UNIVERSITAT POLITÈCNICA DE VALÈNCIA

_ TELECOM ESCUELA TÉCNICA VLC SUPERIOR DE INGENIERÍA DE TELECOMUNICACIÓN

CONTROL DE UN SISTEMA DE INFORMACIÓN PARA BOMBEROS MEDIANTE GESTOS DE LA MANO

Autor: Diego Paracuellos de los Santos

Tutor: Jose Manuel Mossi Garcia

Trabajo Fin de Máster presentado en la Escuela Técnica Superior de Ingenieros de Telecomunicación de la Universitat Politècnica de València, para la obtención del Título de Máster en Ingeniería Telecomunicación

Curso 2019-20

Valencia, 15 de Noviembre de 2020

Escuela Técnica Superior de Ingeniería de Telecomunicación
Universitat Politècnica de València
Edificio 4D. Camino de Vera, s/n, 46022 Valencia
Tel. +34 96 387 71 90, ext. 77190
www.etsit.upv.es

VLC/ CAMPUS VALENCIA, INTERNATIONAL CAMPUS OF EXCELLENCE

EMAS Gestión ambiental verificada REG.NO. ES-CV- 000030

Resumen

El oficio de bombero es una profesión de riesgo, siendo que la eficiencia en las labores de emergencia es clave para asegurar la integridad de posibles operaciones en las que pueden estar en juego las vidas de personas o animales. El Fuego, el humo o los escombros son solo algunas de las dificultades a las que potencialmente deben enfrentarse, siendo que una rápida respuesta y la información del entorno pueden resultar vitales para salvaguardar no solo su propia vida, sino la de la colectividad de las personas.

Este Trabajo de Fin de Master pretende implementar el prototipo para un sistema de información y control en tiempo real de baja latencia empleado para manejar un determinado equipamiento mediante técnicas de Control Gesticular directo y Redes Neuronales empleando “electrónica de consumo”, lo que conllevará en el desarrollo de un sistema económico, fiable y ampliamente desplegable.

El resultado de este trabajo ha sido el desarrollo de un sistema capaz de reconocer siete gestos realizados mediante un sistema de captación del movimiento de la mano y actuar de manera casi instantánea de acuerdo a una programación designada para controlar los modos de una cámara y el movimiento por un mapa.

Resum

L'ofici de bomber és una professió de risc, sent que l'eficiència en les labors d'emergència és clau per a assegurar la integritat de possibles operacions en les quals poden estar en joc les vides de persones o animals. El Foc, el fum o els enderrocs són només algunes de les dificultats a les quals han de potencialment enfrontar-se, sent que una ràpida resposta i la informació de l'entorn poden resultar vitals per a salvaguardar no sols la seua pròpia vida, sinó la de la col·lectivitat de les persones.

Aquest Treball de Fi de Màster pretén implementar el prototip per a un sistema d'informació i control en temps real de baixa latència emprat per a manejar un determinat equipament mitjançant tècniques de Control Gesticular directe i Xarxes Neuronals emprant "electrònica de consum", la qual cosa comportara en el desenvolupament d'un sistema econòmic, fiable i àmpliament desplegable.

El resultat d'aquest treball ha sigut el desenvolupament d'un sistema capaç de reconéixer set gestos realitzats mitjançant un sistema de captació del moviment de la mà i actuar de manera quasi instantània d'acord amb una programació designada per a controlar les maneres d'una càmera i el moviment per un mapa.

Abstract

The Firefighter Trade is a high-risk profession, and the efficiency in emergency work is a key factor to ensure the integrity of possible operations in which human or animal lives may be in danger. Fire, smoke or debris are just some of the difficulties that they must potentially face, being that a quick response and information from the environment can be vital to safeguard not only their own life but the collective lives of people.

This Master's Final Project objective is to implement the prototype for a low-latency real-time information and control system used to manage certain equipment through direct Gesture Control techniques and Neural Networks using “consumer electronics”, which will lead to the development of an inexpensive, reliable and widely deployable system.

The result of this work is the development of a system capable of recognizing up to seven gestures made through a system for capturing the movement of the hand and acting almost instantaneously according to a designated programming to control the modes of a camera and movement in a map.

Índice

Capítulo 1. Introducción

1.1 Introducción

En situaciones de emergencia, tanto la rapidez de acción como la eficacia de los operarios pueden constituir la clave en el cometido de salvaguardar las vidas de las personas. Y, es más: tales acciones en aras a salvaguardar vidas humanas no deben agotarse en acciones (físicas) de los propios operarios de un determinado servicio de salvamento, sino que además deben auxiliarse a través de todas las herramientas que existan a su disposición, dotándose de ese modo de un equipamiento adaptado a las diversas circunstancias que puedan surgir en el desempeño de las labores propias de su trabajo.

En tal sentido, por ejemplo, con un equipamiento adecuado (como puede ser una imagen termográfica en tiempo real) un operario de un determinado servicio de bomberos puede incrementar la probabilidad de solventar las dificultades encontradas en un terreno hostil, como puede ser una sala repleta de humo. En la actualidad, dicha imagen termográfica se visualiza en equipamiento externo y relativamente pesado, operado por un miembro del equipo, lo que en ocasiones puede ser poco práctico y dependiente de la situación.

Con los años, la electrónica, y más concretamente los sistemas de procesamiento de uso general, ha alcanzado unos niveles de miniaturización e integración de circuitos que han planteado la posibilidad de integrar muchas de las funciones que se realizaban con equipamiento externo a lo que sería el equipamiento personal del operario. Si bien han existido y existen propuestas para dicha integración, en muchos casos son propuestas que no han llegado a materializarse, bien sea por cuestiones técnicas, físicas o monetarias.

Por otro lado, el desarrollo y evolución de los sistemas denominados como Redes Neuronales ha permitido diseñar sistemas de análisis y toma de decisiones rápidas en diferentes ámbitos de nuestra sociedad.

Con ello y bajo estos preceptos, este trabajo pretende realizar un prototipo demostrador de un sistema de control para el equipamiento de un agente de bomberos, tomando como punto inicial un determinado equipamiento previamente diseñado, y el empleo de Redes Neuronales como sistema de análisis de señales.

1.2 Objetivos

El principal objetivo de este Trabajo de Fin de Master se podría resumir en desarrollar un sistema interfaz hombre – máquina capaz de ejecutar acciones arbitrarias y de acción inmediata (Sistema en Tiempo Real) mediante gesticulación con la mano.

Además de dicho objetivo, adicionalmente se han planteado dos subobjetivos de cara a la presente memoria:

- Analizar la viabilidad del sistema desde un punto de vista electro-técnico.
- Plantear una visión comparativa con otras soluciones posibles.

Por otro lado, además, se pretende desarrollar un sistema controlado, que sea fácil de modificar y utilizar, partiendo de una base simple como es la electrónica de consumo.

Con estos objetivos en mente, se pretende así dar una visión general de las ventajas que podría suponer la implementación de un sistema de este tipo mediante las técnicas sugeridas y/o desarrolladas a lo largo de esta memoria.

1.3 Metodología de Trabajo.

Para conseguir alcanzar el objetivo principal de este Trabajo de Fin de Master, y en relación al material facilitado, que será descrito más adelante, se ha decidido emplear una metodología de desarrollo basada en la Segmentación Funcional de las diferentes partes que conformarían este proyecto.

Dicha Segmentación Funcional, se ha tomado y diseñado con la idea de simplificar las diferentes subtareas que conforman el proyecto global, de forma que sea lo más simple su implementación y definición, y con la capacidad, en muchos casos, de poder trasladar su funcionamiento entre los diferentes dispositivos que puedan conformar el sistema.

Si bien dicha estructuración supondrá un pequeño sacrificio inicial en temas de optimización y, teniendo en cuenta que el sistema final no ha sido plenamente especificado, dicha optimización podrá ser aplicada a posteriori en función de las capacidades de dicho sistema.

Por otro lado, durante el desarrollo, se han ido planteado cinco fases diferenciables entre sí:

1- Implementación básica del sistema de captura y envío de datos actual en un dispositivo A.
2- Implementación básica de la recepción, captura y tratamiento de los datos en un dispositivo B.
3- Implementación y Testeo de las funcionalidades de la Red Neuronal a un nivel básico en el dispositivo B.
4- Implementación de Funcionalidades para comprobar la operabilidad del sistema en el dispositivo B.
5- Optimización de códigos, ajuste del sistema y resolución de problemas.

1.4 Trabajo Relacionado

Existe una amplia gama de proyectos o investigaciones que tienen algún grado de relación con el trabajo presente. En general, podemos encontrar tres grupos con los que comparar: uno relacionado con la ampliación de las capacidades de los operarios, un segundo relacionado con el control remoto de sistemas y finalmente los relacionados con el empleo de herramientas para el reconocimiento gestual. A continuación, se introducirán se introducirán algunos de estos trabajos:

Por un lado, encontramos el trabajo "*Cognitive Adaptive Man Machine Interfaces for the Firefighter Commander*" de *Maurits de Graaf et al* (2011) [1], en el que se plantea el desarrollo de un control centralizado del estado mental de los operativos por parte del comandante.

También, encontramos "*Remote and In-Situ Multirobot Interaction for Firefighters Interventions under Smoke Conditions*" de *A. Naghsh et al* (2010) [2], en el que se plantea un aumento de las capacidades de los agentes mediante la interacción con robots en situaciones de emergencia.

Relacionado con este último, pero introduciendo el concepto de control gestual, encontramos "*On Field Gesture-based Human–Robot Interface for Emergency Responders*" de *Francesca De Cillis et al* (2013) [3], en el que se plantea un acercamiento de control de robots por reconocimiento gestual visual por medio de un algoritmo.

Por otro lado, estaría "*Wireless communication glove apparatus for motion tracking, gesture recognition, data transmission, and reception in extreme environments*" de *Marion G. Ceruti et al* (2009) [4] en el que se plantea un esquema de reconocimiento gestual con un dispositivo similar a nuestra propuesta, salvo que empleando hardware especifico.

Finalmente, podemos encontrar el trabajo "*Gloves Gesture Recognition*" de *P. Chitra et al* (2020) [5] en el que se plantea un sistema de comunicación entre personas con impedimento visual e impedimento auditivo usando como parte del sistema un modelo de guante con electrónica comercial de uso general, al igual que se plantea en este trabajo.

Capítulo 2. Marco Teórico

2.1 Peligros en el desempeño de las funciones de un bombero

El oficio de bombero incluye un amplio rango de actividades de actuación, incluyendo algunos de severa peligrosidad como pueden serlo las situaciones contra incendios y derrumbes.

Estos tipos de actuaciones, que podríamos denominar de alta peligrosidad, conllevan a la aparición de una serie de riesgos, ya sean directos/indirectos y/o a corto/largo plazo, para el operativo y la operación que han de ser recogidos y analizados con el objetivo de intentar minimizar sus efectos. En este sentido, podríamos categorizar esos riesgos en función de su naturaleza, como puede ser "riegos directos ligados al escenario", "riesgos directos ligados al equipo" o "riesgos indirectos ligados al desempeño de las funciones".

En la primera, los "riesgos directos ligados al escenario" se recogerían aquellos peligros directamente relacionados con la actuación, como puede ser el fuego, el humo, los escombros, etc. Estos tienen una característica común, y es que no son un peligro solo para el éxito de la misión, como puede ser un bloqueo producido por un derrumbe o la falta de visión debida a una cortina de humo, sino para la integridad de los operarios como en el caso de una deflagración en el escenario, cuya aparición puede llegar a elevar de manera localizada la temperatura del escenario hasta los 300º C con flujos de energía de hasta 12 kW/m2 [6].

La información, el desarrollo y empleo de funcionalidades pueden ser vitales para minimizar los efectos de este primer grupo de riesgos.

El segundo grupo, los "riesgos directos ligados al equipo" si bien es el menos relevante, supone también un factor a tener en cuenta de cara al desarrollo. Algunas funcionalidades como la visión térmica/Infrarroja/Cuasi-Infrarroja empleada para sortear las cortinas de humo o detectar focos de incendio se realizan con instrumentos que podríamos denominar "pesados", empleados por uno de los agentes del operativo similar al que se muestra a continuación.

Figura 1. Cámara Térmica de Mano

La maniobrabilidad, el manejo o la dependencia del grupo va ligada al agente con el equipo, siendo en este caso el centro sensitivo de todo el equipo. En la actualidad y gracias a la investigación y miniaturización de las diferentes tecnologías, sería posible trasladar la gran mayoría de funciones realizadas por lo que consideraríamos equipamiento tradicional a un equipo integral y personal para cada operario, habilitando la posibilidad de conocer y controlar la situación para cada miembro del equipo.

En el tercer grupo, "los riesgos indirectos ligados al desempeño de las funciones", encontraríamos aquellas enfermedades o dolencias, ya sea físicas o psicológicas, que pueden aparecer por medio de los dos grupos anteriores. Este grupo abarcaría desde quemaduras que causen un daño permanente que afecten a la efectividad del bombero o fobias hasta la paranoia o el posible cáncer producido por la exposición prolongada a espacios con humo y/o fuego llegando en casos externos a la muerte de personas.

Un correcto manejo de las necesidades de los demás grupos, así como cierto grado de flexibilidad y/o adaptabilidad en el equipamiento pueden ser vitales para minimizar la aparición y las consecuencias de dichas dolencias.

2.2 Sistemas de Adquisición

Dentro de un sistema, existe una clase particular de elementos que permiten la adquisición de información, ya sea del entorno o del propio sistema, mediante la transformación de la misma en algún tipo de señal, como podría ser eléctrica, magnética, térmica o acústica, para su posterior utilización por parte de otros elementos, conocidos comúnmente como sensores y/o transductores.

Si bien existen unidades de sensado para prácticamente todos los fenómenos conocidos, como puede ser la temperatura, la humedad, la presión, el campo magnético o incluso el campo gravitacional, en nuestro caso, vamos a centrarnos en un tipo particular de sensores, aquellos capaces de capturar y traducir el movimiento a alguna entidad que seamos capaces de trabajar con ella, como puede ser una señal eléctrica.

2.2.1 Marco Histórico

Si bien la historia de los sensores tal y como los conocemos hoy en día es relativamente reciente, existen una serie de eventos dignos de mentar dentro del contexto de los instrumentos empleados.

En 1784, el matemático *George Atwood* diseño y construyo lo que se podría considerar el primer instrumento mecánico capaz de medir la aceleración de la historia, conocida como la *Máquina de Atwood*. El objetivo de este dispositivo era demostrar y verificar las leyes mecánicas del *Movimiento Uniformente Acelerado*.

