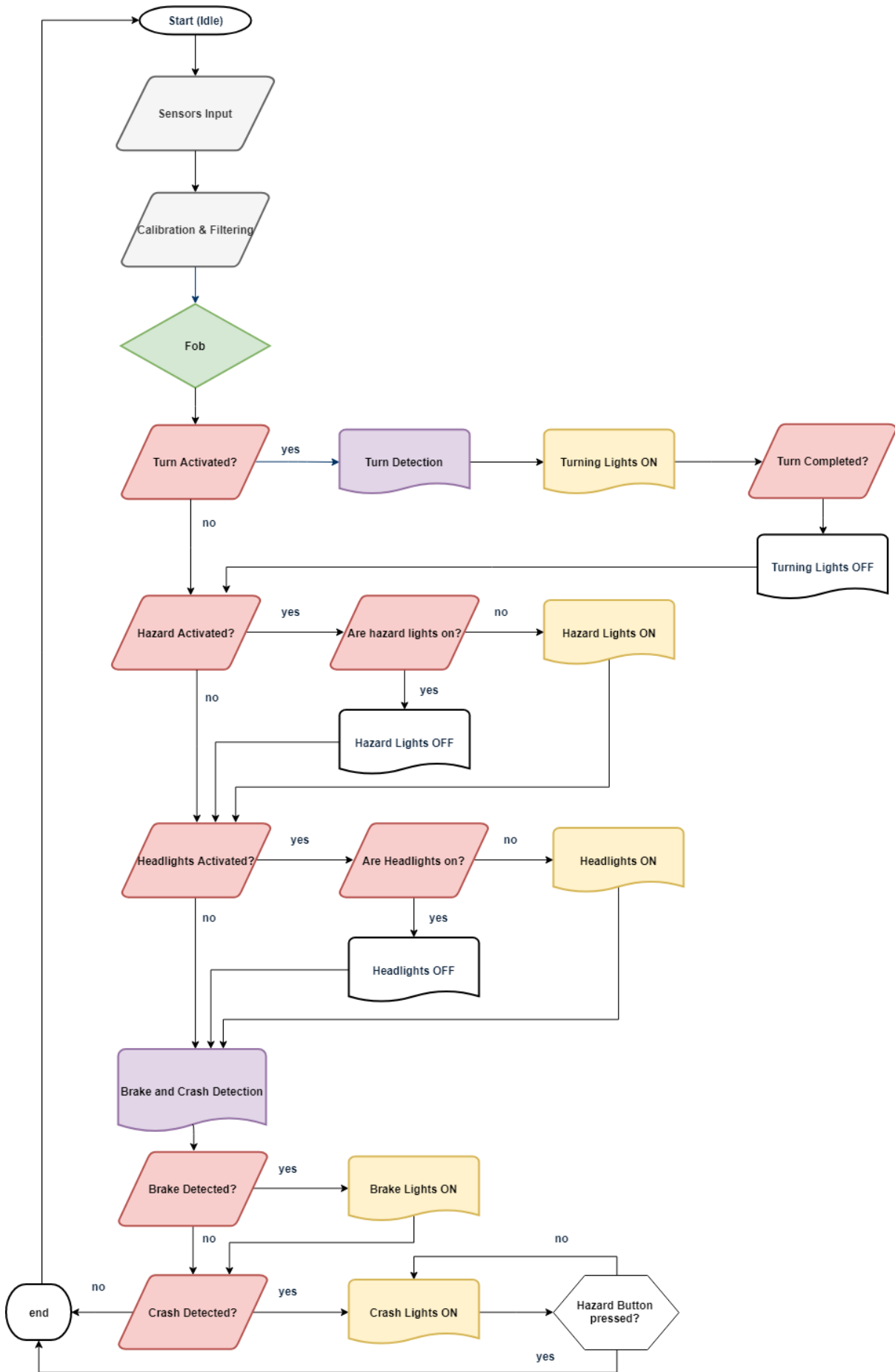


Figure 7. Software Flowchart

6.2. State Machine

The previous section was meant to give the reader an overview of the software involved in this project. However, some details were not exact, as it has been already mentioned. In order to see in greater detail how the behavior of the system has been programmed, a state machine has been included in Figure 8. The code in which the state machine is defined can be seen in subsection A.1.1.

The states included are briefly defined below, and explained in greater detail in section 6.4. The events that trigger the state transitions are also detailed in section 6.3. For the correct understanding of the state machine, the keys used to represent state transitions are included in Table 3.

- **Idle** - This is the main state the vest is in when not doing any signaling. Even though no signaling is being done, the lights are not turned off; on the contrary, they have a colorful and dim light pattern, so that to let the user know that the system is ready to receive orders.
- **Left Turn** - This state corresponds to a left turn signal. These lights are orange and give the sensation of being moving from the right to the left shoulder. After this state has started, there are two ways to transition to idle state: one of them is automatic, and happens when the turn has been considered finished; and the other is by pressing the same button one more time. This option has been included so that the user can rectify their decision.
- **Right Turn** - The Right Turn state is equivalent to the Left Turn, but with exact opposite direction for the lighting, and mirroring detection along the sagittal plane [CLM19].
- **Hazards** - Equivalent to a motor vehicle's hazard lights; they consist on blinking orange lights. This state is entered and exited manually by pressing the corresponding fob button.
- **Crash** - When a crash is detected, the vest will stop all other processes and turn all lights to a blinking orange at full brightness. This state is only exited when hazard button is pressed.

Braking was not included above because it is not a state. Braking detection is done throughout the whole process, and brake lights can be turned on at any moment (except from crash state), not interfering with the rest of the outputs.

A similar situation is that involving the headlights, which can be on at the same time as any other light, except for the case of the crash state. These lights are turned on and off by using the corresponding button from the fob, but this does not interfere in the state transition.

Transition Keys					
Key	H	R	L	Turn	Crash
Event	Hazard Button Pressed	Right Button Pressed	Left Button Pressed	Turn Detected	Crash Detected

Table 3. State Transition Keys

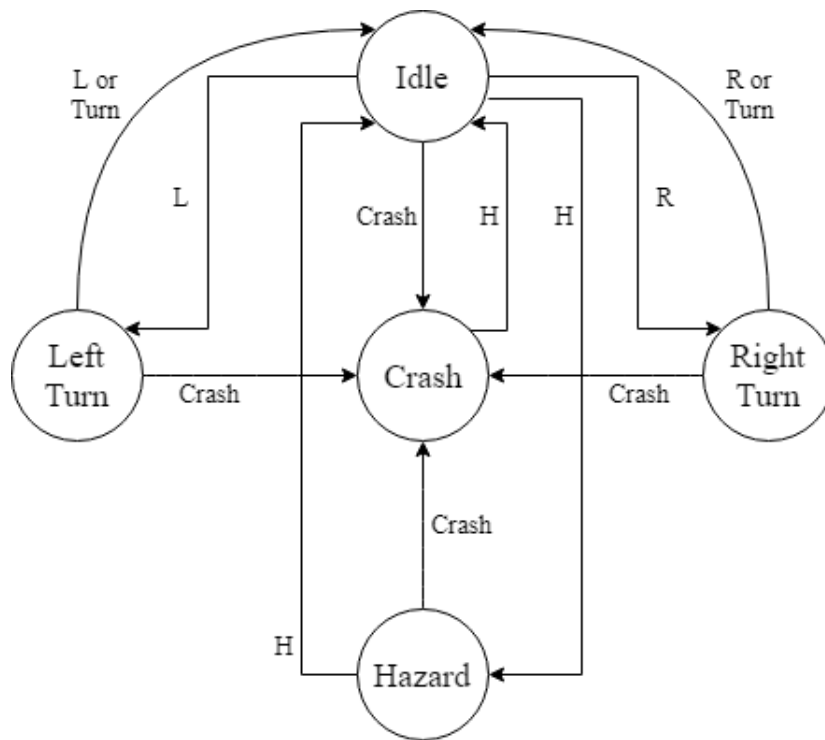


Figure 8. State Machine

6.3. State Transitions

The purpose of this section is to explain the events that need to take place for a state transition to occur. All these transitions are included in the state machine in Figure 8. Furthermore, the code in which the detection functions are generated can be seen in subsection A.2.1. These functions are then called inside the code in subsection A.2.2.

