

Universidad Internacional de La Rioja (UNIR)

ESIT

Máster Universitario en Inteligencia Artificial

Reconocimiento visual en tiempo real de las “señales de rampa” aplicado a sistemas aéreos autónomos para operaciones en tierra.

Trabajo Fin de Máster

Presentado por: de Frutos Carro, Miguel Ángel

Director/a: Sánchez Rodríguez, Antonio José PhD.

Ciudad: Madrid, España.

Fecha: rev 4.0 14-JUL-2020

Resumen

Los vehículos aéreos no tripulados ya son una realidad en nuestros días. El siguiente gran reto tecnológico y operativo al que se enfrenta el sector es el de la convivencia simultánea de plataformas tripuladas y no tripuladas en los espacios actuales existentes. Uno de los mayores retos se presenta en las operaciones terrestres, diseñadas originalmente para interacción entre humanos.

El objetivo principal de este trabajo es el diseño y validación de un software de reconocimiento visual e identificación en tiempo real, de un número limitado de las “señales de rampa” que el personal de tierra, lleva a cabo mediante el movimiento específicos de sus brazos, enfocado para su uso en UAVs u otras plataformas autónomas.

Se hará uso de las técnicas actuales basadas en el uso de redes neuronales pre-entrenadas para la predicción secuencial de la postura de un ser humano y de métodos tipo “ensemble” adicionales para la clasificación de los gestos.

Palabras Clave: Tiempo-real, Reconocimiento de gestos, “Convolutional Pose Machines”, Señales de rampa, “UAVs”

Abstract

Unmanned aerial vehicles are a reality today. The next great technological and operational challenge, that the sector must face is the simultaneous coexistence of manned and unmanned platforms in existing spaces. One of the biggest challenges comes in ground operations, originally designed for human-to-human interaction.

The main objective of this undertaking is the design and validation of a visual recognition and identification software in real time, of a limited number of "airport ramp signals" that ground personnel carry out through the specific movement of their arms, focused for use in UAVs.

With this purpose in mind, techniques based on pre-trained neural networks for the sequential prediction of human pose and additional ensemble methods for the classification of gestures, will be employed.

Keywords: Real-time, Gesture Recognition, Convolutional Pose Machines, Aircraft Marshalling Signals, UAVs.

Índice de contenidos

1. Introducción	1
1.1 Motivación	2
1.2 Planteamiento del trabajo	3
1.3 Supuestos de partida.	8
1.4 Estructura de la memoria.....	9
2. Estado del arte: contexto y propuesta de solución.	10
2.1. Dominio del problema: “Aircraft Marshalling”.	10
2.2. Estado del arte: “Interacción Humano-Robot”	13
2.3. Estado del arte: “Reconocimiento de Gestos”.	15
2.4. Valoraciones finales y propuesta de desarrollo.	21
3. Objetivos y metodología de trabajo.....	27
3.1. Objetivo general.....	27
3.2. Objetivos específicos	27
3.3. Metodología del trabajo	28
3.4. Cronograma de fases y tareas principales.....	29
3.5. Materiales y métodos.....	31
4. Descripción de los requisitos del sistema.	33
4.1. Casos de uso del Sistema.	33
4.2. Requisitos Generales del Sistema.	34
4.3. Arquitectura del sistema.	38
5. Diseño, desarrollo y evaluación de la solución técnica implementada	42
5.1. Descripción de la herramienta software desarrollada.	43
5.2. Generación del Dataset.	46
5.3. Preparación de los datos.	50
5.4. Modelos para la clasificación automática.....	52
5.4.1. Random Forest.....	53

5.4.2. Red Neuronal Artificial: Perceptrón Multicapa.....	57
5.5. Evaluación de la solución en entorno real.....	62
6. Benchmarking: Arquitectura basada en CNN.....	66
6.1. Topología de la red.....	66
6.2. “Data Augmentation Techniques”	67
6.3. Entrenamiento de la red	69
6.4. Evaluación de la predicción en tiempo real	71
6.5. Caso de uso: “Operación Nocturna”	72
7. Conclusiones y trabajo futuro.....	75
7.1. Conclusiones.....	75
7.2. Líneas de trabajo futuro	79
8. Bibliografía	81
Anexos	85
Anexo I. Enlaces al material complementario del proyecto.	85
Anexo II. Artículo de investigación.....	85

Índice de tablas

Tabla 1. Gestos reconocidos en nuestra implementación.....	21
--	----

Índice de figuras

Figura 1. Señalero o “Aircraft Marshalling”. Fuente: Dominio público.....	2
Figura 2. Esquema funcional “alto nivel”. Fuente: Elaboración propia.....	4
Figura 3. Las 5 categorías escogidas. Fuente: Elaboración propia.	6
Figura 4. Funcionamiento red neuronal para CPM. Fuente: Elaboración propia.....	6
Figura 5. “Aircraft Marshaller” en aeropuerto internacional . Fuente: Shutterstock.	10
Figura 6. Colección de señales más empleadas. Fuente: Vectorstock.....	11
Figura 7. Arquitectura CPM usada por OpenPose. Fuente:(Cao et al., 2019).....	19
Figura 8. Salida “CPM” en formato “COCO”. Fuente: Elaboración propia.	20
Figura 9. Arquitectura basada en 3 fases propuesta. <i>Fuente: Elaboración propia</i>	23
Figura 10. Señalero ante un UAV modelo “Boeing MQ-25 Stingray”. <i>Fuente: Boeing</i>	25
Figura 11. Modelo de desarrollo en Espiral. Fuente: ASPgems.....	28
Figura 12. Características principales Jeston Nano. Fuente: NVIDIA®.....	31
Figura 13. Características principales C920HD. Fuente: Logitech®.....	32
Figura 14. Arquitectura HW empelada. Fuente: Elaboración propia.....	38
Figura 15. Arquitectura SW empelada. Fuente: Elaboración propia.	39
Figura 16. Esquema ResNET-18. Fuente: He et al., 2016.....	40
Figura 17. GUI desarrollada en el proyecto. Fuente: He et al., 2016.....	43
<i>Figura 18. Detalle herramienta SW. Fuente: Elaboración propia</i>	44
<i>Figura 19. Detalle herramienta SW. Fuente: Elaboración propia</i>	45
<i>Figura 20. Equipación utilizada durante la captura. Fuente: Elaboración propia</i>	46
<i>Figura 21. Ejemplo información capturada por cada muestra. Fuente: Elaboración propia</i> ...	47
<i>Figura 22. Ejemplo Dataset capturado. Fuente: Elaboración propia</i>	48
<i>Figura 23. Relación “keypoints” y articulaciones . Fuente: Elaboración propia.</i>	49
<i>Figura 24. Ejemplo archivo “CSV” generado . Fuente: Elaboración propia.</i>	49
<i>Figura 25. Generación de etiquetas . Fuente: Elaboración propia</i>	50
<i>Figura 26. Unificación en un único Dataset . Fuente: Elaboración propia.</i>	51

<i>Figura 27. Mezclado aleatorio de datos. Fuente: Elaboración propia.</i>	51
<i>Figura 28. Serialización de los datos. Fuente: Elaboración propia.</i>	52
<i>Figura 29. Hiperparámetros Random Forest. Fuente: Elaboración propia.</i>	54
<i>Figura 30. Ajuste del modelo mediante RF. Fuente: Elaboración propia.</i>	54
<i>Figura 31. Importancia de cada variable en RF. Fuente: Elaboración propia.</i>	55
<i>Figura 32. Matriz de confusión y métricas RF. Fuente: Elaboración propia.</i>	56
<i>Figura 33. Resumen modelo basado en “ANN”. Fuente: Elaboración propia.</i>	59
<i>Figura 34. Resultados del modelo tras entrenamiento. Fuente: Elaboración propia.</i>	60
<i>Figura 35. Resultados comparación 3 modelos. Fuente: Elaboración propia.</i>	61
<i>Figura 36. Matriz de confusión y métricas ANN. Fuente: Elaboración propia.</i>	62
<i>Figura 37. Prueba de ejecución en tiempo real. Fuente: Elaboración propia.</i>	63
<i>Figura 38. Paso adicional para optimizar modelo para GPU. Fuente: NVIDIA®.</i>	64
<i>Figura 39. Pruebas con varios individuos y baja iluminación. Fuente: Elaboración Propia.</i>	65
<i>Figura 40. Idea conceptual “Transfer Learning”. Fuente: Elaboración Propia.</i>	67
<i>Figura 41. Ejemplos del Dataset generado. Fuente: Elaboración Propia.</i>	68
<i>Figura 42. Ejemplos de “Data Augmentation Techniques”. Fuente: Elaboración Propia.</i>	69
<i>Figura 43. Configuración del modelo basado en ResNet-18. Fuente: Elaboración Propia.</i>	70
<i>Figura 44. Validación en tiempo real del modelo. Fuente: Elaboración Propia.</i>	71
<i>Figura 45. Datase generado para “Operación Nocturna”. Fuente: Elaboración Propia.</i>	72
<i>Figura 46. Sobreajuste característico de la red. Fuente: Elaboración Propia.</i>	73
<i>Figura 47. Ensayo del modelo en operación nocturna. Fuente: Elaboración Propia.</i>	74

Declaración responsable derechos de autor. El autor declara que tiene autorización para reproducir todas las imágenes contenidas en este trabajo, cuyo fin es puramente académico y no comercial. En este trabajo se han empleado imágenes de tres orígenes:

1. Dominio público. Se ha indicado el enlace al pie de cada imagen.
2. Reproducción de imágenes por autorización expresa del autor original. Se ha indicado la autoría al pie de cada imagen.
3. Se han adquirido los derechos para su uso editorial y reproducción en la web.