Un tiempo después, en 1817, el astrónomo *Johann Bohnenberger* escribió sobre lo que se conoce como "*Efecto Giróscopo*" mediante la descripción de un aparato al que simplemente denominó como la "Máquina".

En 1851, *Léon Foucault* construyó un dispositivo oscilante basado en la *"Máquina" de Bohnenberger*, con el objetivo de aplicar el concepto y demostrar la rotación terrestre. Este dispositivo, conocido como *Péndulo de Foucault*, era capaz de demostrar dicha rotación de manera indirecta.

Mas tarde, en 1852, el mismo *Foucault* diseñó otro dispositivo capaz de medir de manera directa la rotación de la tierra, al que denominó *Giroscopio*, acuñando el término.

No fue hasta 1904 cuando el giroscopio tuvo una aplicación real, en lo que años atrás se bautizó como *girocompás*, patentado por *Hermann Anschütz-Kaempfe*. Este dispositivo es un tipo de compas no magnético, que se popularizó en el mundo naval y de la aviación.

En el año 1920 aparecieron los primeros dispositivos comerciales de lo que conocemos como acelerómetros, de manos de *McCollum* y *Peters*. Estos sensores, de peso aproximado de medio kilo y con dimensiones de 2 x 4.7 x 22 cm [7], fueron utilizados principalmente en el mundo de la aviación y Dinamómetros.

Posteriormente, y debido a la carrera armamentística producida durante la segunda guerra mundial, estos dispositivos sufrieron un boom de desarrollo y miniaturización, para su empleo en armamento pesado y/o balístico.

Ya en 1990 fue cuando la tecnología permitió la fabricación de dispositivos en masa produciéndose una estabilización y estandarización con la aparición de los primeros acelerómetros microelectromecánicos comerciales.

2.2.2 Sensores de Movimiento

Dentro de lo que sería el conjunto de los sensores, existe una subcategoría encargada de la transducción de lo que sería el movimiento a señales eléctricas. Dicha conversión puede ser llevada a cabo de diferentes maneras, ya sea con galgas extensiométricas o elementos piezoeléctricos para medir la dilatación o expansión de un determinado objeto cuando está en movimiento o bien por medio de fuerzas inerciales.

En relación a estos últimos, existen dos tipos de sensores que resultan relevantes para este proyecto: los acelerómetros y los giroscopios.

El primero, el acelerómetro, es un dispositivo que emplea las fuerzas inerciales aplicadas sobre un condensador o un componente piezoeléctrico para generar una señal variable con la fuerza relativa aplicada a un determinado movimiento del sensor y calcular la aceleración relativa del mismo.

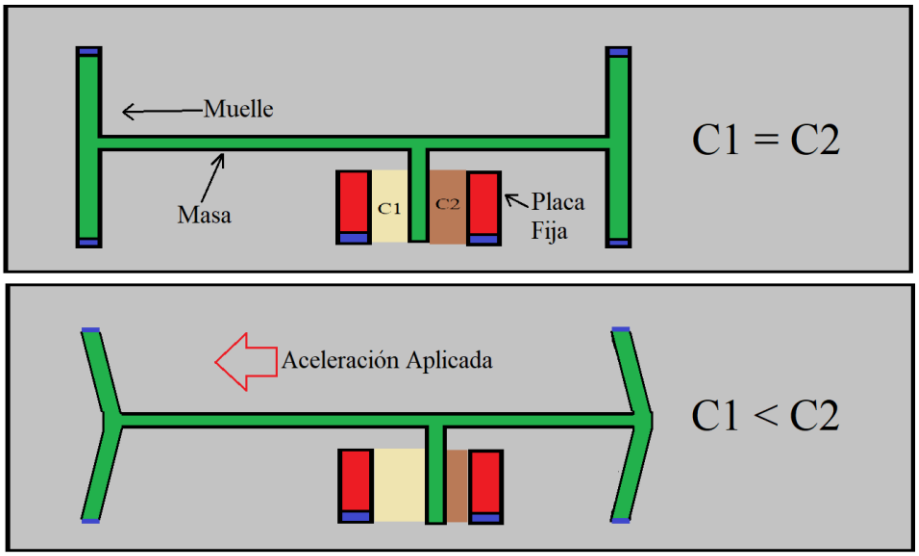

Figura 2. Funcionamiento de acelerómetro (MEMS)

No obstante, la mayoría de acelerómetros, incluso en estado de reposo, emiten una cierta lectura de aceleración en alguno de sus ejes. Esta aceleración es el resultado de la fuerza gravitacional terrestre, que, por el principio de equivalencia de relatividad general, es imposible de diferenciar de una fuerza de aceleración.

Existen multitud de usos para estos dispositivos, como puede ser la captura de movimiento en una pulsera deportiva o un vehículo o la detección de caídas para un dispositivo utilizado por personas de la tercera edad; siendo que, en la actualidad, una gran parte de los dispositivos electrónicos comercializados contienen al menos un acelerómetro.

Por otro lado, están los giroscopios, que son dispositivos de medición de la velocidad angular. Si bien existen diferentes tipos de giroscopio, en nuestro caso, sería más correcto hablar de los giroscopios de estructura vibrante o de Coriolis Oscilante (CVG, del inglés Coriolis Vibratory Gyroscope).

El principio de funcionamiento de estos, se basa en el principio de los objetos a seguir vibrando en un determinado plano con independencia de la rotación del soporte que los mantenga. Con esta vibración y gracias al efecto Coriolis, se aplica una fuerza en el soporte que puede ser medida y convertida en una rotación.

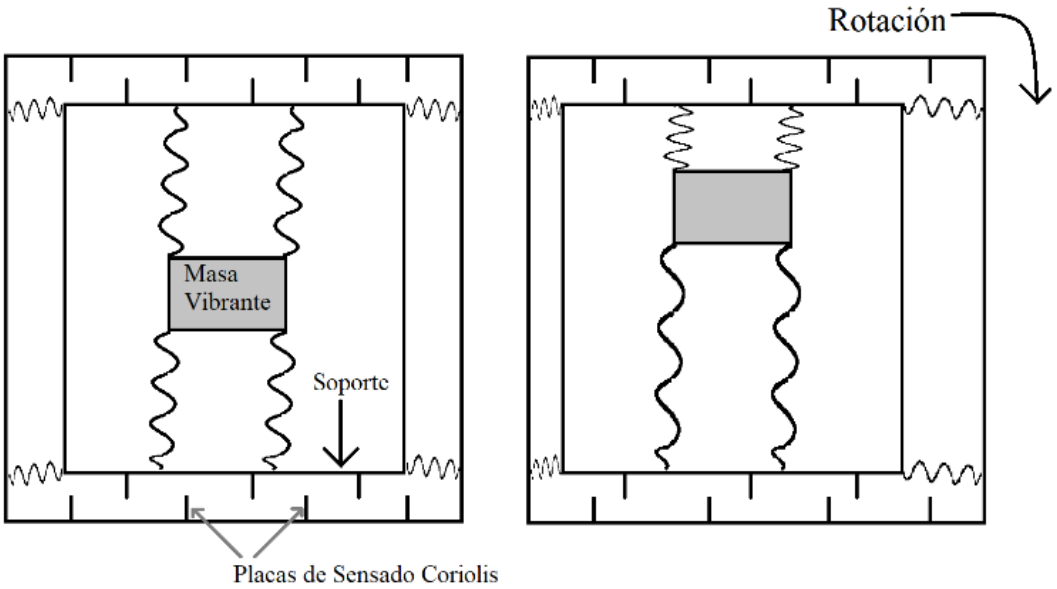

Figura 3. Funcionamiento de Giroscopio (MEMS)

Un uso frecuente de este tipo de giroscopio es encontrado en forma de sistemas microelectromecánicos (MEMS), que, junto a acelerómetros, son capaces de formar sistemas con seis grados de libertad (6DoF) en lo que se conoce como Unidades de Medición Inercial (IMU).

Estos seis grados de libertad se podrían definir como las direcciones de los tres ejes coordenadas, comúnmente denotados como X, Y, Z, y la rotación del IMU en estos mismos ejes, comúnmente detonado con los nombres de Roll, Pitch y Yaw, correspondientes al giro en los planos ZY, XZ y XY.

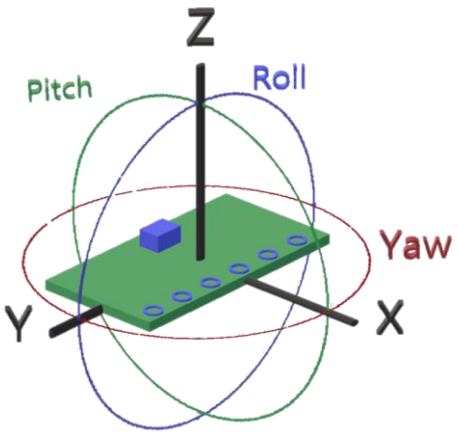

Figura 4. Los seis grados de libertad de un IMU.

2.3 Sistemas de Procesamiento.

2.3.1 Marco Histórico

Considerando los sistemas de procesamiento como la unión de los sectores de la electrónica y la computación, existen una serie de hitos que merecen ser mencionados dentro de estos dos campos.

El primer concepto de computación moderna apareció entre los años 1936 y 1937 de manos de *Alan Turing* en su artículo "*On Computable Numbers, with an Application to the Entscheidungsproblem*", asentando las bases no solo de la arquitectura computacional sino del almacenamiento de programas.

En 1944 se desarrolló el primer ordenador puramente electrónico y digital, recibiendo el nombre de *Colossus*. Un par de años más tarde, en 1946, el primer ordenador programable aparecería bajo el nombre de *Eniac*.

No fue hasta la época de 1950 que aparecerían los primeros lenguajes de ensamblado y compiladores.

En 1958 se crea el primer circuito integrado por parte de *Jack St Clair Kilby* para *Texas Instruments*.

En la época de 1960 surgió un boom en el campo de la miniaturización y optimización, apareciendo los primeros "mini" ordenadores y apareciendo lenguajes como *BASIC* y *PASCAL*.

En 1971, aparecería el primer dispositivo que podría considerarse como un microcontrolador, el *TMS1802NC*, después rebautizado como *TMS series 0100*, desarrollado por *Gary Boone* y *Michael Cochran* para *Texas Instrument*. Este dispositivo implementaba una CPU de 4 bits, 275 bytes de memoria para programa (ROM) y 16 bytes para memoria de acceso aleatorio (RAM).

Años más tarde, en 1980 aparecería el primer sistema operativo comercial bajo el nombre de ***VRTX*** (**V**ersatile **R**eal **T**ime e**X**ecutive) y en 1984 los primeros dispositivos de puerta programable (FPGA) de manos de *Xilinx*.

Finalmente, en la época de 1990 hasta la actualidad se ha producido una segunda etapa de miniaturización y optimización, incluyendo la aparición de multitud de sistemas operativos y lenguajes de programación.

2.3.2 Microcontroladores

Un microcontrolador podría definirse como un circuito integrado (IC) con la capacidad de ejecutar órdenes, las cuales pueden ser definidas por un programador, alojadas en su memoria. Todo microcontrolador consta básicamente de tres partes o módulos fundamentales:

- Una unidad central de procesamiento (CPU), también conocida como microprocesador.
- Una cierta cantidad de memoria, para el almacenamiento de datos.
- Una serie de periféricos o puertos de entrada y salida (I/O) para la interacción con otros Dispositivos o Circuitos.

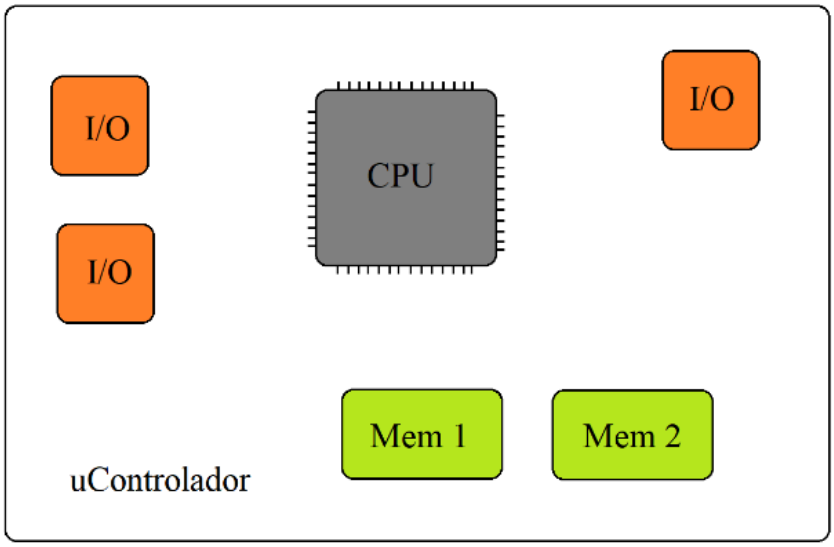

Figura 5. Esquema de microcontrolador.

La ventaja básica de este tipo de dispositivos, es que permiten la automatización de rutinas de diferentes naturalezas, con un coste y consumos mínimos.

En la actualidad, lo más común es encontrar microcontroladores de 8, 16 o 32 bits, aunque existen modelos que son capaces de operar hasta 64 bits (basados en *PowerPC*), todos ellos con una capacidad de almacenamiento variable de unos pocos kilobytes a algunos Megabytes, con unas velocidades de reloj relativamente bajas comparadas con otros dispositivos, rondando el orden de los MHz.

Si bien, de manera tradicional, la gestión de recursos y la interoperabilidad con otros dispositivos debía de implementarse de manera manual, en la actualidad existe la posibilidad al trabajar con muchas familias de microcontroladores de emplear lo que se conoce como *Sistemas Operativos en Tiempo Real* (RTOS), que permite optimizar los tiempos de ejecución y uso de los dispositivos en los que es posible su integración, otorgando además de un contexto temporal definido y unas herramientas internas que permitan facilitar la implementación de códigos.

A continuación, se presenta un ejemplo de sistema RTOS:

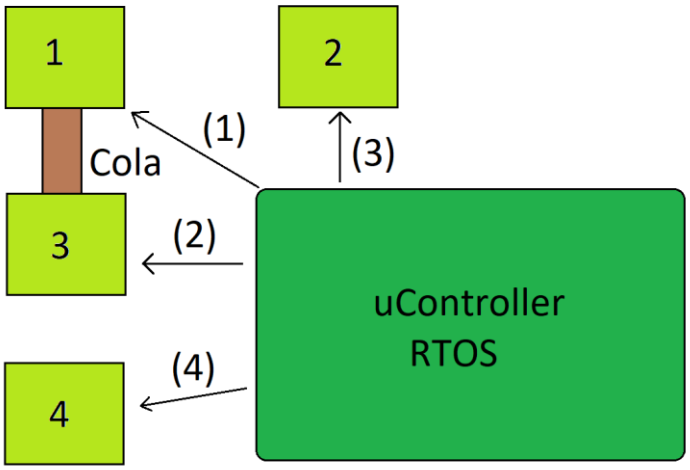

Figura 6. Ejemplo de RTOS.