While the transitions that depend on a button being pressed are clear, the ones that get triggered by processing the data from the sensors need a further explanation, which will be given in the following subsections. These situations include turn detection, crash detection and brake detection.

6.3.1. Turn Detection

When a turn signal is used, two possible events should trigger a complete turn: a ninety-degree cardinal turn, or a lane change [CLM19]. A differentiation between the two has been made, so as to ensure that both situations are being covered by the software.

In order to detect the turn, two variables were used: the acceleration in the y-axis (lateral displacement for the user) and the yaw (angle around the vertical axis of the rider). These two values are representative for this purpose, as they change significantly during a turn, but have an almost constant behavior if the passenger is following a straight path. For that reason, a turn can be detected when each of these values have gone over a specified threshold (meaning the turn is being executed), and after that have had values below another predetermined threshold (the straight path has been retaken) [CLM19].

These thresholds mentioned above were found experimentally, during the development stage, and it was ensured that they covered both a ninety-degree turn and a lane change.

For the purpose of this project, the turn has been divided into three phases:

1. **Signal Initiated** - The turn detection starts when the user presses one of the turn buttons of the fob. At this moment, the turn state (right or left) is entered, and the turning lights are activated.
2. **Turn Made** - When the turn has been considered finished, following the procedure explained above, the turn is considered made.
3. **Turn Completed** - The turn, however, is not considered completed until the user continues going in a fairly straight path for a set amount of time, therefore ensuring that the lights do not turn off too abruptly, to improve safety. When the turn is finally completed, the system transitions to idle state.

The sketch in Figure 9 shows an example of both the lane change and the ninety-degree turn. The numbers included represent the three phases explained above.

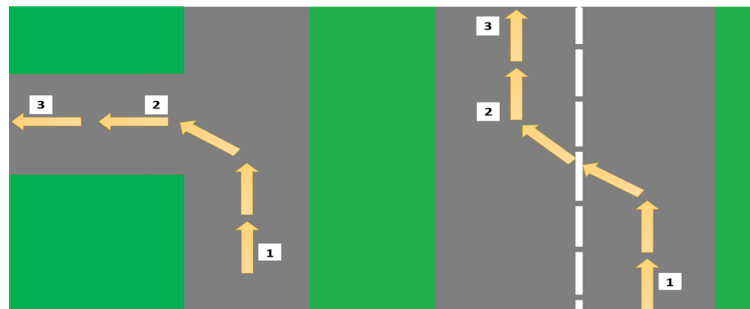


Figure 9. Turn Detection Framework

Lastly, the flowchart in Figure 10 shows in detail the turn detection process.

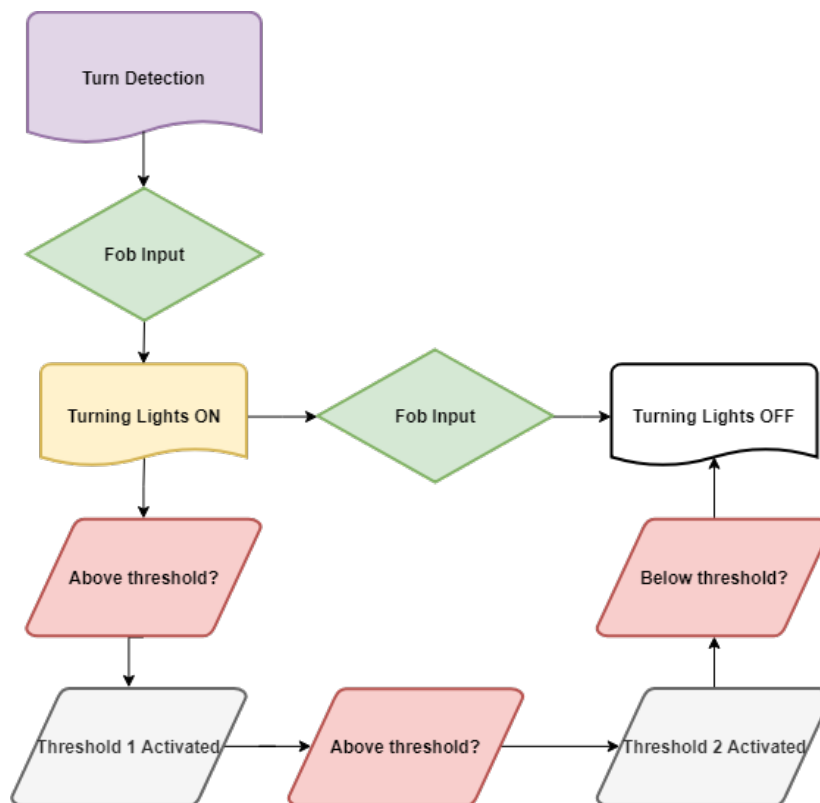


Figure 10. Turn Detection Flowchart

6.3.2. Brake Detection

The braking subroutine runs during every main loop regardless of the state the system is in, unless a crash has been detected. It constantly uses processed IMU data to determine the acceleration along the axis of motion (x-acceleration in this system) [CLM19].

For this specific detection, a derivative control is used, in order to make the acceleration in the x-axis (front direction of the rider) to be more precise and useful. When the deceleration of the rider reaches a predetermined threshold, the brake lights turn on, and their intensity is set to be proportional to the deceleration of the rider [CLM19]. The flowchart in Figure 11 shows in detail the brake detection process.

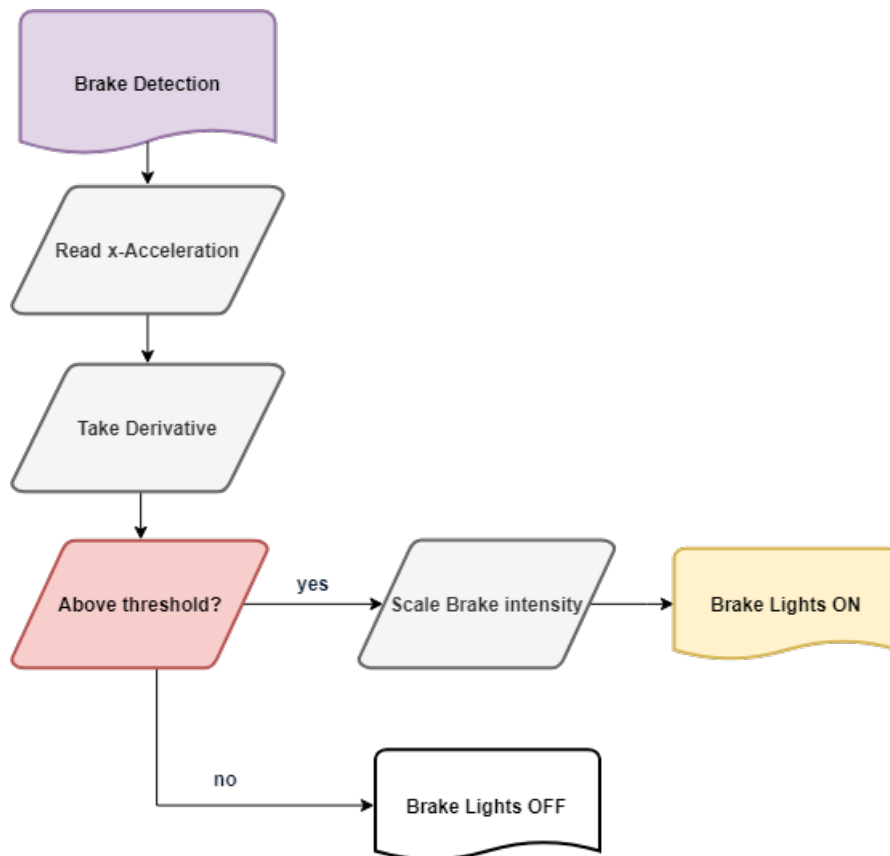


Figure 11. Brake Detection Flowchart

6.3.3. Crash Detection

In the same way as the brake detection, this subroutine is constantly being executed, using sampled IMU data to detect a crash the instant it happens. In this system, a crash is considered to be a situation of a reasonably large vertical drop, which is measured using the acceleration in the z-axis.