1. Introducción

Los aeropuertos y aeródromos disponen de una serie de ayudas visuales (luces, colores, gestos etc.) que ayuda a los pilotos a situarse y transitar de manera segura en dicho entorno, comunicándose de una manera efectiva con el resto de personal que les rodea. Estas ayudas, que están descritas por reglamentos internacionales (Civil Aviation Authority (CAA), 1996) y que originalmente fueron diseñadas para interacción entre humanos, deberán ser ahora identificadas y procesadas por los sistemas no-tripulados, permitiendo una fácil integración y despliegue de estos vehículos en dichos espacios. Por otro lado, el mismo sistema podría ser instalado en aeronaves tripulados y servir como elemento de ayuda o supervisión al personal de cabina, mejorando de esta manera su conciencia situacional y facilitando su labor, cuando las condiciones visuales no son óptimas o en situaciones de gran carga de trabajo, factores ambos que suelen estar relacionados con incidentes en tierra.

Como en toda plataforma aérea imperan ciertas restricciones lógicas de peso y tamaño que deberán ser tenidas en cuenta para que la solución propuesta sea de verdadera aplicabilidad al sector. No busca el presente trabajo abordar todos los aspectos necesarios para el diseño de un producto final que pueda ser integrado en una plataforma, sino que únicamente se centrará en cómo abordar la identificación y clasificación de dichos gestos por un software diseñado para tal fin, pero identificando y manteniendo claro el requisito de que la solución final debería poder ser ejecutada en un dispositivo hardware con capacidades limitadas, ya que no tendría lógica que dicho subsistema fuese más pesado o caro que el resto de la aviónica encargada del control, navegación o guiado.

Además de no abordar otras consideraciones mecánicas o electrónicas, tampoco abordará el presente trabajo los temas específicos relativos a la cualificación o certificación que todo equipo aeronáutico debe de cumplir para poder disponer de las autorizaciones y permisos de vuelo. Ahora bien, cabe señalar por ser de emergente actualidad y directamente aplicable a los objetivos del presente proyecto, el reciente plan de trabajo para la incorporación de tecnología basada en "Inteligencia Artificial" que la Agencia Europea para la Seguridad Aérea ha publicado en Febrero de 2020 (European Union Aviation Safety Agency, 2020), en aras de generar tanto el marco técnico como regulatorio que establezca y defina las pautas a seguir para su uso en aviación, un sector altamente regulado y en el que la seguridad juega un papel decisivo.

1.1 Motivación

Los vehículos aéreos no tripulados ya son una realidad en nuestros días. Más allá de su probada eficacia en operaciones aéreas, el siguiente gran reto tecnológico y operativo al que se enfrenta el sector es el de la convivencia simultánea de plataformas tripuladas y no tripuladas. A pesar de lo que pudiera parecer las operaciones en tierra en el entorno aeroportuario presentan una dificultad mayor que la propia separación de aeronaves en el aire. Según los datos consultados (Tomaszewska et al., 2018) más de un 26% de los incidentes reportados suceden en tierra, y los mismos autores cifran las pérdidas en alrededor de 11M€ cada año. Igualmente identifican que debido al imparable aumento de la demanda, el número de accidentes o incidentes en operaciones terrestres creció un 200% en 2017 respecto a 2012, año de inicio del estudio. Entre las causas de dichos incidentes, predomina el factor humano combinado con situaciones climatológicas adversas, elevada carga de trabajo o insuficiente conciencia situacional a la hora de ejecutar la maniobra.

Con el objetivo de avanzar hacia esta integración, este trabajo se centrará en resolver el problema del reconocimiento visual e identificación de un número reducido de las “señales de rampa” que el personal de tierra lleva a cabo mediante el movimiento específicos de sus brazos para codificar diferentes mensajes hacia los pilotos situados en la cabina.

Este miembro del equipo de operaciones terrestres del aeropuerto, conocido generalmente como “señalero, agente de rampa o aircraft marshaller”, va equipado con identificadores visuales característicos: chaleco reflectante, paletas de colores o barras de luz en la oscuridad.



Figura 1. Señalero o “Aircraft Marshalling”. Fuente: [Dominio público](#).

Se considera por lo tanto que, el problema que este proyecto propone abordar es de actualidad e interés para resolver un problema conocido de un sector emergente con impacto en nuestra sociedad. Un dominio del problema podría derivar en el desarrollo completo de un sistema funcional que pudiera ser integrado en plataformas aéreas no tripuladas, junto al resto de equipos ya presentes, permitiendo un despliegue más rápido y efectivo de estas plataformas en las instalaciones ya existentes, no siendo necesario inversión en equipamiento adicional o ayuda específicas para este tipo de plataformas.

Por otro lado, tal y como se proponía en la introducción, el mismo sistema podría tener interés en la aviación comercial tripulada en la que actuaría como un sistema de ayuda y supervisión a los pilotos situados en cabina, mejorando su conciencia situacional y permitiendo detectar y advertir a los mismos de discrepancias entre la información identificado por el sistema de visión y los propios movimientos del avión.

Por todo ello, parece lógico pensar que todo avance tecnológico destinado a facilitar la integración de las nuevas plataformas no tripuladas, llamadas a revolucionar sectores tan dispares como el envío de mercancías, la agricultura o el transporte de pasajeros, en las instalaciones aeroportuarias actuales así como mitigar los riesgos de que se produzcan incidentes por medio de la mejora de la conciencia situacional y reducción de los pilotos humanos en cabina por medio del uso de sistemas inteligentes de supervisión, será de gran interés para industria, y que tendrá repercusiones muy positivas tanto económicas como sociales.

1.2 Planteamiento del trabajo

El objetivo final será disponer de un conjunto formado por una cámara RGB y una unidad embebida de procesamiento de tamaño reducido y capacidad de procesamiento limitado que pudiera ser instalado en vehículos no-tripulados de medio tamaño (<150 kg). Sobre dicho conjunto se ejecutará un SW tal que permita identificar y clasificar con un porcentaje de acierto de más del 80% un número limitado de gestos de todo el conjunto posible en tiempo real.

Se abordará el problema desde una perspectiva exclusivamente visual, no dependiendo por tanto de sensores adicionales presentes en la propia cámara (e.g. sensor de profundidad) o en los miembros del personal de tierra (e.g. "wereables").

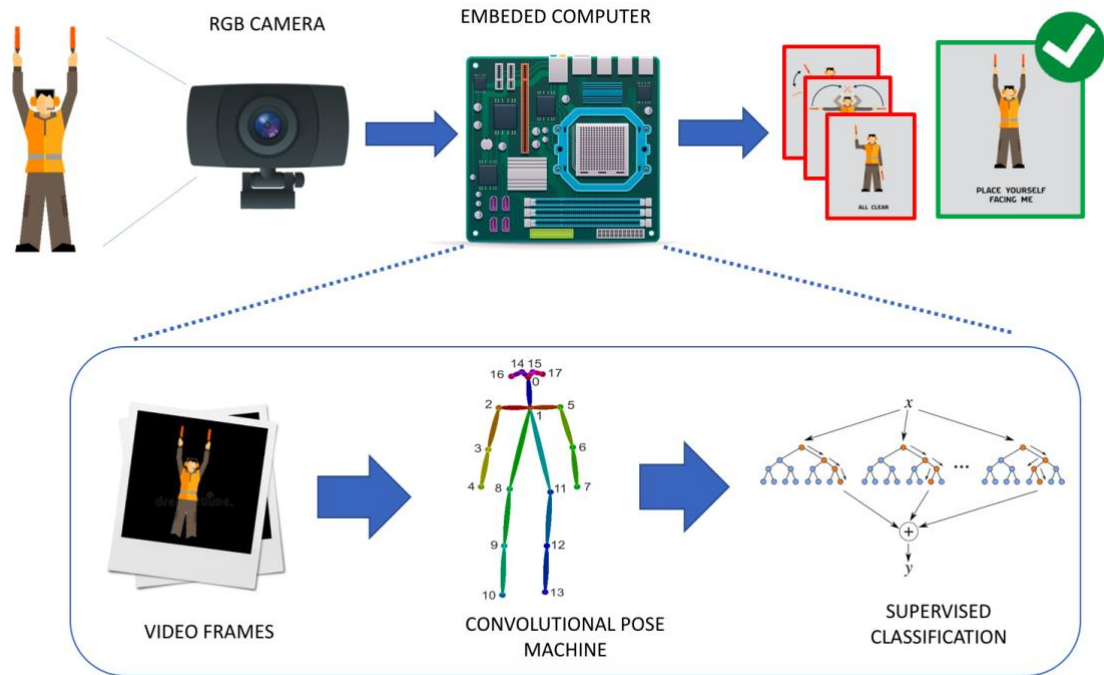


Figura 2. Esquema funcional “alto nivel”. Fuente: Elaboración propia.