En este ejemplo, representado por el entorno RTOS en un microcontrolador, cuatro procesos y una cola se da la siguiente secuencia de acciones.

(1) - Inicialmente, el proceso 1 toma el control del sistema. Este envía un mensaje por la cola al proceso 3 y cede el control del sistema al RTOS para que otro proceso pueda ejecutarse.
(2) - Después, cuando el proceso 3 recibe el mensaje, se inicia, procesándolo y entrando en reposo al completar su función.
(3) - Tras esto, el proceso 2, pide el control del sistema para ejecutar una subrutina propia.
(4) - Al terminar, el proceso 4 es el que toma el control, ejecutando su propio código hasta que otro proceso pida el control o el RTOS deba ejecutar alguna de sus propias rutinas.

Esto, ejemplifica algunas de las funcionalidades capaces de realizar mediante un sistema RTOS, como puede ser la gestión de procesos o la creación de colas controladas.

2.4 Redes Neuronales

2.4.1 Marco Histórico

Las redes neuronales en el ámbito computacional es un concepto que data de la historia reciente, siendo la primera aparición y desarrollo del concepto del año 1943 en mediante un esquema eléctrico de *Warren McCulloch* y *Walter Pitts* [8]. Posteriormente, en el año 1949, *Donald Hebb* propuso en su libro "*The Organization of Behaviour*" en modelo de interconexión y refuerzo entre neuronas. No obstante, no fue hasta 1954 que se logró desarrollar un modelo exitoso basado en la teoría de *Hebb*, por parte del *Massachusetts Institute of Technology* (MIT).

Unos años más tarde, en 1958, apareció por primera vez el concepto de Perceptron a manos de *Frank Rosenblatt* basado en la definición de neurona de *McCulloch* y *Pitts*. En el año 1959, apareció la primera red neuronal aplicada a un problema real, recibiendo el nombre de **MADALINE** (**M**ultiple **ADA**ptative **LIN**ear **E**lements) creadas para eliminar el problema de ruido en las líneas telefónicas.

Posteriormente, en el año 1965, aparecieron las primeras redes funcionales multicapas por parte de *Ivakhnenko* y *Lapa*, hito al cual siguió una época que duraría hasta el año 1982 en la que se paralizó el desarrollo en el campo de las redes neuronales debido al miedo y falsas premisas que aparecieron durante la misma.

En el año 1982, se produjeron dos eventos que volvieron a activar el interés y desarrollo de las redes neuronales. El primero, fue un documento publicado *John Hopfield*, proponiendo su uso para crear dispositivos, y el segundo el anuncio de la quinta generación por parte de Japón durante una conferencia grupal entre Japón y Estados Unidos, que impulsó la inversión en el desarrollo de redes.

El desarrollo e investigación siguió creciendo hasta el año 1997, año en el cual se propuso un modelo de red recurrente funcional (*LSTM*) por parte de *Schmidhuber* y *Hochreiter*.

Por último, en 1998, *Yann LeCun* propone un algoritmo de aprendizaje por gradiente en "*Gradient-Based Learning Applied to Document Recognition*", iniciando una época de avances que perdura hasta la actualidad.

2.4.2 Redes Convolucionales/Recurrentes

En la actualidad existen dos grandes arquetipos de redes que se emplean con mayor frecuencia, las redes convolucionales y las redes recurrentes. Si bien a lo largo de este apartado hablaremos de ambos tipos de redes, cabe destacar que en muchos casos estos tipos de redes incluyen algunas capas que se comparten entre diferentes tipos, por lo que solo se hablará de aquellas capas que diferencian el funcionamiento de las mismas.

Las redes convoluciones (*CNN*) están basadas en lo que se conoce como redes de perceptrones multicapa, las cuales, a su vez, derivan de la idea tradicional de las redes neuronales basadas en una unidad de cómputo llamada perceptrón y siendo las CNN un tipo normalizado de las mismas.

De manera básica, un perceptrón se comporta siguiendo la siguiente ecuación:

$$f(x) = \begin{cases} 1 \; si \; w * x + b > 0 \\ 0 \quad\quad en \; otro \; caso \end{cases} \tag{1.1}$$

Siendo:

- w, un valor peso
- x, la entrada de datos
- b, un valor de polarización

Definiendo de este modo la ecuación condicional una región lineal delimitante.

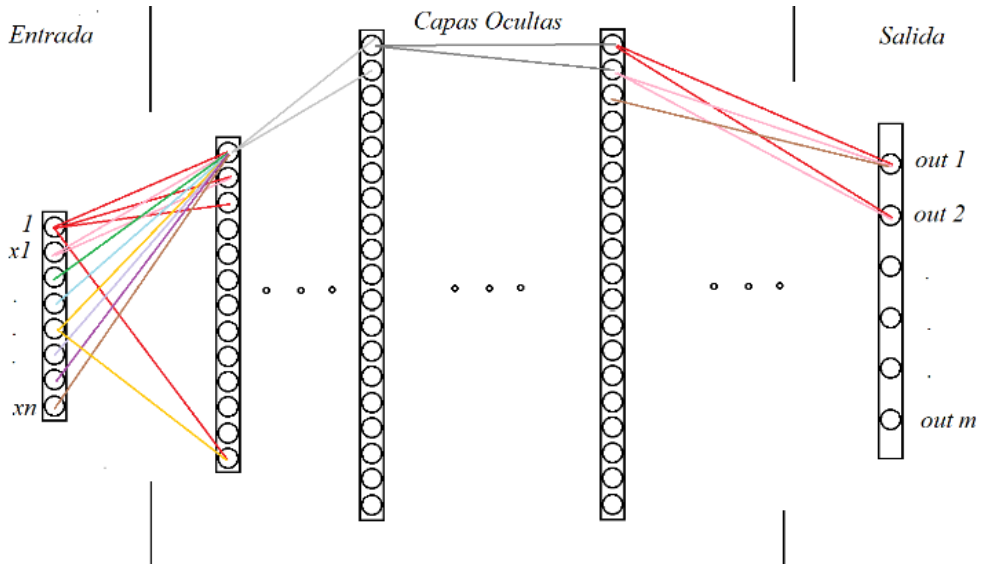

Figura 7. Red Convolucional

De forma tradicional, en una red convolucional, todos los perceptrones de una capa, se conectan con todos los perceptrones de otra capa, formando una red completamente conectada (full-conecte).

Estas capas, por lo general, se traducen en tres categorías: Entradas, Salidas y Capas Ocultas. Si bien la Entrada y la Salida suelen estar dimensionadas por diseño en función del número de entradas y salidas deseadas [N y M], las capas ocultas pueden emplear procesos de expansión y contracción de datos para lograr la diferenciación de eventos o funcionalidades.

Este tipo de redes, son principalmente empleadas para reconocimiento y/o "tratamiento" de imagen, permitiendo tomar la totalidad o parte de una imagen y categorizarla en función de los elementos apreciados en ella.

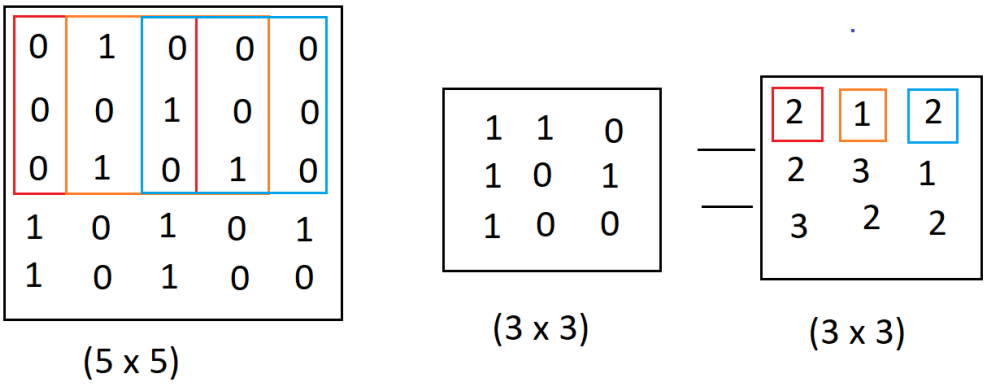

Figura 8. Convolución por multiplicación de matrices.

En la práctica, el funcionamiento de este tipo de redes se basa en producto de convolución entre matrices, además, siendo posible la descomposición en operaciones matriz-matriz más simples.

Por otro lado, encontramos las redes recurrentes. Este tipo de redes derivan de las redes feed-forward, y al contrario que las convolucionales, no presentan conectividad completa, sino que están formadas de nodos formando un grafo dirigido, así como una secuencia temporal.

Esto hace que, de manera teórica, presenten lo que denominaríamos memoria, y, por lo tanto, cierta capacidad de mantener y contabilizar los estados pasados como parte de su cálculo, por lo que resultan óptimas para tareas como reconocimiento de escritura o secuencias de eventos específicas.

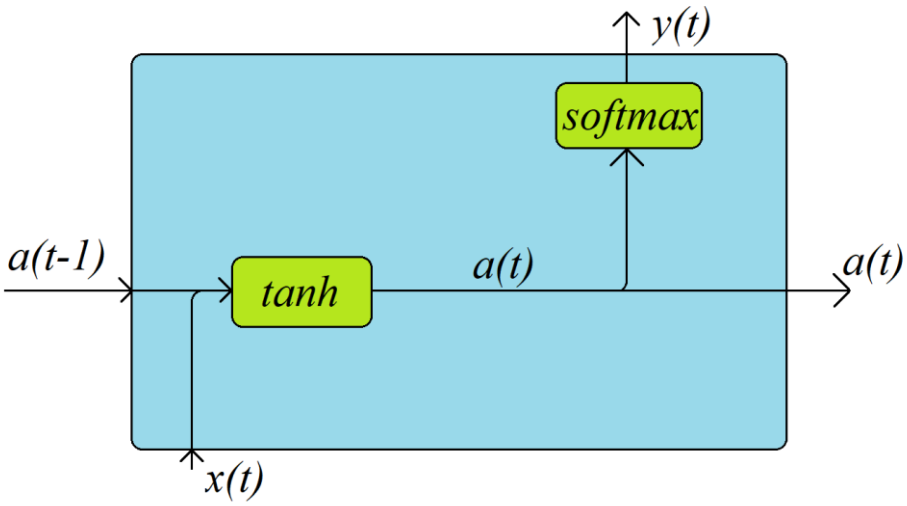

Figura 9. Célula Recurrente

Una RNN de manera básica se definiría por las siguientes ecuaciones:

$$a(t) = \tanh(W_{ax} * x_t + W_{aa} * a_{t-1} + b_a) \quad (2.1)$$
$$y(t) = softmax\left(a_t + b_y\right) \quad (2.2)$$

Siendo:

x_t , el vector de entrada, en un tiempo t.

a_t , el vector de estado "oculto".

W_{ax}, W_{aa} , las matrices con los pesos de las entradas y los estados.

b_a, b_y , los valores de polarización.

Si bien, existen muchas variantes dentro del conjunto de las RNN, como pueden ser las *Gate Recurrent Unit* (GRU), las *máquinas de Turing Neuronales* (NTM) o las redes de *Elman* y *Jordan*, todas ellas se basan en la ejecución sistemática y continua de una unidad denominada Célula y la actualización del estado de dicha célula para generar las salidas.

No obstante, la mayoría de redes neuronales recurrentes, presentan dos problemas básicos a la hora de ejercer su entrenamiento y posteriormente su funcionamiento, y son los problemas del gradiente evanescente y gradiente volátil. Estos problemas se basan en la aparición durante el entrenamiento de derivadas lo suficientemente pequeñas o grandes como para que el valor de subsiguientes células no cambie o cambie de tal forma que no se correlacione con el valor anterior.

Para solventar estos problemas, una de las soluciones que se han planteado son las redes *Long Short-Term Memory* (LSTM).

2.4.3 Redes LSTM/Bi-LSTM

Las redes *LSTM* son un pequeño subconjunto dentro de las redes recurrentes, diseñadas con la finalidad de resolver uno de los principales problemas de las mismas, el problema del gradiente desvanescente. Dicho problema se resuelve empleando una metodología de entrenamiento Feed-back y Feed-forward en la que, al contrario que en las redes recurrentes tradicionales, no solo se distribuyen los pesos hacia la siguiente célula o estado, sino que se transmiten también a las anteriores.

En lo relacionado a su estructura y funcionamiento, una red *LSTM* se basa en la ejecución recurrente de una unidad denominada célula, las cuales poseen por lo menos tres tipos de entrada/salidas, denominadas puertas.

Estos tres tipos de puertas son:

- la puerta de entrada, encargada de los datos de entrada ya sean externos o de la iteración previa.
- puerta de salida, encargada de los datos enviados al exterior o hacia la siguiente iteración.
- puerta del olvido, encargada de determinar la cantidad de información que se reutiliza entre diferentes iteraciones.

Además, ligados a estas puertas, existen una serie de pesos y valores de polarización que permiten caracterizar la red durante el entrenamiento.