This acceleration has little variance during the ride, but changes suddenly during a fall, which makes it suitable for this purpose. As in previous sections, the crash is detected when this variable has overstepped a threshold, which was found experimentally. The flowchart in Figure 12 shows in detail the crash detection process.

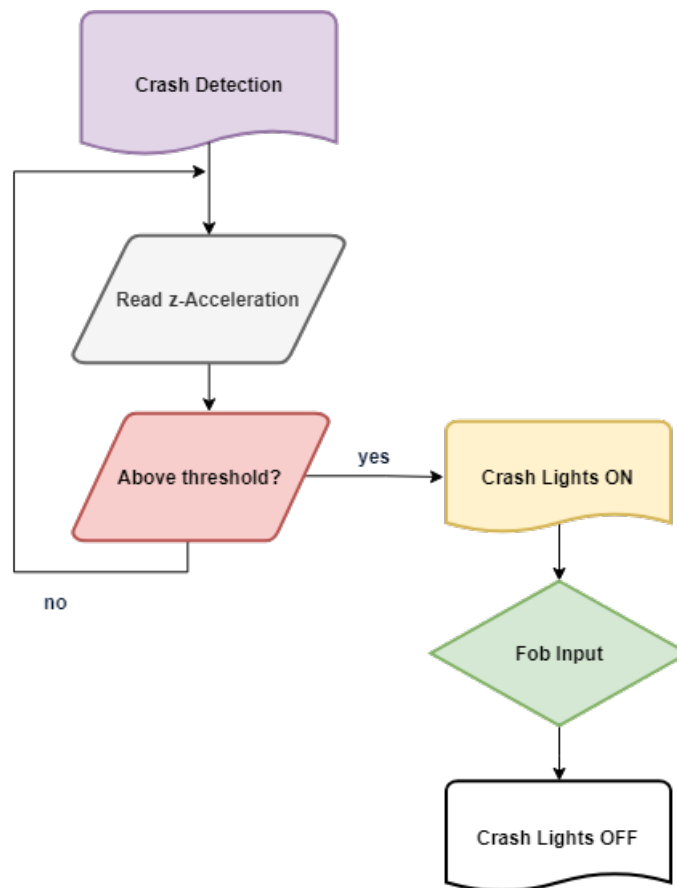


Figure 12. Crash Detection Flowchart

6.4. Lighting States

The lighting of the vest is what the user sees, the output of the system, and therefore is very important. In this section, the lighting and its code is explained. For that purpose, every state is being explained in the following subsection. To see the full code related to lighting, see subsection A.3.1 and subsection A.3.2.

6.4.1. Idle

Even though no signaling is being done during this state, it was considered useful to have some lighting so as to differentiate it from when the vest is off. Therefore, the lighting needed to be different to all other situations.

Finally, this state lighting consists of a colorful pattern, not as bright as the ones corresponding to signaling. However, it is useful to let the rider know that the vest is ready to be used, and it also gives the user better visibility, which improves safety.

The function used for this state is called `rainbowCycle`, and it can be seen in Code 1. For this function, no input is needed.

```

void rainbowCycle() {
    byte *c;
    uint16_t i;
    for(i = 0; i < NUM_LEDS; i++) {
        c = Wheel(((i * 256 / NUM_LEDS) + iter) & 255);
        setPixel(i, *(c)/20, *(c+1)/20, *(c+2)/20);
    }
    iter++;
}

```

Code 1. rainbowCycle function

6.4.2. Right and Left Turn

For both turning states, the lights consist of a bright orange, that moves from one side of the rider to the other, so as to signalize the intended direction. Therefore, the code needed for both states is the same, except for the direction of the lights.

The function used for this purpose is RunningLights. All parameters sent to this function are the same, except for the last one, which is set to false in the case of a left turn, and to true in a right turn. This function can be seen in Code 2.

```

void RunningLights(byte red, byte green, byte blue, bool is_left) {
    Position++; // = 0; //Position + Rate;
    if (is_left) {
        for(int i=0; i<NUM_LEDS; i++) {
            setPixel(i, ((sin((i+Position) % (NUM_LEDS) / 4) * 127 + 128)/255)*red,
                ((sin((i+Position) % (NUM_LEDS) / 4) * 127 + 128)/255)*green,
                ((sin((i+Position) % (NUM_LEDS) / 4) * 127 + 128)/255)*blue);
        }
    } else {
        for(int i=0; i<NUM_LEDS; i++) {
            setPixel(i, ((sin((i-Position) % (NUM_LEDS) / 4) * 127 + 128)/255)*red,
                ((sin((i-Position) % (NUM_LEDS) / 4) * 127 + 128)/255)*green,
                ((sin((i-Position) % (NUM_LEDS) / 4) * 127 + 128)/255)*blue);
        }
    }
}

```

Code 2. RunningLights function

6.4.3. Hazard

Hazard lights are similar to those of a car, consisting of orange blinking lights, as defined in the function Hazards, which can be seen in Code 3.

```

void Hazards(byte red, byte green, byte blue) {
    Position_h = Position_h+3; // = 0; //Position + Rate;
    for(int i=0; i<30; i++) {
        setPixel(i, ((sin((i+Position_h) % (NUM_LEDS) / 4) * 127 + 128)/255)*red,
            ((sin((i+Position_h) % (NUM_LEDS) / 4) * 127 + 128)/255)*green,
            ((sin((i+Position_h) % (NUM_LEDS) / 4) * 127 + 128)/255)*blue);
        setPixel(30+i, ((sin((30+i-Position_h) % (NUM_LEDS) / 4) * 127 +
            128)/255)*red,
            ((sin((30+i-Position_h) % (NUM_LEDS) / 4) * 127 + 128)/255)*green,
            ((sin((30+i-Position_h) % (NUM_LEDS) / 4) * 127 + 128)/255)*blue);
    }
}

```

Code 3. Hazards function

6.4.4. Crash

This state consists of a blinking light at full brightness, to ensure the highest visibility that is possible to achieve in the event of a crash. The code needed for this is included in Code 4.

```

// Functions used:
void setPixel(int Pixel, byte red, byte green, byte blue) {
    strip.setPixelColor(Pixel, strip.Color(red, green, blue));
}

void setAll(byte red, byte green, byte blue) {
    for(int i = 0; i < NUM_LEDS; i++) {
        setPixel(i, red, green, blue);
    }
}

// The following code is the part of the state machine that corresponds to
// a crash situation:
case STATE_CRASH:
    if (flasher) {
        setAll(255, 0, 0);
    } else {
        setAll(255, 200, 0);
    }
    flasher = !flasher;
    break;
}

```

Code 4. Crash state code

6.4.5. Brake

Braking lights are defined in a different way to those explained above. These lights are turned on when the deceleration of the rider has overstepped a predetermined threshold, by calling the function `atl_brake_switch`. When this has happened, brake lights are turned red, with a brightness that depends on the acceleration the user had when the brake threshold was activated, as it can be seen in Code 5.

```

int brake_counter = 200;
void atl_brake_switch(byte brake_on) {
    brake_counter = brake_on;
}
if (brake_counter > 0) {
    brake_counter--;
    for(int i=25; i<36; i++) {
        setPixel(i, brake_counter, 0, 0);
    }
}
}

```

Code 5. Braking lights code

6.4.6. Headlights

Headlights are also different from previous states, as it is the simplest. When the user has pressed the headlights button, the function `atl_chest_switch` is called and the lights from the front of the vest are turned white to full brightness, as seen in Code 6.

```

bool chest_active = false;
void atl_chest_switch(bool chest_on) {
    strip.setBrightness(chest_on ? 255 : 75);
    chest_active = chest_on;
}
if (chest_active) {
    for (int i=0; i < 8; i++) {
        setPixel(i, 255, 255, 255);
        setPixel(NUM_LEDS-1-i, 255, 255, 255);
    }
}
}

```

Code 6. Headlights code

Chapter 7

Project Budget

THE present chapter includes an analysis of the total cost of the final product. To do this, the bill of materials, with the prices of all the components used in this project, is included in Table 4. Furthermore, to see the total cost of the system, the labor cost is also calculated.