La arquitectura de alto nivel de la solución quedaría definida por los siguientes bloques funcionales:

1. **Captura de video.** Engloba las tareas de captura de video a una tasa de refresco y con una calidad tal que permita los posteriores análisis. Quedarían aquí incluidos los trabajos de pre-procesamiento de la imagen iniciales. Cabe destacar que si bien se utiliza una cámara de video, el proceso se realizará de manera discreta sobre las imágenes extraídas como “frames” del propio video.
2. **Identificación en tiempo real de “señalero”.** Busca diferenciar este individuo respecto al fondo (otros objetos, personas..). Para ello se podría hacer uso de técnicas de filtrado por los elementos visuales característicos (chaleco reflectante, luces en las manos etc..). Esto permitirá reducir el problema a un solo individuo.
3. **Aplicación de “Convolutional Pose Machine - CPM”.** Mediante la aplicación de una red neuronal pre-entrenada y de uso abierto se llevará acabo identificación de las estructuras articulares parametrizadas del cuerpo humano. Con este paso, transformaremos la imagen en una serie de variables numéricas que representan los movimientos de las articulaciones, transformando el conjunto de “features” del dominio de los pixels-imágenes al de vectores espaciales, simplificando los cálculos

posteriores. Quedan englobados de igual forma en este grupo las tareas de filtrado y preparación de los datos para las etapas posteriores.

4. **Clasificación de los gestos mediante Machine Learning.** Englobamos aquí las funciones finales que permitirán clasificar el gesto capturado por la cámara, y traducido mediante “CPM” en coordenadas, en una de las categorías previamente definidas. Se propondrán y compararán diferentes métodos, escogiendo aquel que mejor se adapte a las restricciones propias del problema, tales como “*Random Forest*” o “*DNN*”. La parte final de esta etapa ejecutará las lógicas de ponderación y filtros paso-bajo que permitan mantener una salida continua entre las diferentes categorías, minimizando efectos de parpadeo por clasificación en categorías diferentes de frames consecutivos.

Uno de los puntos potencialmente más conflictivos, es la elaboración de un “conjunto de entrenamiento” de los gestos que queramos identificar para poder llevar a cabo el paso de la clasificación, debido a la no existencia de dataset preparados para tal fin. Para solventar este problema, y tras analizar los buenos resultados obtenidos por otros autores (Song et al., 2011) llevando acabo procesos similares con el objetivo de generar un pequeño dataset específico de los gestos deseados, se propone realizar un programa complementario y más sencillo que el anterior que permita las labores de captura y etiquetado de las muestras de entrenamiento.

Dicha aplicación permitirá la captura en video de varios individuos de diferentes estaturas y complejión, repitiendo varias veces cada gesto, en escenarios de iluminación no similares. La información capturada se procesará mediante “CPM” y se almacenará directamente las coordenadas de las articulaciones. El asistente será el encargado de etiquetar mediante un acceso rápido de la propia aplicación cada una de las posturas que se guardan.

Como el objetivo del proyecto se contextualiza al de una prueba de concepto en su forma de validador tecnológico que permita analizar las diferentes técnicas para solucionar este desafío, este proyecto propone que de los 20 gestos reconocidos por las autoridades civiles internacionales (ICAO, 2005) solo se reconozcan 5 categorías, incluyendo una designada como “no-gesto”.

Merece la pena señalar en este punto las características principales de los gestos:

- Solo intervienen las extremidades superiores.
- No son estáticos, pero la información espacial predomina a la temporal.
- Ambos brazos codifican información.
- Son visualmente independientes.

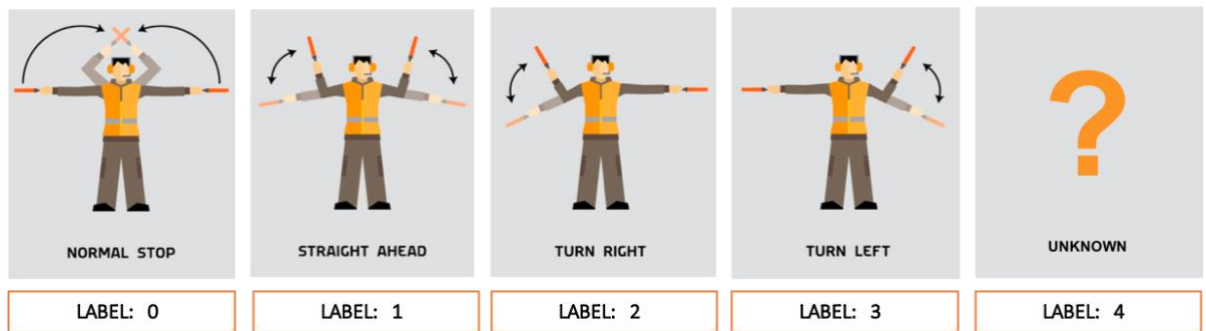


Figura 3. Las 5 categorías escogidas. Fuente: Elaboración propia.

De esta manera se plantea el uso de dos técnicas diferentes estudiadas dentro del ámbito de la inteligencia artificial, que se ejecutarán de manera progresiva e independiente en dos pasos claramente identificados: una primer red convolucional (2D-CNN) pre-entrenada y una posterior clasificación por otra técnica, sobre las “características” extraídas por la primera.

Así pues el método presentado en este trabajo solo usa aprendizaje realmente profundo para la extracción de las posturas del operador, mediante una red en 2D. Las ventajas principal que aporta este enfoque es que permite sacar el máximo provecho de todo el potencial de una red neuronal convolucional muy profunda, pre-entrenada y optimizada haciendo uso de grandes dataset que combinan diferentes fuentes de información. Estas redes (Wei et al., 2016) proporcionan un marco de predicción secuencial para aprender modelos espaciales implícitos, segmentando posteriormente la forma humana presente en la ventana de análisis e incluso llegando a identificar las distintas partes del cuerpo.

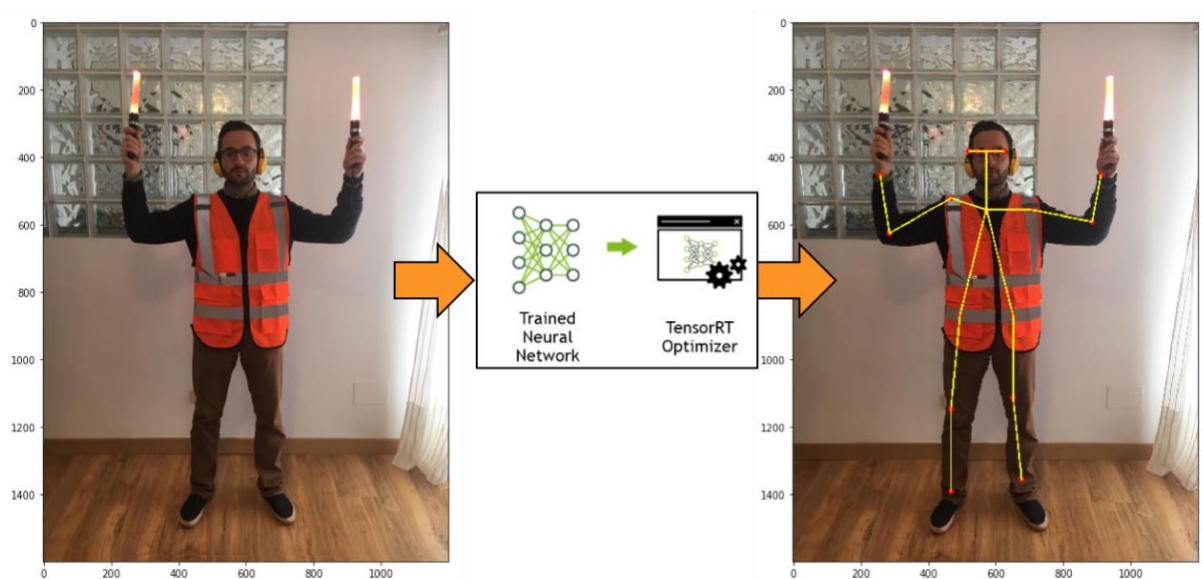


Figura 4. Funcionamiento red neuronal para CPM. Fuente: Elaboración propia.