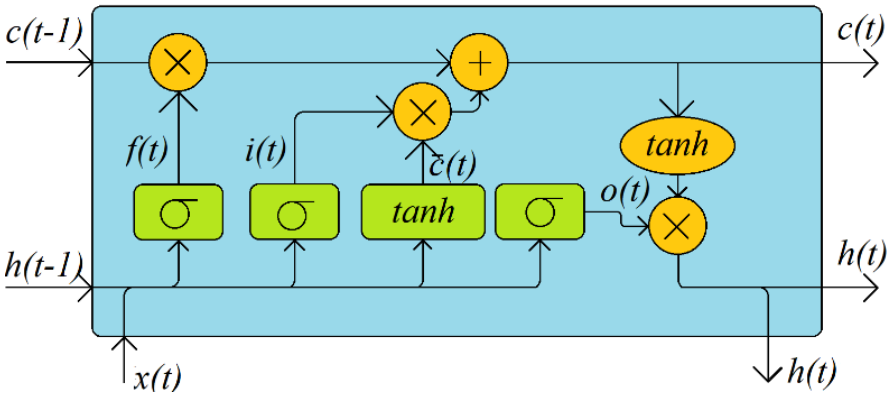

Figura 10. Célula LSTM

Toda red *LSTM* se rige, de manera básica, por las siguientes ecuaciones:

$$f_t = \sigma\left(W_f x_t + U_f h_{t-1} + b_f\right) \quad (3.1)$$

$$i_t = \sigma(W_i x_t + U_i h_{t-1} + b_i) \quad (3.2)$$

$$o_t = \sigma(W_o x_t + U_o h_{t-1} + b_o) \quad (3.3)$$

$$c_t = f_t * \ c_{t-1} + i_t * \tau(W_c x_t + U_c h_{t-1} + b_c) \quad (3.4)$$

$$h_t = o_t * \tau(c_t) \quad (3.5)$$

Siendo:

x_t el vector de entrada en un tiempo t, de tamaño n x 1.

f_t, el vector de la puerta de olvido (forget gate), de tamaño m x 1.

i_t, el vector de la puerta de entrada (input gate), de tamaño m x 1.

o_t, el vector de la puerta de salida (output gate), de tamaño m x 1.

h_t, el vector de salida, con tamaño m x 1.

c_t, el vector del estado de celda (cell state), con tamaño m x 1.

W_f, W_i, W_o, W_c, Las matrices con los pesos de la puerta de entrada, de tamaño m x n.

U_f, U_i, U_o, U_c, las matrices con los pesos de la puerta de salida, con tamaño m x m.

b_f, b_i, b_o, b_c, los pesos de polarización (bias), de tamaño m x 1.

y siendo n, m, σ y τ los tamaños de la entrada, estado de celdas y salida, la función sigmoide y la función tangente hiperbólica respectivamente.

La principal ventaja de este tipo de redes es que son bastante eficientes para mantener y predecir los estados a largo plazo, es decir, son capaces de actuar en relación a una secuencia de eventos relativamente larga, y con independencia a su tamaño, llegando a poder reconocer eventos ocurridos en miles o millones de pasos temporales discretos anteriores.

En los años recientes, se pueden encontrar variedad de ejemplos de utilización de redes *LSTM* en diferentes ámbitos del sector de las TIC, como puede ser la implementación de un algoritmo de reconocimiento de voz por parte de *Google* en el año 2015, que mejoró su tasa de error en un 49% respecto a modelos anteriores, la implantación de las mismas como parte del modelo de

sugerencia de mensajes en el servicio *Allo* de *Google* en el año 2016, con una mejora de 60% o en el año 2017, la implementación de las *LSTM* como ayuda a la traducción automática por parte de *Facebook*, con la capacidad de traducir 4.500 Millones de Palabras diarias.

Existen otros ejemplos como puede ser *OpenAI*, proyecto llevado a cabo en 2018, o *DeepMind*, en 2019, con la intención de simular la toma de decisiones y predicción de escenarios del ser humano aplicándolo al sector competitivo de los eSports, y ambos considerados como un paso adelante hacia la Inteligencia Artificial General.

Además de las *LSTM* normales, existe una variante especial de las mismas llamada *LSTM Bidireccional (BiLSTM)* que se diferencia en que no solo se tiene en cuenta acontecimientos "pasados", sino también "futuros", mediante la formación de una segunda red *LSTM* en dirección contraria.

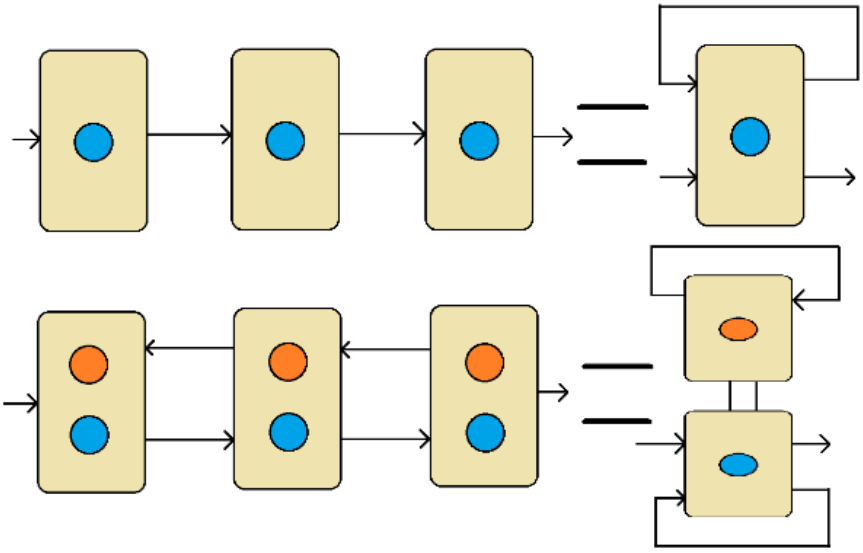

Figura 11. LSTM vs BiLSTM

Tal y como se observa en la figura, para implementar la estructura *BiLSTM*, se desarrolla una segunda red *LSTM* interconectada con la principal cuyas entradas son las mismas que la red original excepto en orden inverso, mejorando la percepción de los conjuntos de entradas cercanas en un sentido temporal.

En este tipo de red, el algoritmo de aprendizaje se introduce tanto en la red directa como en la red inversa, mejorando la capacidad de aprendizaje de la red.

2.5 Sistemas Huésped

2.5.1 Necesidades, Limitaciones y Usos

Para la constitución de un sistema de control en tiempo real, la funcionalidad debe ser implementada en un sistema base. Dicho sistema base, deberá realizar las funciones de co-controlador de un sistema personal que actuará en función de unas determinadas entradas obtenidas a raíz de las salidas de la red neuronal, así como de una serie de variables de estado como discretización del entorno interno del sistema.

Puesto que el sistema deberá ser cargado por una persona, junto al resto de su equipamiento, sería deseable que el sistema huésped fuese un sistema relativamente portátil, de peso y dimensiones reducidas, así como que posea una autonomía relativamente elevada, ya sea por una capacidad de almacenaje elevada o por un consumo relativamente baja.

Por otro lado, interesaría que el sistema posea una potencia lo más elevada posible, puesto que permitira la posibilidad de desarrollar una mayor variedad de funcionalidades y de mayor complejidad. La cantidad de memoria útil que pueda ser empleada también es un punto a valorar.

Además, sería un extra que fuese un sistema transparente para el usuario final, con la finalidad de que este no deba de preocuparse por su funcionamiento.

Sin embargo, deberá de ser un sistema fiable, puesto que estamos hablando de que será usado en ocasiones en situaciones de riesgo con vidas humanas en juego.

Por último, el coste, que, aunque no es un tema que sea primordial, siempre resulta una propuesta interesante si se consigue reducir tanto sus partes de desarrollo como de producción.

2.5.2 Comparativa de Sistemas

Teniendo en mente las condiciones estipuladas en el apartado anterior, se van a presentar una serie de dispositivos que se han considerado interesantes para la utilización como sistema huésped.

En primer lugar, estarían los Microcontroladores o SoC, tal y como se emplea en la captura de señales. Este tipo de dispositivos se caracterizan por aportar una cantidad moderada de capacidad de cómputo por un coste y dimensiones bajos. Si bien entre las opciones que se van a barajar son las que menor capacidad computacional tienen, sobresalen en el resto de condiciones, especialmente en relación a la autonomía/consumo. Dentro de esta familia de dispositivos, podemos encontrar algunos como puede ser el ESP32, diferentes soluciones basadas en ARM64, las cuales podrían resultar una opción muy viable al ofrecer en la actualidad una capacidad de cómputo aceptable por un coste aceptable.

Seguidamente, se podría considerar un sistema basado en microcomputadores. Estos, a pesar de que su rango de precios es relativamente superior a los Microcontroladores, su capacidad de cómputo es igualmente superior, así como sus dimensiones. Dentro de este tipo de dispositivo, se han encontrado dos tipos que podrían adaptarse a las necesidades del sistema, las Raspberry y las NUC, siendo la primera un paso intermedio entre los dispositivos anteriores y estos, y el último simplemente un ordenador de dimensiones limitadas.

Finalmente, se podría contemplar la posibilidad de desarrollar un dispositivo propietario que se ajuste a las necesidades del proyecto basado en procesadores de consumo (i.e ARM64). Si bien seria la opción que mejor adaptase las necesidades del Hardware al proyecto, conllevaría unos costes mucho mayores, aunque permitiría un nivel de integración mayor.

Dispositivo	***Dim. (mm)***	***Controlador***	***bits***	***Velocidad (GHz)***	***Memoria***	***Consumo***	***Coste***
ESP32	55x27x1	Xtensa LX6	32	0.16/0.2	520 kB	1.3W	>8€
Raspberry Pi 4	67x97x27	Cortex-A72	64	1.5	< 8 Gb	15.3W	>50€
NUC7CJH2	114x114x51	Cel. J4005	64	2	< 8 Gb	65W	>114€
NUC8i3BEH	117x112x51	I3-8109U	64	3	< 32 Gb	90W	>274€

Tabla 1. Dispositivos potenciales

La elección del sistema huésped fuera de la etapa de prototipado dependerá meramente de las capacidades de los propios sistemas y del grado de optimización máximo que permitan los mismos, así como de las funcionalidades que se le quiera dotar a el sistema final.

Capítulo 3. Desarrollo

En este apartado, se presentarán los aspectos característicos y específicos de los sistemas y utilidades empleadas durante el desarrollo del proyecto. Si bien se va a incluir determinada información propia del proceso de desarrollo, en general la información será relativa al estado actual del proyecto.

3.1 Definición del Entorno

Este proyecto a grandes rasgos se podría definir en dos ámbitos. Por una parte, un sistema de captación de movimientos de la mano, siendo un sistema previamente diseñado antes del inicio del proyecto, y un sistema huésped que contendrá la base del HMI y la Red Neuronal.

3.1.1 Sistema de Adquisición de Datos (Guante)

Como ya se ha introducido, uno de los ámbitos de este proyecto es el sistema de adquisición de datos. Dicho sistema, consistente en la interconexión de 6 Unidades de Medición Inercial (IMU) a un dispositivo microcontrolador, siendo en este caso el empleado un ESP32 con doble núcleo. Adicionalmente, se dispone de una pequeña batería que otorga autonomía al sistema.

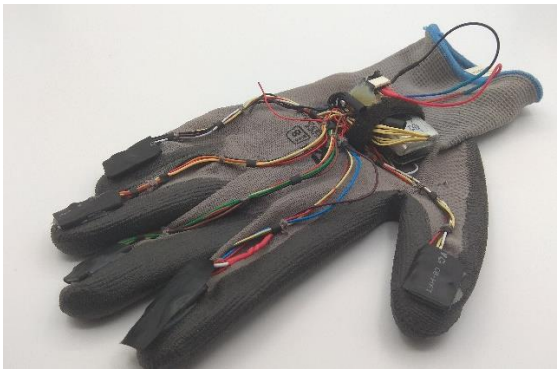

Figura 12. Guante de captura

La interconexión entre todos los distintos circuitos integrados (IC) que componen este diseño, se resumen en la siguiente red/mapa de interconexión.

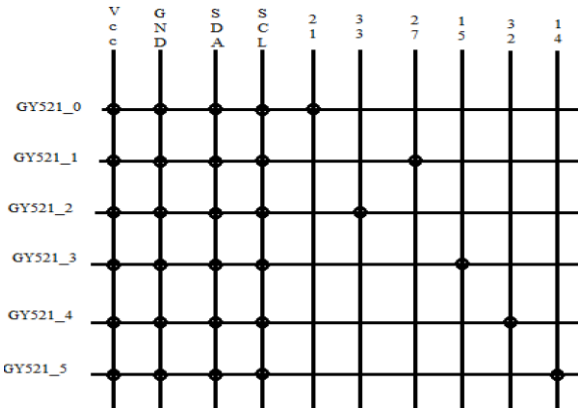

Figura 13. Esquema de interconexión.

Tal y como se aprecia existen cuatro tipos de líneas comunes, y son:

- las líneas comunes relativas a la alimentación VCC a 3.3V.
- las tierras GND.
- las líneas de SDA, que conforman las líneas de datos para el protocolo I2C.
- las líneas de SCL, que conforman las líneas de Reloj para el protocolo I2C.

Adicionalmente, cada IMU cuenta con un conexionado adicional, en su entrada AD0, que sirve como señal de habilitación para designar el IMU activo en un momento determinado. Puesto que las líneas de Datos son comunes a todos los IMUs, se debe procurar que en ningún momento exista la posibilidad de que dos IMUs estén activos al mismo tiempo.

En relación a los dispositivos empleados como IMUs, se tratan de sensores *GY-521*, basado en el *MPU6050* que internamente implementa un Acelerómetro y un Giroscopio. Este sensor permite una configuración bastante variada, siendo posible modificar los fondos de escala tanto del giroscopio como del acelerómetro, así como la tasa de muestro de ambos.

Para este proyecto se ha decidido emplear una tasa de muestro de 50 Hz, lo que limitará el tiempo máximo de operación del sistema a menos de 20 ms en cualquiera de sus etapas. Además, como fondo de escala se ha decidido emplear ±2g en el caso del acelerómetro y 2000º/s en el caso del giroscopio puesto que los movimientos de los dedos de la mano tienen aceleraciones menores a estos valores.

3.1.2 Sistema Huésped

Como sistema Huésped, se ha decidido emplear una computadora personal con las siguientes especificaciones en esta fase del prototipo demostrador:

- Procesador: Intel i5 4460 a 3.2GHz con 4 núcleos.
- Memoria: 16 Gb de DDR3 a 1333MHz.
- Sistema Base: Windows 10 Pro.
- Procesador Gráfico: NVIDIA RTX 2080.

Estas especificaciones, no obstante, actualmente pueden ser encontradas, en forma equivalente y a excepción del procesador gráfico, que será empleado exclusivamente para el entrenamiento de redes, en dispositivos SoC/µPC de coste moderado.

La decisión de emplear dicho sistema como huésped ha sido por la necesidad de unificar el entorno de desarrollo con el entorno de ejecución, y puesto que existen soluciones con capacidades computacionales de CPU equivalentes al entorno de desarrollo para emplear como entorno de ejecución, creemos que no afectará al rendimiento general que pueda manifestarse en un posible producto final.

3.1.3 Entorno de Desarrollo

El entorno de desarrollo, podría separarse en dos partes básicas, el sistema de captura de gestos, en forma de guante con las IMU y el uC, y el entorno de ejecución y procesamiento de los datos, anteriormente llamado Sistema Huésped.