In the bill of materials, both the price for one unit of each device, and the price per unit if more than a hundred were bought, are included. The reason for this is that, in order to compare the vest with what is available in the market right now, it is more accurate to imagine the hypothetical case that it was manufactured in a large scale, so as to compare on an equal footing.

It is necessary to take into account that the cost of these devices may vary, as only one example was chosen from a variety of options. The distributors from which the prices have been withdrawn can be seen in the table. Only three were used: Adafruit [Ada], Digi-Key [Dig] and Mouser [Mou]. Adafruit was chosen for the NeoPixel Led strip, as it is its manufacturer; as for the other two, they were selected because they both have a Spanish site, which can save on shipping cost, in addition to being more convenient in case support is needed. Furthermore, the fact that both are big international enterprises ensured that all components would be found between these two, so as not to need too many distributors.

In addition to this, the components showed in the bill of materials in Table 4 also correspond to those in the board schematic, included in Appendix B.

These parts are the ones that would be included in the final project, not the ones used for the prototype. The biggest difference is the key fob, as wired buttons were used for the prototype, instead of a remote control with a receiver. This was enough for the prototype, and made it more achievable, but for the final system a remote is thought to be more convenient, as to not have wires going from the vest to the hand of the rider.

Furthermore, instead of building a PCB with all these components, the breakout board called SEN-14001 from SparkFun [Ele19] was used, as it had all that was needed for the development stage.

From the bill of materials, it can be calculated that the total cost of the components in the final vest is 55.221€ if only one vest was to be made, or 45.7862€ if it was fabricated in a larger scale.

Bill of Materials				
Qty	Part	Distributor	Cost Per Unit in € @ 1 Vest	Cost Per Unit in € @ 100+ Vests
Control Subsystem			20.308	17.9182
1	ATSAMD21G18	Digi-Key	2.76	2.2712
1	MPU9250	InvenSense	6.64	4.87
1	32.768kHz Crystal	Mouser	0.508	0.377
1	Simple RF M4 Receiver - 315MHz Momentary Type	Mouser	4.32	4.32
1	Keyfob 4 Button RF Remote Control 315MHz	Digi-Key	6.08	6.08
Lighting Subsystem			22.11	17.69
1	Neopixel Weatherproof LED Strip	Adafruit	22.11	17.69
Power Subsystem			12.803	10.178
1	Switch SPTST	Mouser	0.245	0.187
1	USB Micro SMD Connector	Mouser	0.638	0.481
2	15pF Capacitor	Mouser	0.07	0.041
6	0.1uF Capacitor	Mouser	0.02	0.013
1	1uF Capacitor	Mouser	0.087	0.026
1	2.2uF Capacitor	Mouser	0.087	0.02
2	4.7uF Capacitor	Mouser	0.105	0.028
1	BAT20J Schottky Diode 1A/23V/620mV	Mouser	0.332	0.134
3	1k Resistor	Mouser	0.087	0.02
1	2.2k Resistor	Mouser	0.087	0.003
2	4.7k Resistor	Mouser	0.087	0.007
1	10k Resistor	Mouser	0.087	0.044
1	3.3V Voltage Regulator	Mouser	0.515	0.276
1	MCP73831	Mouser	0.498	0.377
1	1206 6V/0.5A Resettable Fuse	Mouser	0.288	0.081
1	SMD LED 0603 Blue	Mouser	0.358	0.177
1	SMD LED 0603 Red	Mouser	0.603	0.306
1	SMD LED 0603 Yellow	Mouser	0.603	0.306
1	JST 2-pin SMD LiPo Connector	Mouser	0.83	0.83
1	JST 3.7V-1000mah LiPo Battery	Mouser	6.64	6.64

Table 4. Bill of Materials

In order to fully calculate the cost of the vest, the labor cost needs to be included. In order to do this, the company Iberdrola was taken as an example to calculate the operator's salary. It is estimated that a worker would need an hour to fully mount the vest, as the design is already done. An estimated salary for this type of worker is 20000€, and Iberdrola's operators work approximately 1680 hours a year [Ibe17]. This is thought to be a fair salary and a good estimation for the labor cost for a project like this one. This means that the labor cost is 11.905€ per hour, so this is the cost that would need to be added to each vest.

Taking both the material prices and the labor cost into account, the total cost of the vest if only one unit was to be made is 67.126€, while if it were manufactured in a larger scale, the price would be 59.6912€. This is consistent with what is currently available on the market, making this product a competitive solution.

Chapter 8

Conclusions

As it has been explained throughout the whole document, a better method for signaling is possible, and it has been, in fact, achieved. This chapter outlines what has been accomplished in this project, as well as improvements that could add enhancements to the product.

8.1. Conclusions on Results

In the state of art, it was concluded that there was room for improvement in this field. There are many options in the market for signaling systems, but they are archaic and non-intelligent. Thus, this idea outperforms its competitors on the market.

The vest proposed in this project has sufficient lighting to make a rider visible at night, complying with current regulations, and also adds visibility to the user during daylight. This increases safety for all the people who share a space with the rider. Furthermore, it makes the signage more convenient for both the person wearing the vest and others on the road, as it includes lighting signs similar to those of cars, which are easily understandable.

Moreover, a new feature has been included, which also improves safety. This is the case of the crash lights, that can notify others on the road of a fall, or can help find the rider if it has happened on a poorly trafficked area.

This system is competitive to what there is on the market at the moment, as it provides innovative features but does not add to the price, as it has been proven from the research in chapter 2 and the cost analysis in chapter 7.

Lastly, the fact that all these features are included in one piece, which does not need to be mounted on the vehicle but only be worn by the user, makes it more versatile and accessible for everyone. Furthermore, its price is even more competitive because of this, due to the fact that no other lights or signals would be necessary.

8.2. Conclusions on Methodology

This project was divided into three different subsystems, in order to be able to work separately on each of the parts. During the development stage, it was possible to work on them individually, and to carry tests in each of them, so as to easily identify possible errors. This led to a faster and more efficient development.

Having done this, the three parts were then joined to build the prototype, and experiments were carried so as to find the needed thresholds for the detection processes.

This methodology facilitated finding errors easier, as the subsystems were small enough to be easily testable. Furthermore, having a prototype made it possible to test everything that was being implemented, and therefore improve the model.

8.3. Future Improvements

Although many aspects of this field have been overcome, there is still room for improvement, some of which will be explained in the upcoming paragraphs.

First of all, one feature of the vest that has been proposed but not implemented yet is the remote control. This is part of the model, but it was not included in the prototype, and it will definitely improve the product.

In addition to this, the lighting patterns could be studied, as to provide the best experience to the user. Specifically, headlights could be improved and converted into directed lights.

Similarly, the thresholds would need a further research, so as to provide the best experience to the user. The system has been tested in one single environment, so more situations would need to be researched, especially in the case of the crash detection, as it might be triggered too often if used on extreme environments such as those that contain terrain, like desert and rocky mountains. Further experimental data and sensor adjustment will be needed for a more accurate behavior.

However, the most important aspect to be improved is its effectiveness to all modes of transport. Right now, the model depends on the axis of acceleration. Additionally, as it is a wearable, if the direction of motion does not correspond to the forward displacement of the user (as is the case with a skateboard), the software would not be as accurate. This affects both the turn and the brake detection. For the first of these two, it has been proven that the model works, as the turn detection depends on both the yaw (which is still useful, as it corresponds to the vertical axis of the rider) and the y-axis, which would be useless in this case, as it corresponds to frontal instead of lateral displacement. As for the brake detection, the axis that is used to find the deceleration of the rider (x-axis) is wrong, so it would be defective.