El “output” de aplicar a una imagen dicha red es un conjunto numerado de coordenadas que identifican las distintas articulaciones (Cao et al., 2017; Lin et al., 2014). Estos valores numéricos, debidamente normalizados y procesados, serán las “*características*” que utilizaremos para la clasificación de los diferentes gestos por la siguiente etapa del proceso.

Si bien, la transformación de imagen a coordenadas cartesianas que nos proporciona “CPM” se traduce en una notable reducción de las dimensiones del problema a tratar, y a pesar del uso de técnicas adicionales para el aumento del banco de ejemplos, el pequeño volumen de datos de entrenamiento de los diferentes gestos a clasificar hace que sea poco recomendable enfrentar el ejercicio como un problema de “*Transfer Learning*” puro, que diese como resultado una nueva capa neuronal de clasificación sobre el modelo neuronal anterior, por el alto riesgo de sobreajuste y así evitar de nuevo tener que rentrenar parcialmente la red completa.

En este sentido, y dados los condicionantes identificados del proyecto en cuanto a capacidad computacional y ejecución en tiempo real, creemos que un método de “*Ensemble*” tipo “*Random Forest*” o una red neuronal “*Feedforward*” de pequeñas dimensiones, podrían desempeñar bien la función de clasificación a pesar de no poder ser entrenadas con un gran volumen de datos de entrenamiento. Sin embargo, se plantean varias alternativas que deberán ser debidamente probadas y cuyos resultados se analizarán de forma metódica con el fin de identificar el método más idóneo.

Finalmente, se propone una lógica adicional que actúe a la salida del clasificador a modo de lógica de ponderación o filtro paso bajo. Con esta última etapa se aborda el problema de que “*frames*” consecutivos sean clasificados de manera distinta, produciendo efecto de parpadeo en la clasificación global.

En un principio, se ha descartado abordar el problema mediante el empleo de redes neuronales 3D con la capacidad de procesar video directamente y reconocer las acciones. Aunque en el capítulo 2 se profundiza más en el tema, justificamos esta decisión por dos motivos principales:

- El coste computacional necesario para su entrenamiento, pero también su explotación requiere de una potencia computacional no disponible en los sistemas embebidos actuales.
- Aunque existen “*datasets*” de acciones en video ejecutadas por humanos, al carecer de las categorías perseguida por este trabajo, sería necesario rentrenar la red y para ello se estima que el banco de imágenes a generar en el proyecto debería ser de un orden de magnitud mayor al necesario para ejecutar otras técnicas de clasificación tradicionales.

1.3 Supuestos de partida.

Tal y como ha quedado descrito anteriormente el alcance del proyecto se circunscribe a una prueba de concepto, que tendrá lugar por medio de la construcción de un validador tecnológico. Es por ello que se cree conveniente identificar en este punto que supuestos se van asumir para facilitar la ejecución del proyecto:

1. El sujeto permanecerá en una posición lo más enfrentada a la cámara posible.
2. La calidad de la imagen será tal que permita al propio ojo humano identificar el gesto que se está ejecutando.
3. La velocidad o cadencia de ejecución de los gestos será similar a la de la realidad, evitando gesticulación excesivamente rápida, cambiante o contradictoria.
4. Los gestos o posturas se ejecutarán de tal forma que las extremidades superiores solapen lo menos posible con el tronco, tal y como sucede en la realidad.
5. La iluminación será tal que al menos se reconozca parcialmente la silueta del operador de rampa.
6. El operador de rampa tendrá una actitud colaborativa en todo el proceso, favoreciendo al máximo la identificación en aras de la seguridad operacional.
7. No se abordarán temas relativos a la integración mecánica o electrónica en la plataforma aérea.
8. No se abordarán temas relacionados con las estrategias de diseño software o su nivel de garantías, necesarias en todo desarrollo software/hardware destinado para su uso en aviación.
9. La solución escogida deberá poder ser ejecutada en computador embebido con características “del estado del arte”.

El lector interesado podrá encontrar en el capítulo 7 un conjunto de propuestas de mejora que el autor propone con el fin de afrontar o minimizar los efectos adversos sobre los resultados derivados del no cumplimiento de estos supuestos, tales como las operaciones nocturnas.

1.4 Estructura de la memoria.

El presente trabajo se organiza en 8 capítulos, cuya distribución es la siguiente:

1. En el capítulo primero, se presenta el problema a resolver y se introduce las líneas generales que seguirá el desarrollo del sistema propuesto.
2. A continuación, en el segundo capítulo, se hace un análisis pormenorizado del dominio del problema así como una revisión de las distintas técnicas extraídas de la literatura del campo que podrían usarse para abordar este proyecto.
3. El tercer capítulo recoge de una manera formal, el alcance del proyecto, sus objetivos así como la metodología de trabajo propuesta.
4. El capítulo número cuatro es el primero de los capítulos en el que se abordarán los detalles de la implementación técnica. En él se definirán los requisitos de la solución final y se presentará la arquitectura adoptada, explicando la función de cada uno.
5. El quinto capítulo aborda la descripción de la herramienta software así como la elaboración del dataset, explicando la metodología seguida y las herramientas desarrolladas.
6. En el capítulo sexto se evaluarán los resultados obtenidos en la fase de experimentación por las diferentes técnicas presentadas, haciendo una comparativa objetiva que permita identificar el método más idóneo.
7. Por último, el capítulo siete cierra esta memoria haciendo un resumen de las contribuciones del trabajo presentado e identificando posibles líneas de investigación y mejora.

La bibliografía relevante así como los anexos necesarios que complementan este trabajo están disponibles al final de la misma.

2. Estado del arte: contexto y propuesta de solución.

2.1. Dominio del problema: “Aircraft Marshalling”.

“Señalero” o “*Aircraft Marshaller*” es la designación oficial (ICAO, 2005) para referirse al miembro del equipo que forma parte del personal de tierra y que ayuda a los pilotos en la ejecución de determinadas maniobras cuando éstas entrañan un riesgo de incidente, siendo un complemento a las indicaciones que los controladores de superficie hacen llegar a los pilotos. Esta figura está también presente en el sector de la defensa, recibiendo el nombre de “personal de vuelo en cubierta” o “*Shooters*”, cuando realizan sus funciones en portaviones, plataformas en la que desempeñan un papel fundamental. El equipamiento habitual de estos operadores incluye: un chaleco de alta visibilidad, unos cascos de protección auditiva y dos “varitas o linternas” de maniobra.

Normalmente el equipo mínimo necesario está compuesto por 3 técnicos especialistas debidamente cualificados, que trabajan de manera coordinada. Son el nexo de unión entre el personal de cabina y el resto de operadores en tierra, debiendo velar por la seguridad e integridad de personas y equipos. Sus funciones básicas se resumen en:

- Guiar a las aeronaves en su rodadura en tierra.
- Autorizar determinadas maniobras a los aviones.
- Coordinar los procedimientos de todo el personal implicado.
- Gestionar plan de actuación ante emergencia.
- Supervisar el cumplimiento de la normativa y alertar de cualquier incumplimiento.



Figura 5. “Aircraft Marshaller” en aeropuerto. Fuente: [SrA D. Sloan. Dominio Público.](#)

Se conocen como “señales de rampa” al conjunto de gestos que el personal de tierra lleva a cabo mediante el movimiento específicos de sus brazos para codificar los diferentes mensajes hacia los pilotos situados en la cabina, desde donde por lo general disponen de una mala visión del espacio alrededor del avión durante la fase de rodadura y parking. Si bien, en muchos aeropuertos internacionales ya existen “sistemas automáticos guía para el acoplamiento con la terminal”, este conjunto de señas siguen siendo empleadas alrededor de todo el mundo, en operaciones tanto civiles como militares, priorizando sus indicaciones frente a la de cualquier otra señalética o sistema guía presente.

Existen diferentes conjunto de gestos dependiendo de su ámbito de aplicación, así pues podemos encontrar ciertas discrepancias entre la señalética utilizada en la pista de un porta aviones de la OTAN (Song et al., 2011) o las empleadas en los aeropuertos internacionales de uso civil (Civil Aviation Authority (CAA), 1996), pero también podemos encontrar diferencias entre el tipo de aeronave, no siendo todos los gestos equivalentes para apoyar la operación de un helicóptero que las de una aeronave de ala fija.