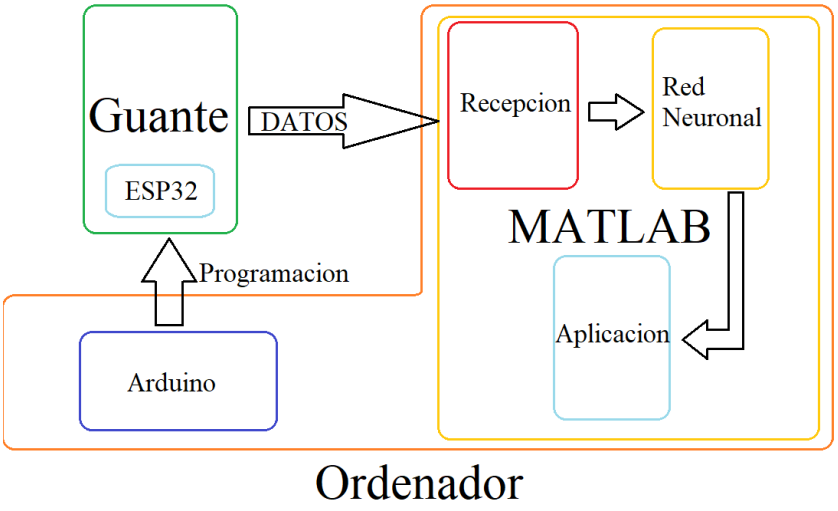

Figura 14. Entorno de Desarrollo

Dentro del sistema de captura, podemos encontrar un controlador básico, el *ESP32*, cuyo programa será preparado desde el sistema de desarrollo, es decir desde la computadora, y cargado en el mismo.

Por otro lado, en el sistema de desarrollo, se han empleado principalmente dos programas para la realización de este proyecto, en primer lugar, *Arduino*, como una IDE que nos permite programar y depurar el código del sistema de gestión de las *IMU* mediante un lenguaje simple y potente gracias a una serie de librerías que permiten programar el *ESP32* directamente desde el mismo, y por ultimo *MATLAB* en su versión *R2019b*, que plantea una suite de funcionalidades que han permitido implementar, desarrollar los diferentes módulos que conforman la red neuronal, así como implementar las diferentes funcionalidades que se han ido planteado para comprobar la viabilidad del sistema.

3.2 Adquisición de Datos

El script facilitado para la adquisición de datos constaba en un sketch de Arduino, que incluía las rutinas de inicialización y una rutina principal de programa.

Dicho programa podría ser definido en 4 diferentes bloques o procesos:

- Bloque de Gestión de IMUs, encargado de realizar los cambios en las líneas de salida para seleccionar el IMU a leer.
- Bloque de Captura de Datos, encargado de recoger y tratar los datos provenientes de los IMUs.
- Bloque de Estructuración de Paquetes.
- Bloque de Envío de Paquetes.

Todos ellos interconectados y ejecutados de manera infinita, con la finalidad de extraer la información sobre la aceleración inducida por el movimiento de los sensores.

Sin embargo, como primer paso para empezar a trabajar, se propusieron dos cambios de cara a mejorar el proyecto, por un lado, convertir lo que por entonces era una estructura mono-núcleo a una estructura bi-núcleo, repartiendo los diferentes bloques entre los dos núcleos que implementa el procesador del ESP32, siendo que uno de los cuales estaba hasta el momento prácticamente sin uso, y siendo el otro la capacidad de enviar la información de los giroscopios como posible información útil para la gestión de gestos por parte de la red neuronal.

3.2.1 Modelo Bi-núcleo del sistema de adquisición

El modelo de captura original, no contemplaba la posibilidad de emplear todo el potencial de un dispositivo multinúcleo. Para atajar este concepto, se identificaron las diferentes secciones de código resumidas anteriormente y se reestructuraron para separarlas en dos programas que correrían en dos núcleos. A continuación, se representa un esquema con la formación actual de dichos programas.

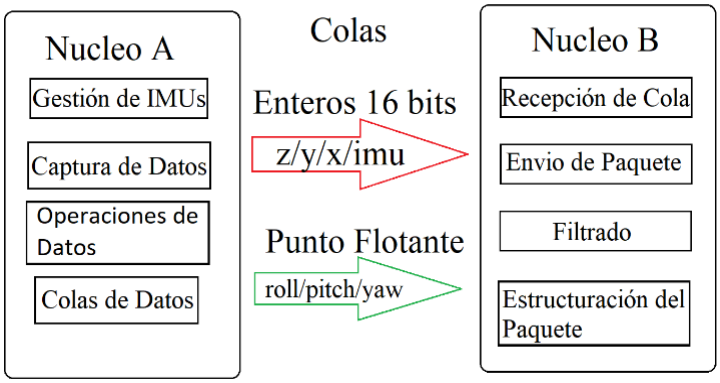

Figura 15. Programa doble núcleo.

Esta estructuración está basada en tres pasos:

Un primer paso realizando la separación de funciones, separando por un lado la gestión de IMUs, la captura de datos y la transformación de los mismos de la estructuración y envío de paquetes.

Un segundo paso correspondiente a la interconexión entre núcleos, realizada mediante la implementación de dos colas, una para datos enteros y otra para punto flotante para poder transmitir todos los datos sin perder precisión.

Finalmente, un paso en el que se implementarían mejoras al código, como puede ser el Filtrado de los valores de aceleración, en forma de un filtro complementario para solventar los problemas de deriva que tienen los giroscopios, así como varias optimizaciones de algunas de las rutinas iniciales.

Esto conllevó un reparto relativo en las cargas computacionales de ambos programas en sus respectivos núcleos. El reparto o factor de mejora exacto no ha sido posible de obtener al no disponer de herramientas específicas que permitan hacerlo de una manera simple, no obstante, como detalle, indicar que se pudo llegar a capturar señales a una frecuencia de muestreo superior de 80 Hz, aunque dicho cambio se descartó, al determinar que la frecuencia original de 50 Hz, en teoría, sería suficiente para el reconocimiento de los gestos y que se podría emplear ese tiempo de ejecución ganado para implementar posibles mejoras en las funcionalidades del sistema de captura.

En lo relacionado a los dos programas principales, su funcionamiento viene dirigido y controlado por el flujo de datos en las colas, en lo que denominaríamos procesamiento on-demand, siendo que el Núcleo A entraría en Reposo en el momento que las colas estén llenas y el Núcleo B entraría en reposo mientras las colas estén vacías, lo que permite dejar tiempo a las rutinas propias del RTOS para ejecutarse.

3.3 Comunicación con el sistema huésped

La comunicación con el sistema huésped se ha planteado mediante el empleo del protocolo UDP, empleando como medio el protocolo IEEE 802.11, es decir, mediante conexión inalámbrica.

El modelo del sistema de captura de datos presentaba de manera inicial un modelo de paquete de tamaño variable en el que los datos se enviaban como caracteres formando mensajes de tamaño variable.

En nuestro caso, se observó que dicha estructuración no resultaba óptima para la recepción en el sistema huésped, al generar un sobre cómputo innecesario, por lo que se decidió convertir dicha estructura de paquete variable por caracteres a una estructura de tamaño fijo en la que enviamos directamente la información del dato en cuestión, aunque manteniendo ciertas partes de la estructura para permitir y facilitar la posible implementación del sistema en otros dispositivos. Al mismo tiempo, se realizaron las modificaciones necesarias para incluir la información extraída de los giroscopios.

La nueva estructura, podría resumirse en la siguiente tabla:

Dato	***"L"***	***Imu***	***Time***	***ax***	***ay***	***az***	***yell***	***pitch***	***roll***	***nIter***	***"\n"***	***TOTAL***
bytes	1	1	4	2	2	2	4	4	4	4	1	29
									nImus		6	174 bytes

Tabla 2. Estructura de Paquetes.

En la misma, inicialmente se envía un carácter de control "L", para después enviar una serie de datos relevantes, relacionados con el sensor del que se ha extraido los datos (IMU), el tiempo aproximado en el que se ha iniciado la extracción (Time), los datos de aceleración de dicho IMU (ax, ay, az), los datos de posición angular (Yell, Pitch, Roll), el número de iteración actual en el sistema de captura y finalmente un carácter de Fin de datos para cada segmento relativo a un IMU ("\n"). Esto aporta un total de 29 bytes de información al paquete por cada una de las IMUs que se analicen, y, en nuestro caso al ser 6 el número de las mismas, resultan en un total de 174 bytes de información dentro del paquete, más una serie de bytes, aproximadamente 42, derivados del envío con el protocolo UDP.

3.4 Sistema Huésped

Como ya se ha explicado, durante el desarrollo del proyecto, el sistema huésped ha sido el propio sistema de desarrollo. En este apartado, abarcaremos la cadena de eventos desde la recepción de las señales por el sistema huésped hasta su utilización final.

3.4.1 Definición y Entrenamiento de la Red

Como modelo básico de nuestra red, se decidió emplear las ya definidas redes LSTM, en particular las BiLSTM. Puesto que el número de salidas de la red LSTM será demasiado grande como para procesar directamente la información, posteriormente se empleará una capa fully-connected para reducir el número de señales útiles al número de gestos diferentes que queramos reconocer. Tras esto, se empleará una capa softmax para reconocer la señal dominante, y posteriormente etiquetarla con una capa de clasificación.

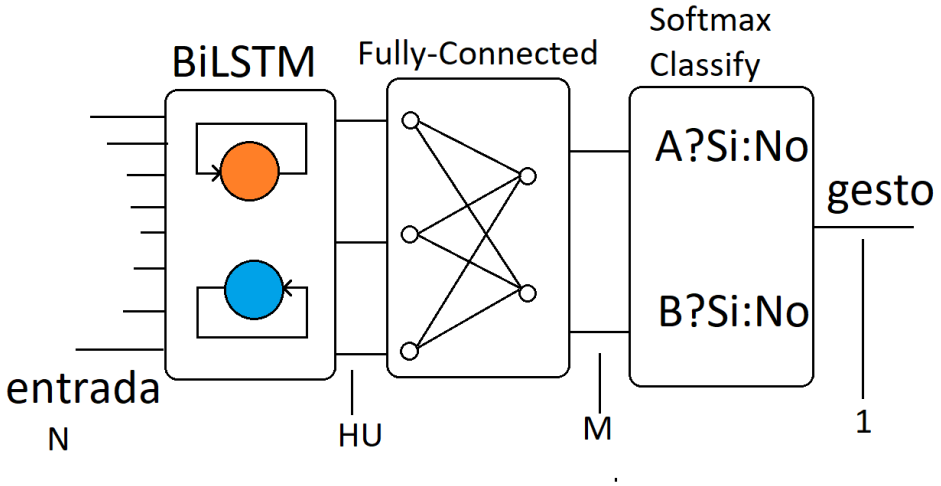

Figura 16. Estructura de la NN.

En el esquema anterior (15), N representa el número de datos de entrada a la etapa BiLSTM. Internamente la etapa *BiLSTM* operará con una cantidad igual de 2*N datos, de los cuales extraerá HU, el número de unidades ocultas, muestras hacia la etapa fully-connected, en la cual, mediante procesos de convolución simples, reducirá a un número M de posibles salidas, representando M la cantidad de posibles gestos a reconocer. Finalmente, mediante una etapa de clasificación se evaluarán la M salidas y se obtendrá una única salida representando una etiqueta única para cada gesto por cada paso temporal.

En lo relacionado al entrenamiento de la red, se ha empleado un entrenamiento en batch por GPU. La capacidad de las GPU para realizar varias sesiones de entrenamiento de manera paralela permite una aceleración de las capacidades de entrenamiento con relación a la propia CPU del sistema.

No obstante, debido a los tiempos de transferencia y creación del entorno en GPU que produce un sobrecoste temporal debido al modelo de funcionamiento de MATLAB, la ejecución de la red una vez entrenada será realizado en CPU. Esto no quiere decir que no sea posible una implementación pura en GPU capaz de realizar los cálculos de manera puramente paralela, si se crease un núcleo que permita mantener el entorno de manera permanente, no obstante, puesto que la mayoría de dispositivos se basan en un esquema de computación en CPU secuencial, se ha optado por mantener la compatibilidad con los mismos.

En lo relacionado a las señales, tras la recepción de las mismas en el sistema, en nuestro caso son señales crudas, sin tratamiento previo obteniendo un conjunto de 36 señales, correspondiente a las aceleraciones relativas en x, y z de los acelerómetros seis y las posiciones yaw, pitch y roll angulares relativas de los seis giroscopios. Dichas señales, antes de alimentarlas a la red, deberán ser como mínimo normalizadas para así evitar problemas de inestabilidad en los pesos de las redes.

Las señales de aceleración serán las empleadas como entradas de la red neuronal, mientras que las de posición angular se reservarán para clasificación de señales.

Para el entrenamiento, se ha optado por obtener dos categorías diferentes de señales:

- Señales de Entrenamiento: Muestras que incluyen varias repeticiones de un ÚNICO gesto.

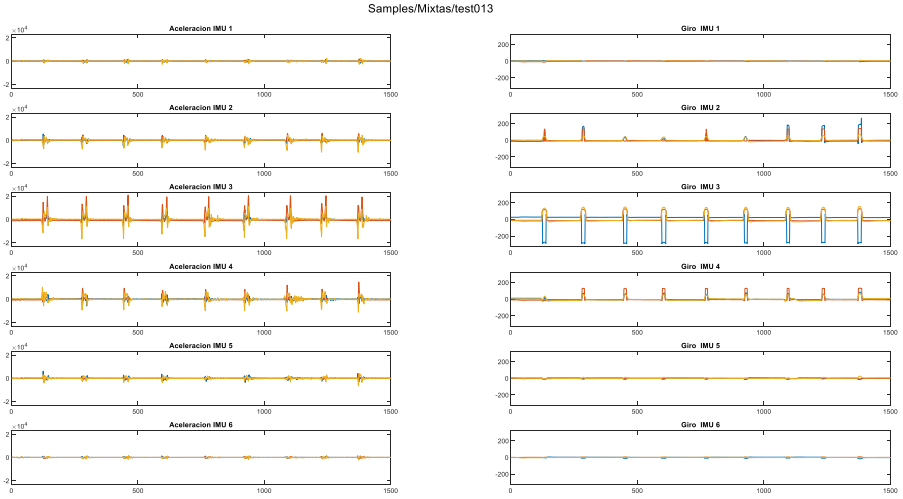

Figura 17. Muestras de Entrenamiento

- Señales de Verificación: Muestras que incluyen una secuencia predefinida de TODOS los gestos realizados.