One way to solve this previous problem is to add a button on the remote control that changes the settings from side-scrolling to front-scrolling. This would change the settings in the software, to exchange what the x-axis and y-axis represent.

Other improvements could be phone connectivity, which could improve both safety and quality of life. Moreover, audio feedback could be added to make the system as a whole even more intuitive [CLM19]. This would also improve safety, as in the current design the user needs to look down at the vest to see what lights are on. In addition to this, a horn could be added to further improve the product.

Lastly, as it is a wearable, beauty is also important in order to make the product more appealing to the public. Furthermore, right now it consists of a one-size model, but this could be changed to better fit the user, having different sizes. If it were to be sold, this aspect would need to be worked on.

Bibliography

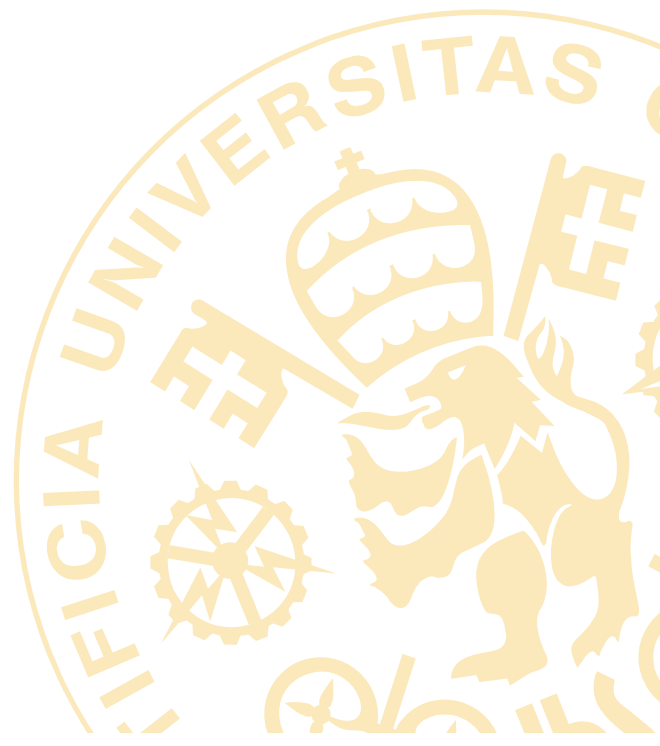
- [ABC18] ABC, “El 7,5% de los accidentes en España tiene a un ciclista involucrado,” Jun. 2018, accessed 30/06/2019. [Online]. Available: https://www.abc.es/motor/reportajes/abci-75-por-ciento-accidentes-espana-tiene-ciclista-involucrado-201806190611_noticia.html
- [Ada] Adafruit, “Adafruit Industries, Unique & fun DIY electronics and kits,” accessed 04/07/2019. [Online]. Available: <https://www.adafruit.com/>
- [Ada12] —, “Arduino library for controlling single-wire LED pixels (NeoPixel, WS2812, etc.): adafruit/Adafruit_neopixel,” 2012, accessed 02/07/2019. [Online]. Available: https://github.com/adafruit/Adafruit_NeoPixel
- [Aga17] T. Agarwal, “What Are The Advantages Of Using A Printed Circuit Board (PCB),” Aug. 2017, accessed 02/07/2019. [Online]. Available: <https://www.edgefx.in/advantages-using-printed-circuit-board-pcb/>
- [All18] M. Alliance, “How Micro Mobility Solves Multiple Problems in Congested Cities,” Jul. 2018, accessed 12/06/2019. [Online]. Available: <https://maas-alliance.eu/how-micro-mobility-solves-multiple-problems-in-congested-cities/>
- [Ama19a] Amazon, “Cateye Kinetic X2 Rear: by Cateye: Amazon.es: Deportes y aire libre,” 2019, accessed 30/06/2019. [Online]. Available: https://www.amazon.es/Cateye-Kinetic-X2-Rear/dp/B01KXIX4CE/ref=as_li_ss_tl?ie=UTF8&linkCode=s11&tag=prema0a-21&linkId=4240a9eec5008d01d93147fa988224d2&language=es_ES
- [Ama19b] —, “LED Light Vest,” 2019, accessed 26/06/2019. [Online]. Available: https://www.amazon.es/s?k=LED+Light+Vest&__mk_es_ES=%C3%85M%C3%85%C5%BD%C3%95%C3%91&ref=nb_sb_noss
- [Arr17] R. M. Arroyo, “Luz trasera inteligente Cateye Rapid X2 Kinetic & Foco delantero Cateye Volt 1600 para bicicletas,” Oct. 2017, accessed 30/06/2019. [Online]. Available: <http://www.iberobike.com/luz-trasera-inteligente-cateye-rapid-x2-kinetic-foco-delantero-cateye-volt-1600-para-bicicletas/>
- [Atm15] Atmel, “SMART SAM D21 Datasheet,” Sep. 2015, accessed 02/07/2019. [Online]. Available: https://cdn.sparkfun.com/datasheets/Dev/Arduino/Boards/Atmel-42181-SAM-D21_Datasheet.pdf
- [BOE14] BOE, “Reglamento General de Vehículos - Actualizado 2014,” 2014, accessed 28/06/2019. [Online]. Available: <https://www.boe.es/buscar/act.php?id=BOE-A-1999-1826&b=53&tn=1&p=20140522#a22>
- [CLM19] P. Contreras, W. Lee, and D. Miller, “SignalMe,” Apr. 2019, accessed 02/07/2019. [Online]. Available: <https://courses.engr.illinois.edu/ece445/project.asp?id=7109>
- [Dav17] N. Davis, “Powering wearables: battery types, current challenges, and energy harvesting,” Jul. 2017, accessed 27/06/2019. [Online]. Available: <https://www.powerelectronicsnews.com/technology/powering-wearables-battery-types-current-challenges-and-energy-harvesting>
- [DGT16a] DGT, “Guía para usuarios de la bicicleta,” 2016, accessed 12/06/2019. [Online]. Available: <http://www.dgt.es/Galerias/seguridad-vial/educacion-vial/recursos-didacticos/jovenes/Guia-Bicicleta-agosto-2016.pdf>