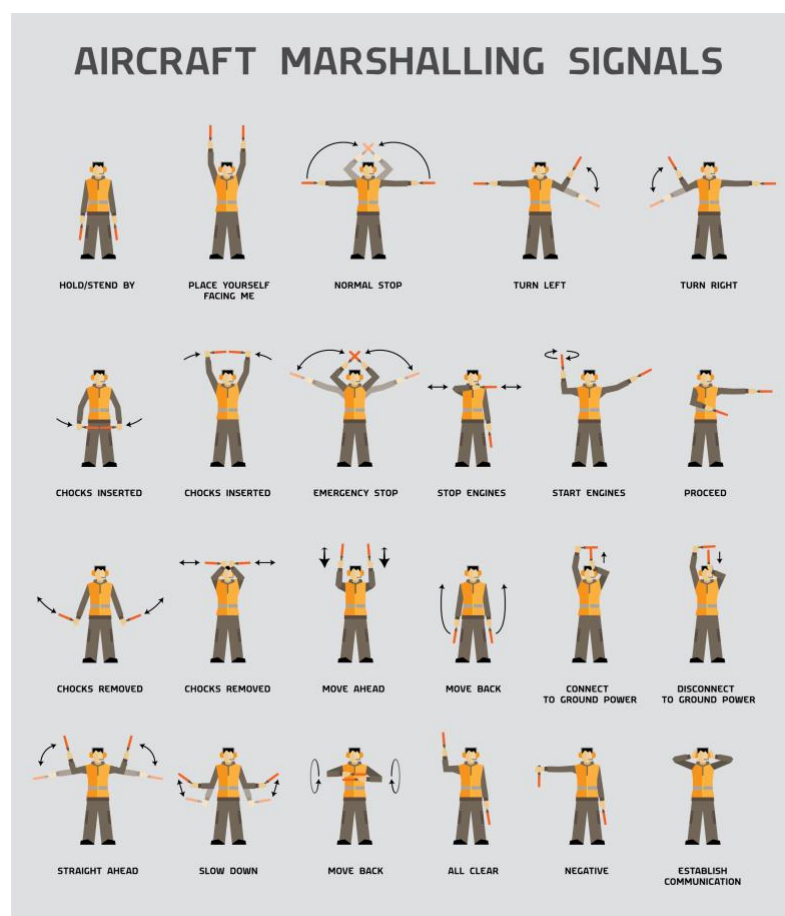


Figura 6. Colección de señales más empleadas. Fuente: Vectorstock. Derechos comprados

La Organización de Aviación Civil, o por sus siglas en inglés ICAO, promulga una serie de recomendaciones de obligado cumplimiento para los países adheridos a ella, con el fin de establecer un estándar internacional en todos los aspectos referentes al transporte aéreo. En este trabajo nos centraremos en las señales empleadas por el personal civil para comunicarse con una aeronave de ala fija, ya que son las más comúnmente utilizadas.

Merece la pena señalar en este punto las características principales de los gestos:

- Solo intervienen las extremidades superiores.
- No son estáticos, pero la información espacial predomina a la temporal.
- Ambos brazos codifican información.
- Son visualmente independientes.

2.2. Estado del arte: “Interacción Humano-Robot”.

La interacción entre hombre y máquina, generalmente referido en inglés como HRI (“*Human-Robot Interaction*”) (Castillo et al., 2019) es un campo de estudio multidisciplinar que aborda el diseño de mecanismos de comunicación más eficientes y efectivos entre humanos y máquinas, que permita a ambos colaborar en la ejecución de tareas.

El objetivo final es identificar y desarrollar aquellas técnicas que puedan permitir una comunicación entre los hombre y los robots de su entorno de tal manera que no sea necesario usar telecomandos específicos. Es un campo de gran interés y dificultad, en el que se estudia tanto la manera en la que los propios humanos nos comunicamos e interaccionamos entre nosotros, de manera explícita e implícita, así como los mecanismos y procesos necesarios para implementar habilidades semejantes en las máquinas dotadas de un cierto nivel de inteligencia. Esta comunicación puede usar uno o varios canales de manera simultánea, entrando en juego los sistemas de procesamiento del lenguaje natural, sistemas de visión artificial, reconocimiento y clasificación de gestos etc.

Desde un punto de vista industrial, dotar de capacidades de este tipo a las máquinas supondría una mejora en la eficiencia, al permitir una rápida coordinación y cooperación entre operadores, a la vez de un incremento en la seguridad del trabajo. En este sentido algunos autores (Martin & Moutarde, 2019) introducen las actuales limitaciones que, por ejemplo, se imponen en las líneas de producción a la presencia de operarios humanos cuando los brazos robots están funcionando para evitar accidentes. Un sistema capaz de detectar la presencia de un operario e incluso analizar que está asustado y consecuentemente detener la acción en ejecución, sería una ventaja en términos de seguridad.

Por otro lado, desde la vertiente más social, se expone (Castillo et al., 2019) igualmente la importancia de las intenciones, intereses, sentimientos e ideas que de manera natural el ser humano comunica durante sus interacciones con otros individuos, y como un agente inteligente de tipo artificial debería poder ser capaz de captarlos para permitir alcanzar niveles de comunicación que realmente generen una confianza total.

El campo de los sistemas autónomos también está muy interesado en esta área y sus avances. El equipo de investigación en IA de NVIDIA® defiende en su trabajo (Molchanov et al., 2016) como una comunicación no verbal entre coche autónomo y humano, basada en el uso de determinados gestos, o expresiones fáciles incluso, conduciría a un aumento de la seguridad y confort de los pasajeros.

En este sentido, destaca también el trabajo del equipo de investigación del “*Georgia Tech Institute*” sobre comunicaciones bajo el agua entre submarinistas y vehículos de asistencia subacuática (Demarco et al., 2014). Es relevante en tanto que en este tipo de aplicaciones cualquier otro medio de comunicación, como la propia voz u otras formas de ondas, no son posibles al verse fuertemente atenuadas por el agua. Los submarinistas aprenden y utilizan una serie de señales y gestos para comunicarse cuando están bajo la superficie. Este trabajo analiza el problema y presenta una solución basada en la codificación de ciertas señales mediante un código de luces que permiten la comunicación con el robot subacuático.

El problema que el presente trabajo aborda se circunscribe dentro de este campo en tanto que intenta resolver la comunicación entre los operadores de rampa y los vehículos autónomos a su alrededor sin utilizar controles o mandos remotos. Esta interacción se clasificaría como comunicación explícita, en la medida en que los mensajes codificados son únicos e identificables de manera biunívoca, y al no jugar un papel importante la comunicación implícita más propia de valoraciones subjetivas o en las que se tengan que hacer asunciones basadas en los rasgos faciales o manera de realizar los gestos.

Por otro lado, los factores de comunicación (Shannon, 1997) que modelan nuestra interacción se podrían clasificar como:

- *Emisor*: Señalero emite el mensaje mediante el movimiento de sus brazos.
- *Receptor*: Aeronave (no tripulada) que transita por el aeropuerto.
- *Código*: Las señales de rampa descritas por reglamentos internacionales.
- *Dirección*: Unidireccional de emisor hacia receptor, sin esperar respuesta.
- *Contexto-Libre*: Cada gesto no depende del gesto anterior para poder ser entendido.
- *Canal*: La información del mensaje está codificada de manera visual.
- *Ruido*: Interferencias debidas a la falta de visibilidad o luminosidad.
- *Redundancia*: El gesto se mantienen durante todo el tiempo que el mensaje tenga prevalencia.

2.3. Estado del arte: “Reconocimiento de Gestos”.

El reconocimiento de gestos es un área profusamente estudiada durante las últimas dos décadas. Como se venía analizando en la sección previa sobre la interacción hombre-máquina, esto se debe a la gran cantidad de información que codificamos de manera habitual mediante gestos, y que complementan la comunicación verbal o directamente la sustituyen cuando esta no es posible, debido a limitaciones del canal (Demarco et al., 2014) o de los propios agentes involucrados en el proceso (e.g. pérdida de capacidad auditiva).

Las aplicaciones actuales de las técnicas de “reconocimiento de gestos” son muchas y muy variadas, desde interacción con sistemas autónomos, domótica, video juegos o robots socio-sanitarios. Un campo de estudio muy habitual es el del reconocimiento de la “lengua de signos” que permite la comunicación natural por medios no orales a través de un canal visual mediante el uso de gestos con las manos. Hay numerosas publicaciones al respecto que abordan este complejo problema desde diferentes perspectivas, teniendo en cuenta las diferentes variaciones geográficas, así como la importancia no solo espacial sino temporal que involucra una solución general al problema. Una completa revisión del tema, así como las diferentes estrategias presentadas en los últimos años puede encontrarse en (Zaman Khan, 2012).