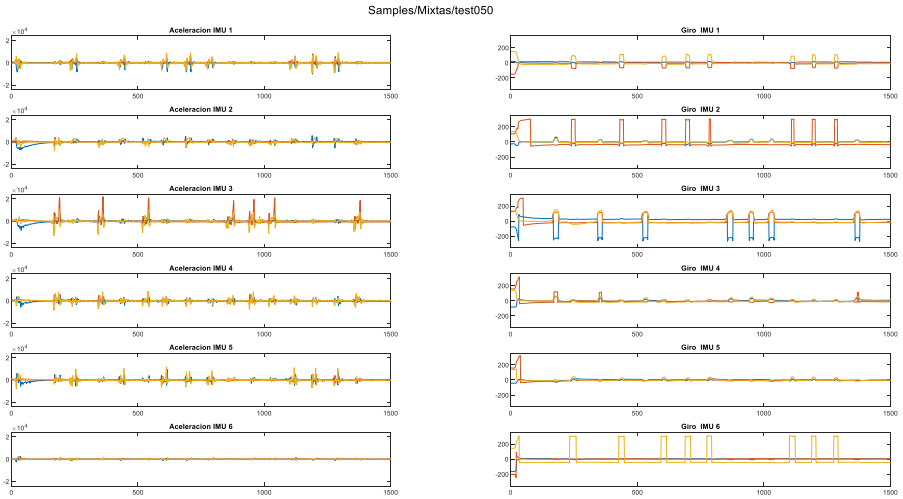

Figura 18. Muestras de Verificación.

En el caso de las primeras, es posible asignar una clasificación previa, puesto que se conoce el gesto realizado, siempre que sea posible de detectar de manera aproximada los instantes de activación deseados para la ejecución del gesto. En nuestro caso, hemos empleado las señales del giroscopio como punto de partida para detectar la gesticulación, empleando la siguiente ecuación:

$$F_t = \begin{cases} gesto\ si\ \frac{|G_{xt}|+|G_{yt}|+|G_{zt}|}{3} > Umbral \\ \text{'Null'} \qquad\qquad en\ otro\ caso \end{cases} \tag{4.1}$$

Siendo $G_{xt}, G_{yt}\ y\ G_{zt}$ las señales de pitch, yaw y roll en un paso temporal t.

A continuación, se muestra un ejemplo de detección:

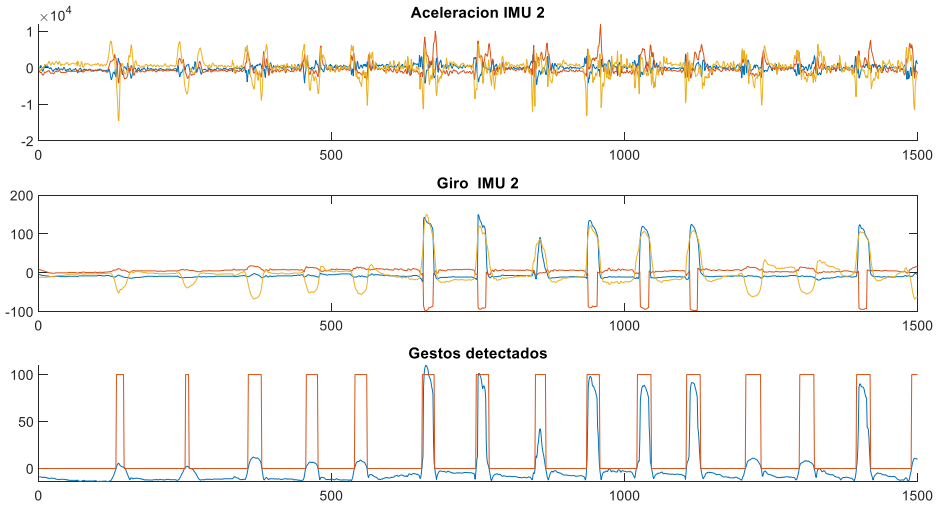

Figura 19. Detección de Gestos.

El conjunto de las señales de entrenamiento junto a esta categorización de las mismas es lo que se conoce como verdad de referencia (ground truth) y constituye la base fundamental para el desarrollo funcional de la red.

Para las segundas, las muestras de verificación, al ser una combinación de gestos su categorización automática resulta demasiado para ser planteada; sin embargo, al ser una muestra limitada de gestos predefinidos, dicha categorización es posible y se realizará mediante la creación de secuencias fijas de gestos que sean reconocibles de cara a una verificación visual.

Para mantener la coherencia en el entrenamiento, se ha decidido no emplear las muestras de verificación para el mismo, estando reservadas únicamente para comprobar que las redes desarrolladas consiguen seguir el funcionamiento deseado. En su lugar, se ha empleado una secuencia de gestos correspondientes a la siguiente alternancia entre gestos:

2 – 1 – 2 – 1 – 2 – 1 – 1 – 1 – 2 – 2 – 2 – 1 – 1 – 1 – 2

Siendo esta la secuencia básica de verificación que se empleara a lo largo de la memoria por simpleza.

Una vez aclaradas las diferencias entre los conjuntos de señales con los que tratamos, tras realizar un entrenamiento, podemos verificar el funcionamiento básico de la red, obteniendo algo similar a la siguiente figura.

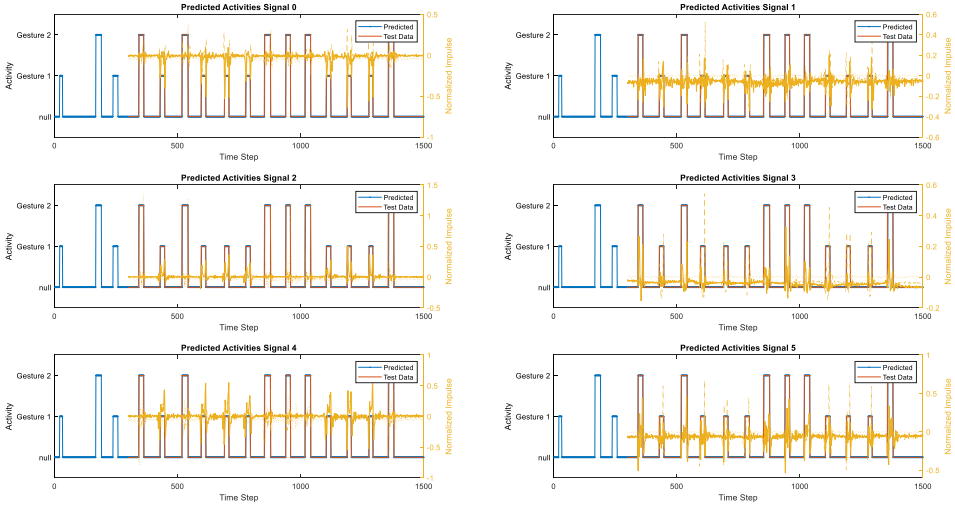

Figura 20. Resultado del Entrenamiento.

A continuación, se presentará una ampliación de una de las graficas de la figura anterior.

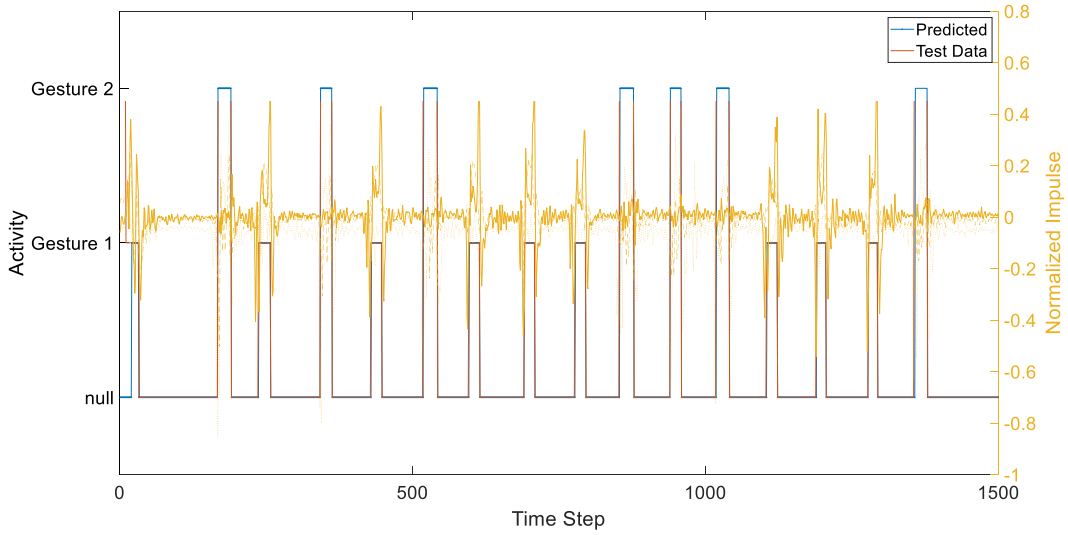

Figura 21. Zoom de la Figura 18.

A primera vista, se aprecia que la red es capaz de reconocer los gestos y categorizarlos, así como de reconocer aparentemente los periodos en los que no hay gesto.

No obstante, esta muestra es de una secuencia pregrabada. Si realizamos un análisis introduciendo las muestras tal y como se tomarían en un contexto de tiempo real, es decir, introduciendo las muestras de una en una en lo que denominaríamos un contexto de pseudo tiempo real, encontramos un fenómeno que no se aprecia en la ejecución y análisis de la secuencia en bloque, una serie de cambios en el reconocimiento de la señal debido al cambio de contexto.

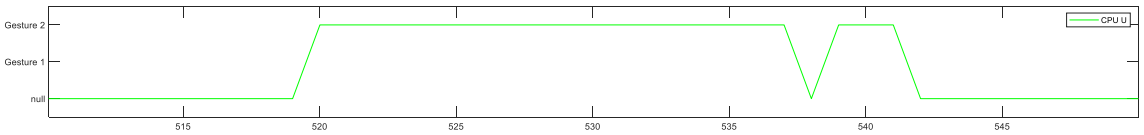

Figura 22. Muestra con fallo de detección.

Si bien estos "fallos" de reconocimiento pueden ser problemáticos, en un principio deberían ser controlables mediante algún tipo de técnica de post-procesamiento.

3.4.2 Post-Procesado de la señal de salida

Como ya se ha introducido, al realizar la introducción secuencial de las muestras a la red, aparecían una serie de "glitches" en el reconocimiento de los gestos.

Como medio para solucionar este comportamiento, se ha optado por realizar un post-procesamiento de la señal de salida de la red. Dicho procesamiento, se ha decidido que sea una comprobación de estabilidad de señal durante tres ciclos de ejecución de la red. Mientras que la señal no se mantenga estable durante tres ciclos de ejecución, se devolverá el ultimo estado estable conocido. A continuación, se plantea un ejemplo de dicho post procesamiento.

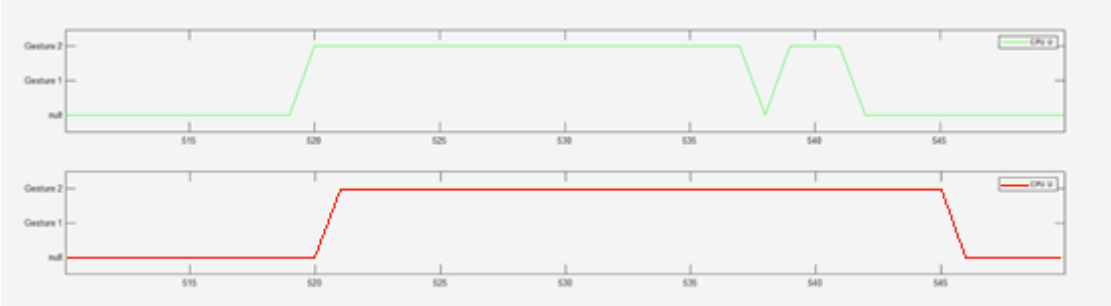

Figura 23. Fallo solucionado con post-procesado.

En este caso, se logra solventar el cambio repentino en el reconocimiento dado entre los pasos temporales 535 y 540. No obstante, este modelo plantea un pequeño retardo en el reconocimiento inicial de gesto de un ciclo y un retardo de pulso de aproximadamente tres ciclos con respecto el cambio a no-gesto de la señal original, o, en otras palabras, introducimos un retardo de entre 20 y 60 ms al sistema, que en principio no afectará a la apreciación de gestos en tiempo real.

Así, tras comprobar que no vamos a encontrar cambios repentinos en la señal, podemos indicar que nuestra señal de salida está preparada para alimentarla a una posible aplicación y comprobar su funcionamiento.

3.4.3 Aplicación.

Para probar las capacidades de la red es necesario de algún tipo de contexto para realizar los testeos pertinentes.

Dos de las funcionalidades que se han planteado, y que han resultado más atractivas, han sido: por un lado, el control de los modos de una cámara y, por otro, la implementación de un mapa que se pueda controlar por gesticulación

Para el caso de la primera, el control de la cámara, el funcionamiento propuesto es permitir controlar los diferentes modos de una cámara, así como su activación/desactivación, mediante un

menú contextual. Dicho menú deberá como mínimo permitir navegar entre las diferentes opciones (2 gestos) y seleccionarlas (1 gesto).

Esto hace que como mínimo para esta funcionalidad sea necesario la implementación de un catálogo de 3 gestos diferenciables. Esta funcionalidad se ha modelado como la posibilidad de elegir entre 3 modos de una posible cámara (Normal, Térmica y Nocturna) así como el apagado de la misma.

La cámara empleada para la captura de imágenes en tiempo real ha sido una PS-Eye modelo SLEH-00448, capturando a 30 frames por segundo. Puesto que la cámara no dispone de capacidades de trabajar de manera nativa con diferentes modos de funcionamiento, se ha modelado dicho funcionamiento mediante la implementación de filtros y/o mapas de color.