- [DGT16b] —, “Vehículos de Movilidad Personal (VMP),” 2016, accessed 30/06/2019. [Online]. Available: http://www.dgt.es/Galerias/seguridad-vial/normativa-legislacion/otras-normas/modificaciones/2016/Instr_16_V_124_Vehiculos_Movilidad_Personal.pdf
- [DGT19a] —, “Las nuevas formas de moverse por la ciudad,” 2019, accessed 30/06/2019. [Online]. Available: <http://revista.dgt.es/es/noticias/nacional/2019/02FEBRERO/0211Uso-patinetes-en-ciudad.shtml>
- [DGT19b] —, “Real Decreto XXXX/2019 de medidas urbanas de tráfico.” 2019, accessed 30/06/2019. [Online]. Available: http://www.interior.gob.es/documents/642012/9796404/02_2019_Medidas_urbanas_de_trafico.pdf/dbde2652-6dce-4465-b260-767fae80be72
- [Dig] DigiKey, “DigiKey Electronics España,” accessed 04/07/2019. [Online]. Available: <https://www.digikey.es/>
- [Ele19] S. Electronics, “SparkFun 9dof Razor IMU M0 - SEN-14001,” 2019, accessed 12/06/2019. [Online]. Available: <https://www.sparkfun.com/products/14001>
- [HV15] H. Hyyti and A. Visala, “A DCM Based Attitude Estimation Algorithm for Low-Cost MEMS IMUs,” 2015, accessed 27/06/2019. [Online]. Available: <https://www.hindawi.com/journals/ijno/2015/503814/>
- [Ibe17] Iberdrola, “VII CONVENIO COLECTIVO DE IBERDROLA GRUPO,” p. 124, 2017, accessed 05/07/2019.
- [Ind19] A. Industries, “Adafruit NeoPixel LED Side Light Strip - Black 60 LED,” 2019, accessed 12/06/2019. [Online]. Available: <https://www.adafruit.com/product/3636>
- [Inv16] InvenSense, “MPU-9250 Product Specification Revision 1.1,” Jun. 2016, accessed 27/06/2019. [Online]. Available: <http://www.invensense.com/wp-content/uploads/2015/02/PS-MPU-9250A-01-v1.1.pdf>
- [Mad10] S. O. Madgwick, “An efficient orientation filter for inertial and inertial/magnetic sensor arrays,” Apr. 2010, accessed 27/06/2019.
- [Mou] Mouser, “Mouser Electronics Spain - Electronic Components Distributor,” accessed 04/07/2019. [Online]. Available: <https://www.mouser.es/>
- [noa19a] “10 Mejores Luces Intermitentes Bicicleta 2019,” 2019, accessed 30/06/2019. [Online]. Available: <https://bicicletaselectricas.xyz/luces-intermitentes-para-bicicleta/>
- [noa19b] “The Micromobility Conference, California 2019,” 2019, accessed 28/06/2019. [Online]. Available: <http://movmi.net/micromobility-conference-2019/>
- [Pas18] J. Pastor, “Preocupación sobre ruedas: 273 accidentes con patinetes eléctricos en España en 2018, la mayoría provocados por los patinadores,” Dec. 2018, accessed 30/06/2019. [Online]. Available: https://www.xataka.com/vehiculos/preocupacion-ruedas-273-accidentes-patinetes-electricos-espana-2018-mayoria-provocados_patinadores
- [RAC19] RACE, “¿Cómo circular con patinetes eléctricos? Normativa,” Mar. 2019, accessed 30/06/2019. [Online]. Available: <https://www.race.es/patinete-electrico-legislacion>
- [Rea18] J. Reade, “LiPo Batteries and Safety for Beginners,” Jan. 2018, accessed 27/06/2019. [Online]. Available: <https://www.cnydrones.org/lipo-batteries-and-safety-for-beginners/>
- [Rem18] R. Remick, “The 7 Best Bike Lights Reviewed & Compared For 2019,” Apr. 2018, accessed 30/06/2019. [Online]. Available: <https://www.outsidepursuits.com/best-bike-headlight-reviews/>

- [Tri19] E. U. Triatleta, “Mejores luces traseras para bicicleta 2018 | Todo el mundo te verá,” Jan. 2019, accessed 30/06/2019. [Online]. Available: <https://elultimotriatleta.com/mejores-luces-traseras-para-bicicleta/>
- [Vic16] Victor-Premarathon, “El valor de las ideas: luz trasera (de freno) para bicicletas.” Feb. 2016, accessed 30/06/2019. [Online]. Available: <https://www.premarathon.com/el-valor-de-las-ideas-luz-freno-bici/>
- [WGP17] J. Wang, M. Geier, and B. Peter, “Razor AHRS,” Dec. 2017, accessed 27/06/2019. [Online]. Available: <https://github.com/Razor-AHRS>

PART II

APPENDICES



Appendix A



Complete Code



A.1. Main Program - State Machine

A.1.1. main.cpp

```

#include <detection.hpp>
#include <lighting.hpp>
#include <Arduino.h>
#include <JC_Button.h>
#include <Razor_AHRS.hpp>
#include <SD_Logger.hpp>

void logIMUData();
void setup_buttons();
void button_update();
void update_state();

ToggleButton chest_button(10);
ToggleButton left_button(11);
ToggleButton right_button(12);
ToggleButton hazard_button(13);

int state_var = 0;
bool chest_on = false;
unsigned long IMU_timestamp;

void setup() {
  SerialPort.begin(115200);
  setup_buttons();
  setup_AHRS();
  setup_detection();
  setup_SDcard();
  atlighty_init();

  IMU_timestamp = millis();
}

void loop() {
  AHRS_cmd();
  if ((millis() - IMU_timestamp) >= 20) {
    IMU_timestamp = AHRS_update();
    update_detection();
    atl_light_state(state_var);
    logIMUData();
  }
  update_state();
}

void setup_buttons() {
  left_button.begin();
  right_button.begin();
  hazard_button.begin();
  chest_button.begin();
}

void button_update() {
  left_button.read();
  right_button.read();
  hazard_button.read();
  chest_button.read();
}

void update_state() {
  button_update();

  // STATE MACHINE
  switch (state_var) {
    case STATE_IDLE:
      reset_turn();
      if (left_button.changed()) {
        state_var = STATE_LEFT;
        SerialPort.println("State: LEFT");
      } else if (right_button.changed()) {
        state_var = STATE_RIGHT;
        SerialPort.println("State: RIGHT");
      }
  }
}

```

```

    } else if (hazard_button.changed()) {
        state_var = STATE_HAZARD;
        SerialPort.println("State: HAZARD");
    } else {
        brake_detect();
        crash_detect();
    }
    break;

case STATE_LEFT:
    if (left_button.changed() || turn_complete) {
        state_var = STATE_IDLE;
        reset_turn();
        SerialPort.println("State: IDLE");
    } else {
        brake_detect();
        crash_detect();
        turn_detect();
    }
    break;

case STATE_RIGHT:
    if (right_button.changed() || turn_complete) {
        state_var = STATE_IDLE;
        reset_turn();
        SerialPort.println("State: IDLE");
    } else {
        brake_detect();
        crash_detect();
        turn_detect();
    }
    break;

case STATE_CRASH:
    if (hazard_button.changed()) {
        state_var = STATE_IDLE;
        SerialPort.println("State: IDLE");
    }
    break;

case STATE_HAZARD:
    if (hazard_button.changed()) {
        state_var = STATE_IDLE;
        SerialPort.println("State: IDLE");
    } else {
        brake_detect();
        crash_detect();
    }
    break;
}

if (crash_detected && (state_var != STATE_CRASH)) {
    state_var = STATE_CRASH;
    SerialPort.println("State: CRASH");
    crash_detected = false;
}

atl_brake_switch(byte(brake_detection_value / 4.0) * 255);

if (chest_button.changed()) {
    chest_on = !chest_on;
    atl_chest_switch(chest_on);
    SerialPort.println(chest_on ? "Headlight On" : "Headlight Off");
}
}

void logIMUData() {
    String imuLog = "";
    switch (state_var) {
    case STATE_IDLE:
        imuLog += "Idle " + String(millis()) + "\n";
        break;

    case STATE_LEFT:
        imuLog += "Left " + String(millis()) + "\n";

```

```

imuLog += "yaw_difference = " + String(yaw_detection_value) + ", ";
imuLog += "acc_difference = " + String(accel_detection_value) + ", ";
break;

case STATE_RIGHT:
  imuLog += "Right " + String(millis()) + "\n";
  imuLog += "yaw_difference = " + String(yaw_detection_value) + ", ";
  imuLog += "acc_difference = " + String(accel_detection_value) + ", ";
  break;

case STATE_CRASH:
  imuLog += "Crash " + String(millis()) + "\n";
  break;
}

imuLog += "brake_detection_value = " + String(brake_detection_value) + ", ";
imuLog += "crash_detection_value = " + String(crash_detection_value) + "\n";

// If SD card logging is enabled & a card is plugged in (only useful for prototype)
if (sdCardPresent) {
  // If adding this log line will put us over the buffer length:
  if (imuLog.length() + logFileBuffer.length() >= 1024) {
    sdLogString(logFileBuffer); // Log SD buffer
    logFileBuffer = ""; // Clear SD log buffer
  }
  // Add new line to SD log buffer
  logFileBuffer += imuLog;
}
}
}