Una primera distinción que encontramos se puede referir a, si las técnicas y herramientas de reconocimiento se centran en zonas concretas del cuerpo, como podría ser el caso de las manos en el ejemplo expuesto anteriormente para la identificación y reconocimiento de “lengua de signos”, o por el contrario procesan información del cuerpo entero. Esto es relevante en tanto que limita la precisión o nivel de detalle esperado. Así por ejemplo, un sistema como el presentado por (Ribó et al., 2016) en el que únicamente se persigue clasificar posturas corporales no se analiza la posición de los dedos de la mano, mientras que en la aplicación presentada por (Molchanov et al., 2016) para la mejora de la conducción de coches autónomos, por el contrario, la posición de la mano y sus diferentes falanges predomina a la información corporal al estar sentado el conductor.

A continuación, podemos establecer una segunda clasificación entre las técnicas que nos permiten distinguir posturas o señas en las que la información espacial predomina sobre la temporal, de aquellas otras en las que juega un papel importante la propia dinámica del gesto, es decir, la velocidad de ejecución codifica también información. Un ejemplo del primer grupo lo encontramos en (Pullen & Seffens, 2018) en la que se propone un sistema para clasificar posturas de Yoga, que destacan por ser eminentemente estáticas. Estas técnicas se diferencian de las segundas por ser más simples y sencillas, al no tener que prestar atención a la dimensión temporal, que suele adquirir la forma de una secuencia de datos relacionados.

La clasificación de gestos dinámicos es más compleja al requerir establecer dicha relación en una secuencia. Encontramos numerosos ejemplos cotidianos que ejemplifican este tipo de comunicaciones, como puede ser la velocidad a la que un director de orquesta mueve su brazo para marcar el ritmo o el cambio de velocidad en el movimiento de la mano cuando estamos dando indicaciones de aparcamiento y queremos alertar que aminore la maniobra o por el contrario que está libre de obstáculos. Un excelente reflexión sobre la influencia de la velocidad y la posición en el reconocimiento dinámico de gestos se puede encontrar en (Castillo et al., 2019).

Encontramos diferentes propuestas a lo largo de la historia que abordan el problema de la secuencias de gestos encadenados. Destaca entre otros el uso de “*Modelos ocultos de Markov-HMM*” (Kapuscinski et al., 2015), el “*Problema de la subsecuencia común más larga*” (Choi et al., 2008) o el “*Algoritmo de Viterbi*” (Waldherr et al., 2000). Sin embargo, sobre todos los demás destaca el uso de técnicas basadas en “*Dynamic Time Warping*”. Esta técnica permite comparar dos secuencias con escalas de tiempos diferentes y determinar su grado de similitud. Junto con un clasificador que ayude a determinar cuál de las secuencias de referencia es la más similar permite conseguir resultados óptimos en la clasificación de gestos dinámicos. Son varios los autores que han seguido este enfoque tanto para gestos de cuerpo entero (Ribó et al., 2016) como solo de la mano (Raheja et al., 2015).

En la actualidad han surgido nuevas técnicas basadas en el uso de redes neuronales, especialmente aquellas destinadas a la clasificación de video, ya que al ser una secuencia de fotogramas no es suficiente una mera clasificación de imágenes estáticas. En este sentido encontramos diferentes enfoques que combinan múltiples redes. Destaca el trabajo de (Donahue et al., 2017) que propone extraer las características de cada fotograma de video por medio de una red convolucional (CNN) para luego pasar la secuencia a una segunda red recurrente (RNN) independiente. Una arquitectura similar la encontramos en el trabajo de (Zhou et al., 2018) pero en este caso, las “features” extraídas por la red convolucional se pasan a un “*Multi Layer Perceptron*” (MLP). Esta arquitectura se basa en la hipótesis de que una “MLP” podrá inferir las características temporales de la secuencia de forma natural, sin tener que saber que es una secuencia. Por otro lado, otros autores (Simonyan & Zisserman, 2014a) defienden las ventajas de una arquitectura basada en 2 redes de tipo convolucional en paralelo, en la que una de ellas procesa información espacial de cada fotograma, mientras que la segunda trata de identificar variaciones basadas en el flujo óptico en una secuencia de imágenes fijas. Finalmente debemos mencionar el trabajo basado en redes convolucionales en tres dimensiones (3D-CNN), que son capaces de extraer características gráficas directamente de un conjunto de fotogramas con relación temporal entre ellos, identificando una representación de bajo nivel del conjunto por medio de operaciones convolucionales en

3D. Este enfoque ha dado excelentes resultados como se puede ver en el trabajo presentado por el equipo de investigación de NVIDIA® (Molchanov et al., 2016), pero también presenta algunas desventajas importantes, tal y como analiza (Hara et al., 2018) respecto a las anteriores arquitecturas basadas en redes bidimensionales. Por un lado, las 3D-CNN son mucho más demandantes en términos computacionales, tanto de recursos de cálculo como de memoria, lo que a pesar de los grandes avances en el estado actual de la tecnología limita su uso para aplicaciones en tiempo real o en plataformas embebidas. Por otro lado, a día de hoy no existen tantas redes pre-entrenadas para la extracción de características en videos, como si existen numerosas y muy potentes redes entrenadas usando arquitecturas 2D-CNN. Tal y como presenta (Hara et al., 2018) esta limitación se empieza a superar y están surgiendo redes 3D-CNN que utilizan los mismos patrones de sus homólogas en 2D por lo que se podrían beneficiar del avanzado estado de madurez de estas.

Volviendo de nuevo al análisis del reconocimiento de gestos, por último, podemos nuevamente clasificar las técnicas de reconocimiento de gestos presentadas hasta la fecha en dos categorías: colaborativas o no colaborativas. Entendemos por técnicas colaborativas, o también invasivas, aquellas que basan su funcionamiento en el uso de hardware adicional que el usuario que ejecuta la acción debe llevar puesto con el objetivo de capturar el movimiento y luego enviar a la máquina. En este sentido podemos encontrar propuestas basadas en guantes (Kim et al., 2009) o chalecos que por medio de diferentes sensores, como sistemas inerciales o “*encoders*” de posición, miden el movimiento de las diferentes falanges o extremidades.

Más interesante para nuestro propósito son los trabajos llevados a cabo basados en aquellas técnicas no dependientes de sistemas hardware adicionales que los operadores de rampa deban llevar encima, más allá de los elementos pasivos que ya utilizan. En este sentido, debemos mirar hacia las técnicas basadas en “visión por ordenador” al basarse estas en información no colaborativa que el sistema debe capturar y analizar de manera independiente del emisor.

En este sentido encontramos una basta colección de artículos que recogen numerosos enfoques a lo largo de los años basados en diversas técnicas de visión por ordenador, comenzando por el enfoque de Paul Viola y Michael Jones (Viola & Jones, 2001) basado en la función de “*Ondícula de Haar*” para la identificación de objetos basados en la detección en cascada de características simples. Posteriormente, le siguió el trabajo presentado por Dalal y Trigs (Dalal & Triggs, 2005) basado en el empleo de “*Histogramas de gradientes orientados - HOG*”. Ambas técnicas fueron revolucionarias y todavía se usan en diferentes aplicaciones debido a que son poco demandantes computacionalmente y son fácilmente accesibles, pero

sin embargo presentan algunas desventajas de base, como la cantidad de falsos positivos o la incapacidad para detectar humanos en diferentes posturas si éstos no están perpendiculares a la cámara.

Enfoques más modernos se basan nuevamente, en el empleo de redes convolucionales profundas. Desde que en 2012 AlexNet (Krizhevsky et al., 2012) mostrase el potencial de las “CNN” para la clasificación de imágenes, son numerosas las aplicaciones en las que se han usado. De especial interés para el reconocimiento de gestos y posturas son las “*Convolutional Pose Machines - CPM*” (Wei et al., 2016) un conjunto de redes pre-entrenadas que permiten la identificación de las articulaciones principales del cuerpo, o zonas específicas como manos, pies o incluso rasgos faciales, mediante una serie de coordenadas o “punto singulares” que conforman un “esqueleto artificial”. Dado el alcance del proyecto nos centraremos en el reconocimiento de posturas del cuerpo humano completo mediante la identificación de sus articulaciones y extremidades principales.

En la actualidad existen varias implementaciones basadas en diferentes arquitecturas y entrenadas con distintos modelos que permiten la detección de múltiples personas en una misma imagen. El problema del reconocimiento de posturas se puede subdividir en dos problemas más pequeños: segmentación y reconocimiento de puntos singulares (Schneider et al., 2019). En este sentido podemos hacer una distinción entre las diferentes implementaciones atendiendo al tipo de arquitectura. Si siguen un enfoque en dos pasos, podemos hablar de modelos “de arriba hacia abajo” (Fang et al., 2017), en las que primero se ejecuta una segmentación para identificar a todas las personas en la imagen para luego extraer las posturas, o por el contrario, siguen una arquitectura “de abajo arriba” (Cao et al., 2019) en las que primero se identifican los puntos singulares de la imagen y posteriormente se agrupan para formar “esqueletos” con sentido, dando como resultado la segmentación de múltiples personas.