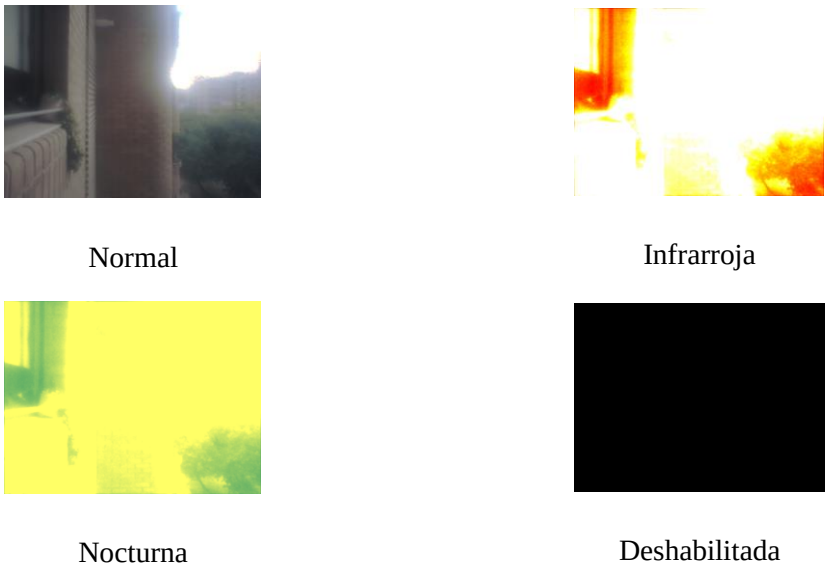

Figura 24. Ejemplos de funcionamiento para la cámara.

Por otro lado, para el caso de la segunda, el mapa, el funcionamiento propuesto es permitir navegar por un determinado mapa; por ello, es necesario permitir moverse por el mapa en mínimo 2 direcciones discretas (4 gestos), y la posibilidad de realizar la acción de acercar y alejar el mapa (2 gestos). Esto conlleva un catálogo mínimo de 6 gestos diferenciables.

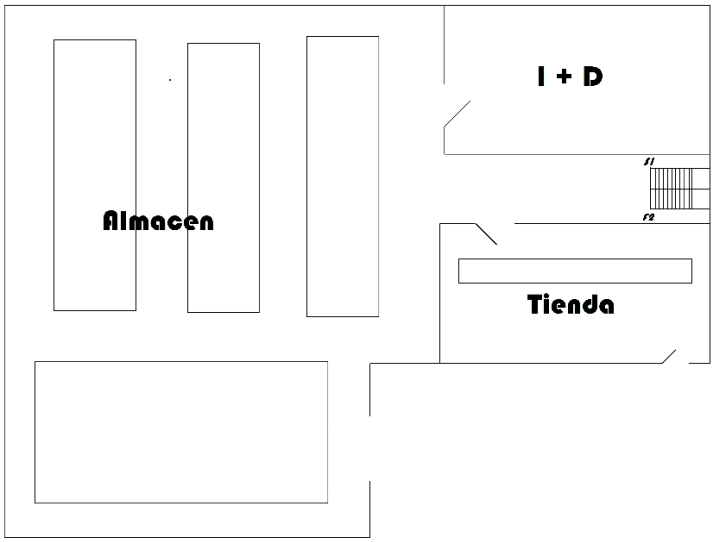

Figura 25. Ejemplo de mapa.

Adicionalmente, puesto que ambas funcionalidades deberán convivir en un mismo sistema, junto posiblemente otras funcionalidades, será necesario implementar los mecanismos necesarios para poder permitir el cambio de contexto entre ellas. Esto conllevaría a un catálogo de gestos que incluyese al menos 4 gestos para la selección de las diferentes funcionalidades.

El siguiente esquema, representa una posible implementación de las funcionalidades junto a los controles ligados a cada una de ellas.

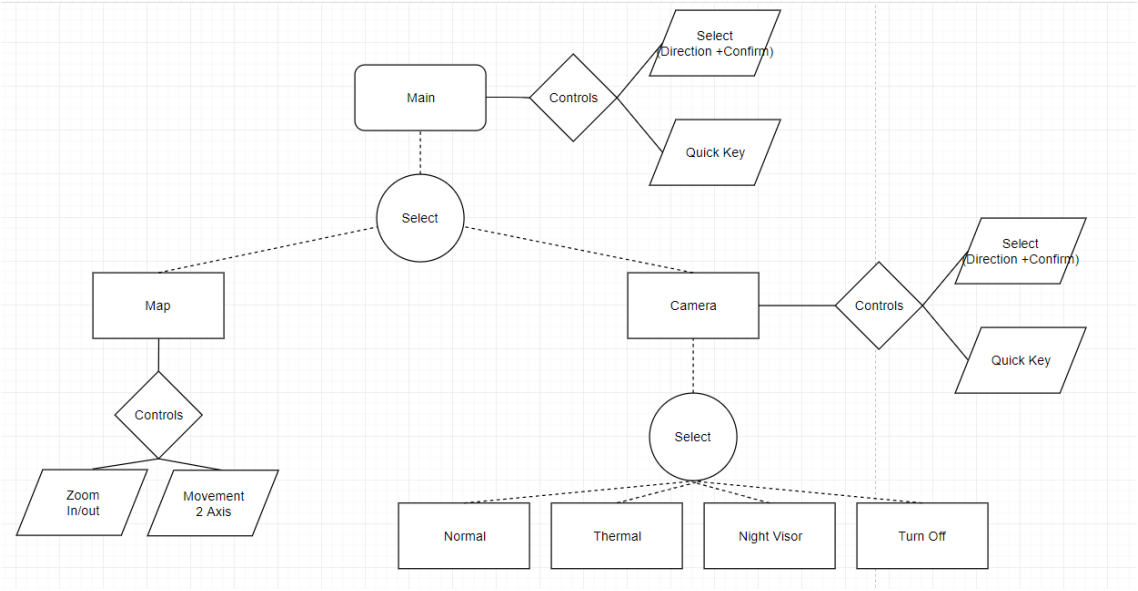

Figura 26. Esquema de implementación

Así, el catálogo mínimo que se ha determinado para un correcto funcionamiento de la aplicación es de 7 gestos, tal y como se recoge en la siguiente tabla.

Gesto (Etiqueta)	Menu/Camara (Accion)	Mapa (Accion)	Gesto (Accion)
UP	Arriba	Arriba	Movimiento Arriba
DOWN	Abajo	Abajo	Movimiento Abajo
LEFT	-	Izquierda	Movimiento Izquierda
RIGHT	-	Derecha	Movimiento Derecha
SELECT	Seleccionar	Zoom Out	Flexion Pulgar
AUX	-	Zoom In	Giro de Muñeca
BACK	Salir	Salir	Flexion Meñique

Tabla 3. Definición de Gestos.

Capítulo 4. Resultados

4.1 Comprobación funcional en tiempo real.

Tras los ajustes realizados a la red y la creación de un contexto de pruebas, el siguiente paso lógico es comprobar que el sistema es capaz de funcionar verdaderamente en tiempo real. Si bien para este caso es difícil plasmar los resultados de las pruebas realizadas, en este apartado se procurará detallar dichos resultados, así como facilitar una serie de contenidos audiovisuales para demostrar el funcionamiento del proyecto.

El primer detalle no destacado con anterioridad es que, en un contexto real, vamos a captar una serie de movimientos que posiblemente no deseemos que sean capturados y/o procesados. En nuestro caso y como funcionamiento auxiliar, se ha configurado para que mediante la pulsación de una tecla en un teclado se deshabilite el procesamiento de los gestos por parte de la red, congelando de manera relativa el contexto del programa aplicación. Esta funcionalidad, sería posible su implementación de manera trivial en un posible sistema final mediante la inclusión de algún tipo de sensor RFID y una etiqueta RFID, que habiliten o deshabiliten la captura o procesado de las señales.

Un segundo detalle, es la necesidad de una tasa de funcionamiento del programa aplicación similar o superior a la tasa de recepción de señales. El principal motivo de esto es para asegurar una fluidez decente en el proceso de gestión de gestos en relación a la percepción del usuario observado durante la creación del programa para la aplicación. Así, se ha fijado un objetivo de funcionamiento de 60 ciclos por segundo para el mismo.

Una vez aclarados estos detalles, para nuestra prueba, lo que se va a intentar es manejar las funcionalidades de la aplicación mediante la captura y procesado directo de las señales recibidas. Dichas funcionalidades pueden ser comprobadas en el siguiente enlace (link a YouTube):

URL: https://www.youtube.com/watch?v=w2KTubF-cvY

En el mismo primeramente se entra al modo cámara y se transita por los diferentes modos. Posteriormente se entra en el modo mapa para moverse por el mismo, mostrando además las funcionalidades del zoom y mostrando en conjunto la aplicación de los siete gestos.

4.2 Análisis paramétrico de la red.

Uno de los puntos destacados anteriormente para la implementación del sistema es la posibilidad de emplear microcontroladores, procesadores de bajo consumo o incluso arquitecturas paralelas como base del sistema huésped de computación, o al menos del procesado de la red neuronal.

Dicha posibilidad dependerá en buena medida de la capacidad del mismo para ser capaz de ejecutar de manera efectiva el contexto creado por la red neuronal en una parte del tiempo disponible entre muestras, que estará ligado a su vez del coste computacional de la propia red.

Si nos basamos puramente en la red, dicho tiempo puede ser influido por diferentes parámetros, como el tamaño del catálogo de gestos, el dimensionamiento de la red o la estructuración de la misma. En este apartado, comprobaremos dichas variaciones, así, como intentar obtener algún tipo de límite correlacional entre dichos parámetros.

Tipo	Ecuación de cantidad de datos
Pesos BiLSTM	$N_{weight} = (2 * 4 * N_{HU}) * (N_{input} + 1)$
Bias BiLSTM	$N_{Bias} = (2 * 4 * N_{HU})$
Pesos fully-con	$N_{weight} = (N_{outputs}) * (2 * N_{input_{fc}})$
Bias fully-con	$N_{Bias} = N_{outputs}$

Tabla 4. Ecuaciones de tamaño de Red.

Para estos apartados tomaremos dos objetivos para determinar si una configuración resultaría efectiva para su funcionamiento:

- El tiempo de procesado entra dentro de los límites necesarios (t < 15 ms).
- No aparecen errores durante la secuencia de verificación en "tiempo real".

El primero servirá para verificar que efectivamente la configuración es capaz de funcionar correctamente procesando todas las muestras que se reciben, mientras que el segundo nos servirá para verificar que la red es capaz de funcionar de la manera esperada.

En el caso de los tiempos de procesado, se dará la media obtenida de la ejecución de una única muestra, que incluirá los tiempos de transferencia, setup del entorno y calculo, así como del tiempo medio de un conjunto de muestras intentando eliminar los tiempos de transferencia y setup mediante la ejecución de dos conjuntos, uno de 1500 y otro de 3000 muestras.

Para la verificación de errores, se ha definido un criterio simple. Se ha realizado hasta un máximo de 5 entrenamientos para cada configuración. Si al menos uno de esos 5 entrenamientos se ha considerado correcto o posible de corregir por post-procesado, se dará la configuración por correcta. Si los 5 entrenamientos han dado errores no recuperables o difíciles de recuperar, se tomará la configuración como fallida.

4.2.1 Variación del número de celdas ocultas

Para este apartado, se va a realizar una variación en la complejidad de la red aplicando una variación al número de Unidades Ocultas (HU). Esto, nos dará un indicativo de la evolución de la red en cuanto a capacidad de cómputo, dimensionamiento y tiempo de cómputo.

Tipo	HU	Pesos	Bias	¿Fallos?	Env	$t_{muestra}$ (1 muestra)	$t_{muestra}$ (1.500 muestras)
Base	200	350.000	1.603	No	CPU	3.4 ms	110 us
					GPU	23.2 ms	17 us
	100	95.000	803	No	CPU	3.9 ms	104 us
					GPU	26.9 ms	17 us
	75	56.250	603	No*	CPU	2.7 ms	84 us
					GPU	24.66 ms	17 us
	50	27.500	403	Si	CPU	-	-
					GPU	-	-

Tabla 5. Comparación variación del número de HU.

En este caso de estudio, encontramos una barrera de reducción en las 75 Unidades Ocultas, siendo que al reducir el número de las mismas a 50 no ha sido posible hallar una red funcional sin errores reparables.

Mientras que el tiempo de procesamiento en CPU refleja una clara reducción proporcional al tamaño de la red, en el caso de GPU se obtiene un tiempo estable, al disponer de una capacidad de cómputo por encima de la necesaria. Como ya se adelantaba, se obtiene un sobretiempo superior en GPU con respecto a CPU relacionado con la creación del entorno y la transferencia bruta de datos que bajo el modelo actual por la cual resultaría inviable de manera aparente realizar la implementación en GPU. Sin embargo, eliminando dicho sobretiempo, la implementación GPU resulta mucho mas rápida, alcanzando los 17 us de tiempo de procesado por muestra.

Aclarar que, aunque el caso de 75 Unidades Ocultas se ha considerado exitoso, la frecuencia de errores base reparables es mayor que en el resto de casos, requiriendo de un extra en la etapa de post-procesado, afectando de manera sensible a la estabilidad de reconocimiento y a la latencia del sistema.

Concluimos que, reducir la cantidad de unidades ocultas permitirá realizar redes sensiblemente mejores en cuanto a tiempos de cómputo, pero a costa de la estabilidad de la red, siendo necesario emplear más recursos en la etapa de post-procesado para obtener una salida deseable.

4.2.2 Variación del tamaño del catálogo

En esta ocasión, modificaremos la cantidad de gestos del catálogo para analizar cómo afecta a la complejidad requerida para su completa detección. Para ello, se va a realizar un aumento progresivo del número de gestos en el catálogo y una variación de las Unidades ocultas tal y como se recoge en la siguiente tabla junto a los resultados

Cat	HU	Pesos	Bias	¿Fallos?	Env	$t_{muestra}$ (1 muestra)	$t_{muestra}$ (1.500 muestras)
3	200	350.000	1.603	No	CPU	3.4 ms	110 us
					GPU	23.2 ms	17 us
	100	95.000	803	No	CPU	3.9 ms	104 us
					GPU	26.9 ms	17 us
5	200	350.800	1.605	No	CPU	3.7 ms	165 us
					GPU	24.3 ms	18 us
	100	95.400	805	No	CPU	3.6 ms	101 us
					GPU	23.8 ms	18 us
7	200	351.600	1.607	No	CPU	3.8 ms	192 us
					GPU	23.7 ms	18 us
	100	95.800	807	Si	CPU	-	-
					GPU	-	-

Tabla 6. Comparación variación tamaño del catálogo de gestos.

En este caso, hallamos una barrera en el caso de siete gestos con 100 unidades ocultas. La complejidad de la red en este caso resulta aparentemente insuficiente como para llegar a funcionar correctamente.

El numero de gestos a detectar, parece afectar de manera sensible al tiempo de procesado por muestra en CPU, viéndose incrementado conforme aumenta el número de gestos.

Finalmente, el tamaño de la red sufre un aumento mínimo, siendo de 2*HU*N_new + N_new la cantidad de datos adicional a considerar en el dimensionamiento de memoria.