```

A.2. Turn, Crash and Brake Detection

A.2.1. detection.cpp

```

#include "lighting.hpp"
#include "Razor_AHRS.hpp"
#include "Ticker.h"
#include "detection.hpp"

int count_accel_sum = 0;
float accel_sum = 0.0;

float yaw_vector[100];
float accel_x_vector[100];
float accel_y_vector[100];
float accel_z_vector[100];
bool turn_detection = false;
bool threshold_yaw = false;
bool threshold_accel = false;
int count = 0; // Counter to NOT get acceleration data in every loop
int dummy_counter = 0; // To delete the first wrong acceleration data

float brake_detection_value = 0.0;
float crash_detection_value = 0.0;
float yaw_detection_value = 0.0;
float accel_detection_value = 0.0;

bool crash_detected = false;
bool brake_detected = false;
bool turn_complete = false;

void crash_detect() {
  float z_threshold = 12.0;
  float current_z_accel = accel_z_vector[99];
  float previous_z_accel = accel_z_vector[0];
  float z_difference = previous_z_accel - current_z_accel;
  float z_interval = 20 * 100;
  float z_result = z_difference / z_interval;
  crash_detection_value = z_result;
  crash_detected = (z_result >= z_threshold) ? true : crash_detected;
}

```

```

void brake_detect() {
    float threshold = -1;
    float current_accel = accel_x_vector[99];
    float previous_accel = accel_x_vector[0];
    float difference = current_accel - previous_accel ;
    float interval = 20 * 100;
    float result = difference / interval;
    brake_detection_value = result;
    brake_detected = (result >= threshold) ? true : brake_detected;
}

void turn_detect() {
    float acc_difference = abs(accel_y_vector[0] - accel_y_vector[99]);
    float yaw_difference = abs(yaw_vector[0] - yaw_vector[99]) * 100;
    yaw_detection_value = yaw_difference;

    count_accel_sum += 1;
    if (count_accel_sum < 5)
    {
        accel_sum += acc_difference;
    }else{
        accel_detection_value = accel_sum/5;
        count_accel_sum = 0;
        accel_sum = 0;
    }

    /*
    SerialPort.print(accel_detection_value);
    SerialPort.print(", ");
    SerialPort.println(yaw_difference);
    */

    if (accel_detection_value > 1000 && !threshold_accel) {
        threshold_accel = true;
        //SerialPort.println("Turn: Acceleration Threshold Reached");
    }

    if (yaw_difference > 60 && !threshold_yaw) {
        threshold_yaw = true;
        //SerialPort.println("Turn: Yaw Threshold Reached");
    }

    if (threshold_yaw && threshold_accel && !turn_detection) {
        //SerialPort.println("Turn: Detected");
        turn_detection = true;
    }

    if (turn_detection && acc_difference < 400 and yaw_difference < 15 &&
        !turn_complete) {
        //SerialPort.println("Turn: Completed");
        turn_complete = true;
    }
}

void reset_turn() {
    threshold_accel = false;
    threshold_yaw = false;
    turn_detection = false;
    turn_complete = false;
    for (int i = 0; i < 100; i++) {
        accel_y_vector[i] = 0.0;
        yaw_vector[i] = 0.0;
    }
}

void setup_detection() {
    // Initialization of acceleration and yaw vectors
    for (int i = 0; i < 100; i++) {
        accel_x_vector[i] = 0.0;
        accel_y_vector[i] = 0.0;
        accel_z_vector[i] = 0.0;
        yaw_vector[i] = 0.0;
    }
}

```



```

void update_detection() {
  // Update Acceleration Vector
  for (int i = 0; i < 99; i++) {
    accel_z_vector[i] = accel_z_vector[i + 1];
    accel_y_vector[i] = accel_y_vector[i + 1];
    accel_x_vector[i] = accel_x_vector[i + 1];
  }
  accel_x_vector[99] = accel[0];
  accel_y_vector[99] = accel[1];
  accel_z_vector[99] = accel[2];

  //Update Yaw Vector
  for (int i = 0; i < 99; i++) { yaw_vector[i] = yaw_vector[i + 1]; }
  yaw_vector[99] = (yaw > 3.14) ? yaw : (yaw = 6.28 - yaw);
}

```

A.2.2. detection.hpp

```

#ifndef DETECTION_HPP
#define DETECTION_HPP

#include <Arduino.h>
#include "Ticker.h"

extern float acce[3];
extern float yaw;
extern float brake_detection_value;
extern float crash_detection_value;
extern float yaw_detection_value;
extern float accel_detection_value;
extern bool crash_detected;
extern bool brake_detected;
extern bool turn_complete;

void setup_detection();
void update_detection();

void crash_detect();

void brake_detect();
void release_brake();

void turn_detect();
void reset_turn();

#endif

```

A.3. Libraries for Lighting Control

A.3.1. lighting.cpp

```

#include <lighting.hpp>
#include <Arduino.h>
#include <Wire.h>

#include <Adafruit_NeoPixel.h>
Adafruit_NeoPixel strip = Adafruit_NeoPixel(NUM_LEDS, LED_STRIP_PIN, NEO_GRB + NEO_KHZ800);

void showStrip() {
  strip.show();
}

void setPixel(int Pixel, byte red, byte green, byte blue) {
  strip.setPixelColor(Pixel, strip.Color(red, green, blue));
}

void setAll(byte red, byte green, byte blue) {
  for(int i = 0; i < NUM_LEDS; i++ ) {
    setPixel(i, red, green, blue);
  }
}

```

```

byte * Wheel(byte WheelPos) {
    static byte c[3];
    if(WheelPos < 85) {
        c[0]=WheelPos * 3;
        c[1]=255 - WheelPos * 3;
        c[2]=0;
    } else if(WheelPos < 170) {
        WheelPos -= 85;
        c[0]=255 - WheelPos * 3;
        c[1]=0;
        c[2]=WheelPos * 3;
    } else {
        WheelPos -= 170;
        c[0]=0;
        c[1]=WheelPos * 3;
        c[2]=255 - WheelPos * 3;
    }
    return c;
}

int maximum = 0;
int iter = 0;

void rainbowCycle() {
    byte *c;
    uint16_t i;
    for(i = 0; i < NUM_LEDS; i++) {
        c = Wheel((i * 256 / NUM_LEDS) + iter) & 255;
        setPixel(i, *(c)/20, *(c+1)/20, *(c+2)/20);
    }
    iter++;
}

int Position=0;
void RunningLights(byte red, byte green, byte blue, bool is_left) {
    Position++; // = 0; //Position + Rate;
    if (is_left) {
        for(int i=0; i<NUM_LEDS; i++) {
            // sine wave, 3 offset waves make a rainbow!
            //float level = sin(i+Position) * 127 + 128;
            //setPixel(i,level,0,0);
            //float level = sin(i+Position) * 127 + 128;
            setPixel(i, ((sin((i+Position) % (NUM_LEDS) / 4) * 127 + 128)/255)*red,
                ((sin((i+Position) % (NUM_LEDS) / 4) * 127 + 128)/255)*green,
                ((sin((i+Position) % (NUM_LEDS) / 4) * 127 + 128)/255)*blue);
        }
    } else {
        for(int i=0; i<NUM_LEDS; i++) {
            // sine wave, 3 offset waves make a rainbow!
            //float level = sin(i+Position) * 127 + 128;
            //setPixel(i,level,0,0);
            //float level = sin(i+Position) * 127 + 128;
            setPixel(i, ((sin((i-Position) % (NUM_LEDS) / 4) * 127 + 128)/255)*red,
                ((sin((i-Position) % (NUM_LEDS) / 4) * 127 + 128)/255)*green,
                ((sin((i-Position) % (NUM_LEDS) / 4) * 127 + 128)/255)*blue);
        }
    }
}

int Position_h = 0;
void Hazards(byte red, byte green, byte blue) {
    Position_h = Position_h+3; // = 0; //Position + Rate;
    for(int i=0; i<30; i++) {
        // sine wave, 3 offset waves make a rainbow!
        //float level = sin(i+Position) * 127 + 128;
        //setPixel(i,level,0,0);
        //float level = sin(i+Position) * 127 + 128;
        setPixel(i, ((sin((i+Position_h) % (NUM_LEDS) / 4) * 127 + 128)/255)*red,
            ((sin((i+Position_h) % (NUM_LEDS) / 4) * 127 + 128)/255)*green,
            ((sin((i+Position_h) % (NUM_LEDS) / 4) * 127 + 128)/255)*blue);
        // sine wave, 3 offset waves make a rainbow!
        //float level = sin(i+Position) * 127 + 128;
        //setPixel(i,level,0,0);
        //float level = sin(i+Position) * 127 + 128;
    }
}