Entre los diferentes modelos destaca el desarrollado por el laboratorio de “Percepción Computacional” de la universidad norteamericana de “*Carnegie Mellon*”, que ganó la competición “*COCO Keypoint Challenge*” en el año 2016. Los autores del trabajo (Cao et al., 2017) diseñaron y entrenaron una arquitectura basada en varias redes profundas que consta de dos etapas.

La primera consta de 10 capas basadas en la conocida red “VGGNet” (Simonyan & Zisserman, 2014b) y su función es la de extraer un mapa de características de la imagen analizada. La segunda etapa, más compleja, consta de dos redes convolucionales en paralelo.

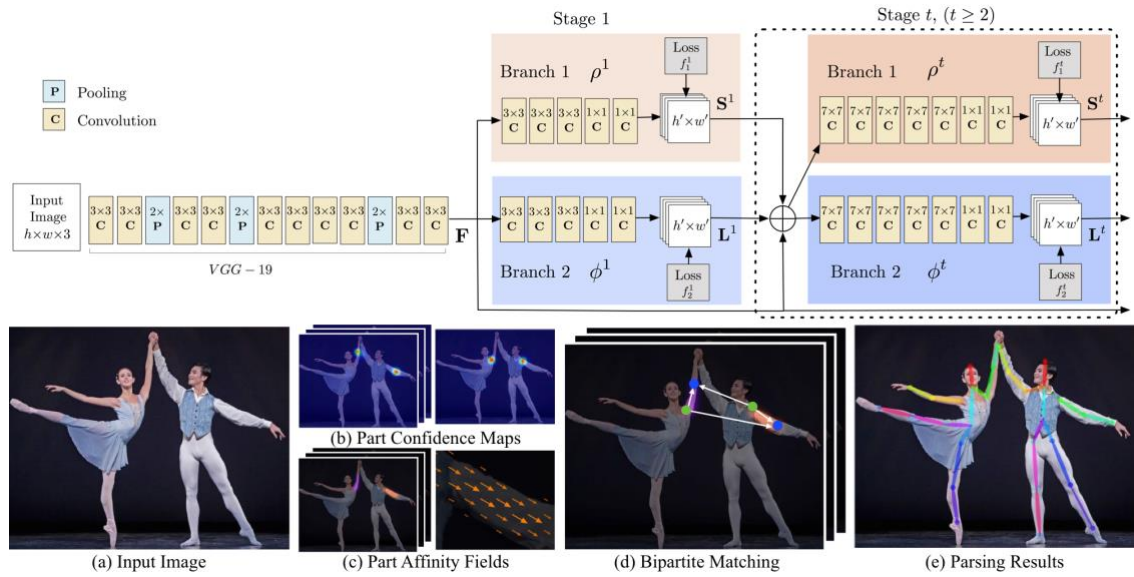


Figura 7. Arquitectura CPM usada por OpenPose. Fuente:(Cao et al., 2019).

La primera rama predice un conjunto de mapas de confianza, o figuras de mérito, de la probabilidad de que la zona identificada corresponda a alguno de los puntos de interés, que como hemos visto anteriormente son las articulaciones o puntos singulares del cuerpo humano: hombros, rodillas, cuello etc.

La segunda rama convolucional por su parte infiere un mapa de confianza que codifica el grado de asociación entre dichos puntos singulares para formar los vectores que conformaran el esqueleto. El resultado de ambas redes se cruza utilizando un algoritmo voraz para finalmente identificar todos los puntos singulares y vectores entre éstos que configuran los esqueletos que identifican la posición que ocupan los diferente cuerpos de la imagen.

Al igual que cualquier otro problema que se aborde mediante una arquitectura basada en redes profundas los resultados del modelo pre-entrenado dependerán en gran medida del conjunto de entrenamiento utilizado. Afortunadamente, debido al gran interés científico e industrial en este campo en los últimos años han surgido numerosos datasets de gran calidad entre los que cabe destacar “*Common Objects in Context - COCO*” (Lin et al., 2014) o “*CMU Panoptic Dataset*” (Joo et al., 2015). Este último destaca por el empleo de unas instalaciones especiales en las que mediante el uso de más de 500 cámaras distribuidas de manera estratégica en una cúpula, han grabado y añadido de forma manual más de 1.5 millones de etiquetas de las diferentes posturas y acciones de un grupo de personas. Encontramos, sin embargo, que no existe un estándar y que dependiendo del modelo utilizado producen resultados diferentes. En el caso de “COCO” por ejemplo, el esqueleto está compuesto por 18 puntos significativos, siendo el punto inicial el cuello. Otros modelos identifican un número diferentes de puntos singulares y sistemas de referencia.

Por lo general, cada punto singular está compuesto por un vector de 3 componentes, siendo las dos primeras las coordenadas espaciales “x e y” respecto a una referencia y el último una medida de confianza de que dicho punto se corresponda realmente con el aérea del cuerpo identificada.

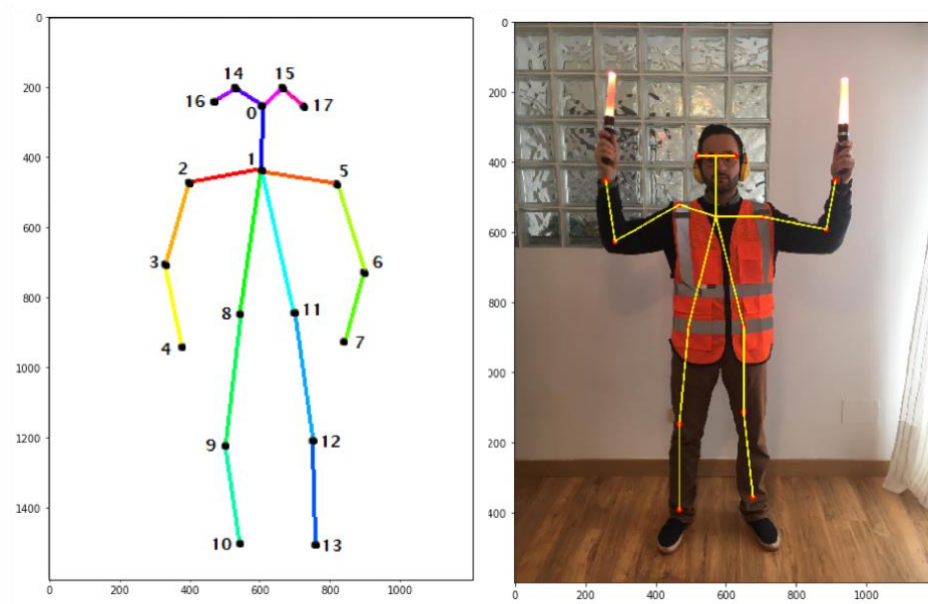


Figura 8. Salida “CPM” en formato “COCO”. Fuente: Elaboración propia.

Cabe mencionar aquí que si bien los modelos presentados hasta ahora eran modelos bidimensionales, es decir, cada articulación o punto singular se definía con solo dos coordenadas espaciales, también existen modelos que permiten expandir información a una tercera dimensión. Para ello es necesario el empleo de, o bien cámaras con sensor de profundidad como es el caso de la popular plataforma *Microsoft Kinect®*, que permiten la captura de información visual y de profundidad de manera simultánea (Ribó et al., 2016), o por el contrario de modelos matemáticos que permitan inferir a partir de un modelo 2D una representación 3D. Entre los diferentes enfoques destaca “*Human Mesh Recovery – HMR*” (Kanazawa et al., 2018) que se basa en el empleo de redes antagónicas entrenadas previamente con un conjunto de fotos en dos dimensiones etiquetadas con información espacial.

Siguiendo esta idea destaca el enfoque de “*Skepxels*” (Liu et al., 2017) que propone añadir a las anteriores coordenadas otras con información temporal o de velocidad que permita tener una representación no solo espacial sino temporal usando el mismo concepto de “esqueleto”.

2.4. Valoraciones finales y propuesta de desarrollo.

Comenzamos este capítulo analizando en más detalle el origen e importancia de las “señales de rampa” que el personal de tierra lleva a cabo para codificar determinados mensajes a los tripulantes en cabina. Hemos visto como su uso se justifica por la dificultad de transitar por el aeropuerto, la coordinación entre los tripulantes y el resto del equipo de asistencia en tierra, así como por la seguridad operativa. Existen diferentes variaciones dependiendo de si es un entorno militar o civil. En el caso civil, se ha demostrado que existe normativa de carácter internacional que recoge y especifica como han de ser estos gestos, así como la equipación y características que debe cumplir el personal de tierra encargado de ejecutarlas. Por último, hemos hecho un análisis de cómo codifican información, destacando la información espacial sobre la temporal y el movimiento del tronco superior respecto al inferior. Esto se justifica debido a que las señales fueron diseñadas para que un tripulante humano situado en la cabina del avión pudiese identificar cada gesto a una distancia considerable, por lo que se evita codificar información en profundidad, se basan en el movimiento perceptible únicamente de los brazos respecto al tronco y las señales son claramente diferenciables entre ellas.