4.2.3 Variación de la estructura de la red

Para este apartado, lo que se pretende comparar es lo que consideraríamos nuestra estructura actual, en adelante en este apartado denominada Base, con una modificación de la misma que permita reducir de manera teórica el tamaño general de la red, en adelante Reducida.

Dicha modificación, se ha realizado mediante lo que llamaríamos una capa de reducción fully-connected que permita reducir en este caso el número de entradas de la red BiLSTM. Puesto que el tamaño de la red, así como su dimensionamiento en memoria, está ampliamente ligado al número de entradas a procesar (X), y al hecho de que al entrar en la red BiLSTM las entradas originales ya llevarán anexado algún grado de procesamiento de la información, lo que debería permitir de manera teórica reducir el número de celdas ocultas y, por ende, reducir la carga computacional de la misma, a coste de aumentar la complejidad de la red de cara a su implementación en otros dispositivos.

Así, las estructuras de ambos modelos son las recogidas en la siguiente tabla:

Base	***Reducida***
Entrada (X)	Entrada (X)
BiLSTM (X x N)	Fully Connected (X x M)
Fully_connected (N x Y)	BiLSTM (M x N)
Softmax	Fully Connected (N x Y)
classify	Softmax
-	classify

Tabla 7. Estructuración de redes.

Inicialmente, para facilitar la realización de este experimento, se ha realizado con un pequeño catálogo de 3 gestos(Y), además de un tamaño inicial de 18 señales de entrada (X) y para el caso de la red reducida, realizando una variación en el número de entradas a la BiLSTM (M) y el número de células ocultas para analizar cómo afecta al tamaño y al tiempo de procesado.

A continuación, se presentan los resultados del entrenamiento de diferentes redes, con variación de parámetros de las mismas en forma de tabla.

Tipo	X/M	HU	Pesos	Bias	¿Fallos?	Env	$t_{muestra}$ (1 muestra)	$t_{muestra}$ (1.500 muestras)
Base	-	200	350.000	1.603	No	CPU	3.4 ms	110 us
						GPU	23.2 ms	17 us
		100	95.000	803	No	CPU	3.9 ms	104 us
						GPU	26.9 ms	17 us
Reducida	18/12	200	340.616	1.615	No	CPU	3.9 ms	151 us
						GPU	26.7 ms	17 us
		100	91.116	815	No	CPU	4 ms	110 us
						GPU	28.6 ms	19 us
		75	54.264	615	No	CPU	4 ms	91 us
						GPU	28.2 ms	19 us
		50	21.316	415	Si	CPU	-	-
						GPU	-	-
	18/9	200	336.862	1.612	No	CPU	4.4 ms	147 us
						GPU	24.3 ms	20 us
		100	87.962	812	No	CPU	4 ms	126 us
						GPU	27.8 ms	19 us
		50	24.062	412	No	CPU	2.5 ms	66 us
						GPU	23.3 ms	19 us
	18/6	200	330.908	1.609	No	CPU	4 ms	156 us
						GPU	23.7 ms	17 us
		100	85.508	809	No	CPU	3.4 ms	106 us
						GPU	23.6 ms	20 us
		50	22.808	409	Si*	CPU	-	-
						GPU	-	-
	18/3	200	326.054	1.606	No	CPU	3.8 ms	181 us
						GPU	24.1 ms	18 us
		100	83.054	806	No	CPU	4.1 ms	151 us
						GPU	25.7 ms	18 us
		50	21.554	406	Si	CPU	-	-
						GPU	-	-

Tabla 8. Comparación variación de la complejidad de la red.

En este caso no parece haber una correlación directa entre la complejidad de la red, dada por X/M y HU, con la posibilidad de aparición de fallos. Podemos encontrar un caso optimo en una reducción 18/9 y 50 unidades ocultas, encontrando una reducción en este caso de memoria de aproximadamente 15 veces menor, con una reducción de entorno a un 40% en el tiempo de procesado. Otro caso a destacar es el de la configuración 18/6 con 50 unidades ocultas, que si bien se logro una muestra que podría considerarse recuperable, se ha considerado que la cantidad de post-procesamiento necesaria es demasiado elevada como para resultar verdaderamente aceptable.

Se puede observar una reducción sensible en el dimensionamiento en memoria de la red conforme el índice de compresión X/M aumenta. Por otro lado, el tiempo de procesamiento a igualdad de unidades ocultas se ve claramente aumentado al aumentar la compresión de señales.

Para finalizar este apartado, a continuación, se presentan algunos tipos de redes encontradas que se podrían considerar recuperables por post-procesado.

Figura 27. Respuesta con saltos entre niveles reparables

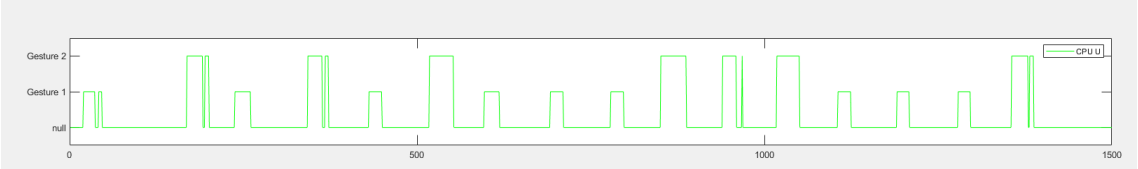

Figura 28. Respuesta con “caídas” a “No gesto” reparables.

Estas muestras representan lo que se considera una traza recuperable. Los errores que aparecen en la muestra podrían ser solventados mediante el post-procesamiento. En estos casos, el fallo de detección se produce o bien por transiciones “prohibidas” entre gestos o por vueltas indeseadas al nivel de no gesticulación, ambos solucionables con un bajo impacto sobre la latencia.

Las siguientes muestras son lo que consideraríamos muestras de redes no válidas, al no ser posible su recuperación por post-procesado, o en caso de serlo, la cantidad de recursos empleada para hacerlo resultaría inviable.

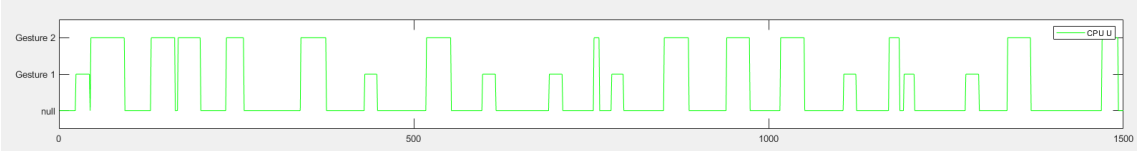

Figura 29. Respuesta con detecciones incorrectas.

En este caso, se producen detecciones de gesto en puntos en los que no deberían de aparecer. Esto claramente es una red no reparable, al no poder diferenciar las activaciones reales de las ficticias.

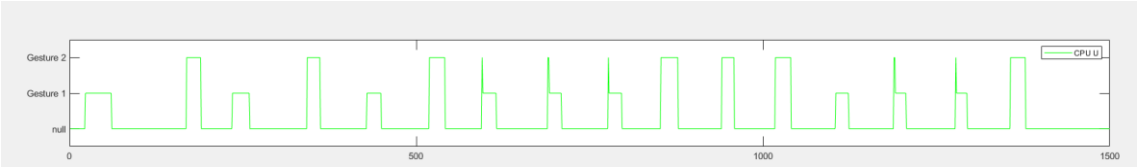

Figura 30. Respuesta con reconocimientos no reparables.

Para esta muestra, si bien la forma general es adecuada, no produciéndose detecciones fuera de lugar, la incitación inicial de algunos de los gestos es incorrecta, produciéndose una detección del Gesto 2 en algunas ocurrencias del Gesto 1 de manera inicial. Si bien esto podría ser recuperable por post-procesado, esto afectaría de manera severa al comportamiento de tiempo-real con baja latencia del sistema, empeorando la sensación de uso del mismo.

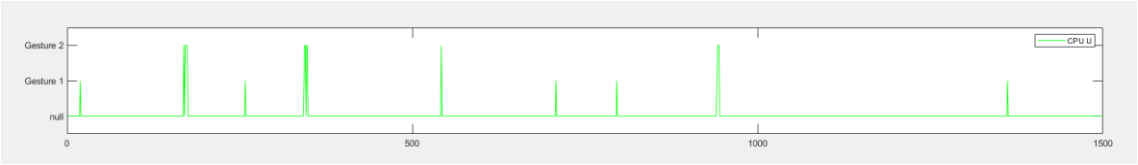

Figura 31. Respuesta red inconsistente

Finalmente, la figura anterior muestra una red inconsistente, en la que no solo no se detectan todas las activaciones de gesto, sino que las que sí detecta ven su ancho de pulso reducido.

Capítulo 5. Conclusiones y Trabajos Futuros

Tal y como se ha demostrado a lo largo de esta memoria, el proyecto es posible de desarrollar, adaptar y utilizar para el que será su cometido, la acción inmediata, siendo el caso que se ha conseguido desarrollar un sistema con un catálogo de gestos lo suficientemente rico como para manejar varias posibles aplicaciones que puedan ser desarrolladas.

Si bien el hardware empleado para la realización de las pruebas del sistema no se adapta completamente a la necesidad del dispositivo ideal que se esperaría en emplear como sistema huésped, los resultados son prometedores.

Por otro lado, contando con las especificaciones del actual sistema huésped y, el hecho de que actualmente estamos con un entorno no adaptado, a la vista de los resultados de rendimiento obtenidos, se refuerza la teoría de que una implementación en otros dispositivos de menor potencia podría llegar a ser posible con las técnicas de optimización oportunas.

Además, el estudio del comportamiento de la red, ha permitido obtener información que puede resultar importante a la hora de adaptar el sistema a otro tipo de hardware, como es el caso de la implementación de la capa de compresión mediante la cual ha sido posible reducir en un factor de 15 la cantidad de datos propios de la red, sin un coste real en el tiempo de ejecución para un pequeño catálogo básico.

Para finalizar este apartado y la memoria, existen una serie de proyectos o líneas de trabajo futuras que se han ido planteando a lo largo del desarrollo del Trabajo de Fin de Master, los cuales serán enunciados y explicados a continuación.

- *Estudio detallado de Sistema Huésped*. Se considera que esta línea de trabajo podría llegar a ser importante a la hora de decidir las especificaciones del sistema final. No todos los sistemas huésped se comportan de la misma manera y puede darse el caso que algunos se adapten en mayor medida a los requisitos del sistema final.
- *Viabilidad de la integración en un dispositivo único.* Otra línea de trabajo relacionada con la anterior. Una posibilidad que se planteó fue el realizar la captura de datos y procesamiento mediante el sistema de captación de gestos. Actualmente dicha implementación se ha descartado por imposible puesto que el ESP32 empleado no tiene suficiente potencia para manejarlo todo, sin embargo, no se ha descartado la posibilidad de que exista un dispositivo que si permita dicha implementación, lo que permitiría eliminar un sistema intermediario y potencialmente podría acelerar el rendimiento del sistema.
- *Viabilidad del proyecto más allá del prototipado*. Otro de los puntos que se considera como interesantes y no se han desarrollado en esta memoria con profundidad, asumiendo en la misma que es un producto que puede funcionar fuera del entorno de prototipado, es precisamente esta viabilidad fuera de una etapa de prototipado.
- *Ampliación de funcionalidades.* Puesto que se ha demostrado que el sistema actual funciona, la creación de aplicaciones para el mismo se considera una línea de trabajo interesante, realizando mejoras sobre las ya creadas, como puede ser otorgando interactividad a la función mapa, o creando nuevas, como puede ser el control de un dron para zonas de difícil acceso.
- *Adaptabilidad del bombero al sistema propuesto.* Lamentablemente por la situación en la que estamos (COVID-19), no ha sido posible realizar una parte importante en este tipo de trabajos, trabajar con el usuario final, el cual puede aportar un feedback vital para la adaptación del sistema. Consideramos que este punto puede ser interesante de cara a un futuro posible proyecto relacionado con este.

Capítulo 6. Bibliografía/Webgrafía

[1] Maurits de Graaf et al, *"Cognitive Adaptive Man Machine Interfaces for the Firefighter Commander: Design Framework and Research Methodology",2011*, Springer, LNAI 6780, pp. 588–597, https://link.springer.com/content/pdf/10.1007/978-3-642-21852-1_68.pdf, visitado por última vez 13 de Noviembre de 2020.

[2] A. Naghsh et al, *"remote and in-situ multirobot interaction for firefighters interventions under smoke conditions"*, 2010, IFAC, pp. 109-115, DOI 10.3182/20101005-4-RO-2018.00036

[3]Francesca de Cilles et al , *"On Field Gesture-based Human–Robot Interface for Emergency Responders",2013*,10.1109/SSRR.2013.6719345,https://www.researchgate.net/profile/Francesca_De_Cillis/publication/261155223_On_field_gesture-based_human-robot_interface_for_emergency_responders/links/5798736508aed51475e83c39.pdf , visitado por última vez el 13 de Noviembre de 2020.

[4] Marion G Ceruti et al, "*Wireless communication glove apparatus for motion tracking, gesture recognition, data transmission, and reception in extreme environments*", 2009, DOI 10.1145/1529282.1529320,https://www.researchgate.net/publication/221000424_Wireless_communication_glove_apparatus_for_motion_tracking_gesture_recognition_data_transmission_and_reception_in_extreme_environments , visitado por ultima vez el 13 de Noviembre de 2020.

[5] P. Chitra et al, "*Gloves Gesture Recognition", 2020, IJSTR*, 2277-8616, pp. 418-423, http://www.ijstr.org/final-print/feb2020/Gloves-Gesture-Recognition.pdf, visitado por última vez el 13 de Noviembre de 2020.

[6] Joseph M. Willi et al, "*Characterizing a Firefighter's Immediate Thermal Environment in Live-Fire Training Scenarios*", 2015, DOI 10.1007/s10694-015-0555-1, Fire Technology 52, pp. 1667-1696, https://link.springer.com/article/10.1007/s10694-015-0555-1, visitado por última vez el 13 de Noviembre de 2020.

[7] Patrick L. Walter, "The History of the Accelerometer", 2007, SOUND AND VIBRATION, 41(1):84-92, https://qringtech.com/TryMe/wp-content/uploads/2014/01/HistoryOfTheAccelerometer.pdf, visitado por última vez el 13 de Noviembre de 2020.

[8] Kate Strachnyi, "Brief History of Neural Networks", 2019, Medium, https://medium.com/analytics-vidhya/brief-history-of-neural-networks-44c2bf72eec, visitado por última vez el 14 de Noviembre de 2020.