```

```

        setPixel(30+i, ((sin((30+i-Position_h) % (NUM_LEDS) / 4) * 127 + 128)/255)*red,
                ((sin((30+i-Position_h) % (NUM_LEDS) / 4) * 127 + 128)/255)*green,
                ((sin((30+i-Position_h) % (NUM_LEDS) / 4) * 127 + 128)/255)*blue);
    }
}

unsigned l_timestamp;
void atlighty_init() {
    strip.begin(); // initialize the strip
    strip.setBrightness(75);
    setAll(255, 0, 0);
    strip.show();
}

bool chest_active = false;
void atl_chest_switch(bool chest_on) {
    strip.setBrightness(chest_on ? 255 : 75);
    chest_active = chest_on;
}

int brake_counter = 200;
void atl_brake_switch(byte brake_on) {
    brake_counter = brake_on;
}

bool flasher = false;
void atl_light_state(byte state) {
    switch (state) {
        case STATE_IDLE:
            rainbowCycle();
            break;
        case STATE_LEFT:
            RunningLights(150, 100, 0, false);
            break;
        case STATE_RIGHT:
            RunningLights(150, 100, 0, true);
            break;
        case STATE_HAZARD:
            Hazards(250, 20, 0);
            break;
        case STATE_CRASH:
            if (flasher) {
                setAll(255, 0, 0);
            } else {
                setAll(255, 200, 0);
            }
            flasher = !flasher;
            break;
        default:
            rainbowCycle();
    }
}

if (brake_counter > 0) {
    brake_counter--;
    for(int i=25; i<36; i++) {
        setPixel(i, brake_counter, 0, 0);
    }
}

if (chest_active) {
    for (int i=0; i < 8; i++) {
        setPixel(i, 255, 255, 255);
        setPixel(NUM_LEDS-1-i, 255, 255, 255);
    }
}
showStrip();
}

```

A.3.2. lighting.hpp

```

#ifndef HEADER_LIGHT_REGS
#define HEADER_LIGHT_REGS

```

```

#include <Arduino.h>

#define ATLIGHTY_I2C 0x13

#define LED_STRIP_PIN 9
#define NUM_LEDS 60

// Base Addresses
#define LEFT 0x00
#define RIGHT 0x08
// +
#define FRONT 0x00
#define BACK 0x10
#define SHOULDER 0x20
#define CHEST 0x30
// +
#define COLOR_1_RED 0x00
#define COLOR_1_GREEN 0x01
#define COLOR_1_BLUE 0x02
#define COLOR_2_RED 0x03
#define COLOR_2_GREEN 0x04
#define COLOR_2_BLUE 0x05
#define PATTERN 0x06
#define PARAMETER 0x07

// Pattern Types
#define MODE_OFF 0x00
#define MODE_STEADY 0x01
#define MODE_BLINK 0x02
#define MODE_ALT 0x03

// Lighting States
#define STATE_IDLE 0x00
#define STATE_LEFT 0x01
#define STATE_RIGHT 0x02
#define STATE_CRASH 0x03
#define STATE_HAZARD 0x04

// Code Interface
void atlighty_init();
void atl_chest_switch(bool chest_on);
void atl_brake_switch(byte brake_on);
void atl_light_state(byte state);

void showStrip();
void setPixel(int Pixel, byte red, byte green, byte blue);
byte * Wheel(byte WheelPos);
void rainbowCycle();
void RunningLights(byte red, byte green, byte blue, bool is_left);
void Sparkle(byte red, byte green, byte blue);
#endif

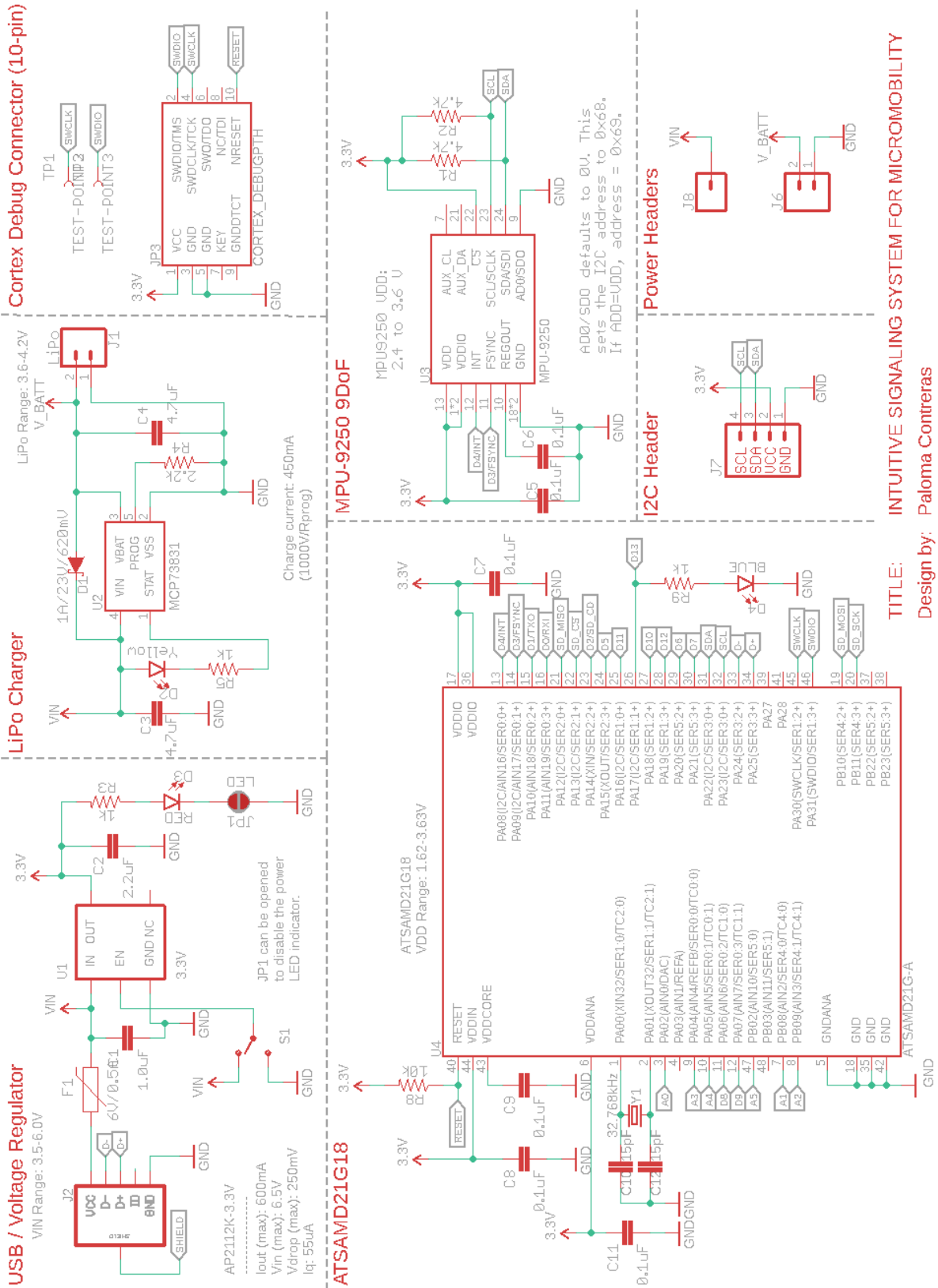
```

Appendix B



Board Schematic





TITLE: INTUITIVE SIGNALING SYSTEM FOR MICROMOBILITY

Design by: Paloma Contreras
based on the SparkFun 9DoF Razor

Figure 13. Detailed Board Schematic