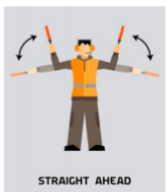


0	<p>STOP</p> <p>Indica al piloto cuándo debe detener la aeronave. El señalero generalmente levanta sus manos lentamente para anticipar la reacción del piloto. La señal de "parada de emergencia" es similar, pero el movimiento de las manos será rápido, lo que indica que la aeronave debe detenerse de inmediato.</p>	 <p>NORMAL STOP</p>
1	<p>AVANCE</p> <p>Indica al piloto que la aeronave está libre de cualquier obstrucción y puede avanzar desde su posición actual directamente hacia la posición del señalero.</p>	 <p>STRAIGHT AHEAD</p>
2	<p>GIRO DERECHA</p> <p>Indica al piloto que necesita girar el morro del avión hacia la derecha, en la dirección del brazo extendido. Si bien la cadencia de giro suele hacerse a velocidad estándar de cada aeronave, un incremento de esta le comunica al piloto que aumente la tasa de giro.</p>	 <p>TURN RIGHT</p>
3	<p>GIRO IZQUIERDA</p> <p>Indica al piloto que necesita girar el morro del avión hacia la izquierda, en la dirección del brazo extendido. Si bien la cadencia de giro suele hacerse a velocidad estándar de cada aeronave, un incremento de esta le comunica al piloto que aumente la tasa de giro.</p>	 <p>TURN LEFT</p>

Tabla 1. Gestos reconocidos en nuestra implementación.

Como el objetivo del proyecto se contextualiza al de una prueba de concepto en su forma de validador tecnológico que permita analizar las diferentes técnicas para solucionar este desafío, este proyecto propone que de los 20 gestos reconocidos por las autoridades civiles internacionales (ICAO, 2005) solo se identifiquen 5 categorías, incluyendo una designada como “no-gesto”.

A continuación hemos identificado como este problema se circunscribe dentro del ámbito de la interacción “hombre-máquina”. Se ha demostrado porqué es un campo de gran interés para la ciencia e industria, y las múltiples aplicaciones posibles en diversos campos. Desde luego, para el objetivo del proyecto las aplicaciones más relevantes son aquellas que analizan la interacción con sistemas autónomos. Hemos analizado el caso de la comunicación submarina como un ejemplo en el que el medio condiciona el canal disponible de comunicación. Por último hemos identificado los factores que describen el proceso de comunicación de nuestro problema, destacando que se trata de una comunicación explícita, unidireccional, libre de contexto y cuyo canal debe ser el visual, dado el dominio de problema.

En la última sección, y la más extensa, hemos abordado el problema del reconocimiento de gestos por medios visuales. Se han presentado numerosos trabajos que abordan las funciones necesarias principales del proceso: percepción, segmentación, reconocimiento y clasificación del gesto. Se han presentado diferentes técnicas que permiten identificar y reconocer gestos concretos de las manos o del cuerpo entero, destacando el uso de redes neuronales convolucionales para el reconocimiento de las partes principales de un objeto o un ser humano. También se ha reflexionado sobre la importancia que juega el aspecto temporal en el reconocimiento cuando los gestos no pueden ser tratados como posturas aisladas sino como partes de una secuencia más compleja, y en concreto se han analizado las técnicas que se están empleando en la actualidad basadas en arquitecturas que combinan varias redes para la captura de características espaciales y temporales, destacando las redes convolucionales 3D, especialmente interesantes para la clasificación de video. Por último se ha hecho una clasificación atendiendo a si las técnicas se basan en el uso de hardware adicional o muy específico, como sensores en los brazos del operador o cámaras de profundidad que permitan capturar información tridimensional además de una imagen de video en 2D.

Dado nuestro problema objetivo podemos concluir que la posición de los dedos de las manos o las expresiones faciales no juegan un papel relevante en la clasificación de los gestos aquí estudiados. Además, tal y como hemos visto anteriormente las piernas no codifican información en ningún caso y que por lo tanto la herramienta a plantear debería centrarse únicamente en el tronco superior. Por otro lado no es necesario codificar, en principio,

información tridimensional para resolver la clasificación. Esto es conveniente ya que permite no añadir hardware adicional, como una cámara de profundidad o algoritmos más complejos que nos permitan distinguir gestos en tres dimensiones, y trabajar con simples capturas de una cámara RGB.

Una vez analizado en mayor profundidad el dominio del problema a resolver y explorado el estado del arte actual para conocer qué técnicas son las más interesantes, debemos ponerlo en contexto con los requisitos y supuestos previamente identificados de nuestro sistema objetivo. Tras este ejercicio, la combinación de técnicas y arquitectura que nos parece más idónea para el reconocimiento de las “señales de rampa” es una implementación basada en 3 herramientas independientes:

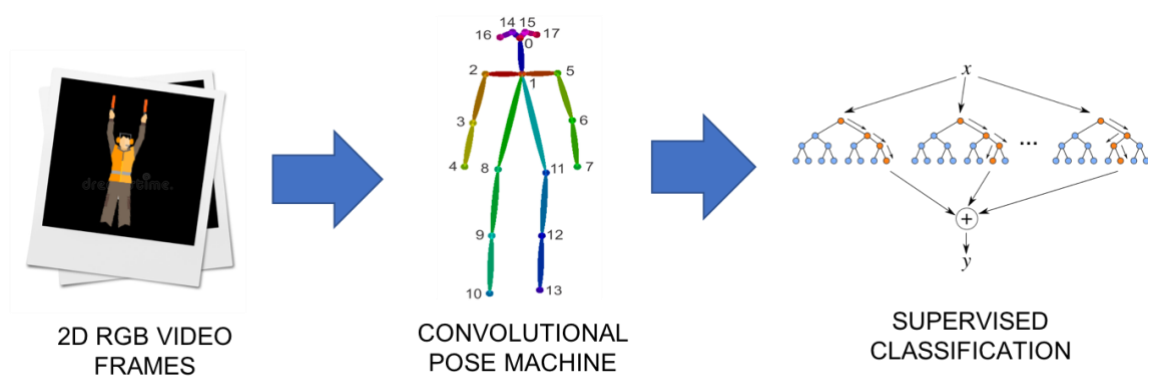


Figura 9. Arquitectura basada en 3 fases propuesta. *Fuente:* Elaboración propia

- Captura:** Hemos identificado que una solución escalable y fácilmente integrable debe poder funcionar con cámaras RGB normales, que tengan una calidad de imagen y velocidad de captura suficiente para poder proporcionar resolución espacial y temporal aceptable. Con esto evitamos dependencias con sistemas basados en sensores de profundidad, como “*MS Kinect*”, el uso de sensores específicos que deban ser incorporados a la equipación del personal de tierra o algoritmos más complejos que permitan extraer información de profundidad de una captura en dos dimensiones.
- Segmentación/Extracción de la postura:** El método empleado para la segmentación y extracción de la postura se basará en redes convolucionales profundas pre entrenadas, específicamente en las denominadas “*Convolutional Poses Machines-CPM*”. Concretamente usaremos el modelo “*ResNet-18*” (He et al., 2016) pre entrenado sobre el conjunto de datos “*MS-COCO*” (Lin et al., 2014), que nos generará un esqueleto de 18 articulaciones. Consideremos que esta solución es un compromiso entre complejidad y resolución espacial. Los modelos pre-entrenados son beneficiosos para nosotros por muchas razones. En primer lugar nos ayuda a mitigar la carencia de un conjunto de entrenamiento específico para las “señales de pista”. Además nos ayuda a ahorrar tiempo de entrenamiento y recursos de computación.

- **Clasificación:** La clasificación se abordará desde diferentes enfoques de aprendizaje supervisado para determinar que método es el más apropiado, poniendo especial interés en los métodos de ensamble (Breiman, 2001), que han mostrado excelentes actuaciones en problemas similares en los que la cantidad de datos de entrenamiento es muy limitada (Castillo et al., 2019). Cabe resaltar que el problema de clasificación no tomará como entrada “imágenes”, ya que estas han sido procesadas en el paso anterior para generar información vectorial sobre la posición que ocupan las diferentes articulaciones del cuerpo. Además de las propias coordenadas espaciales de las articulaciones (x,y), durante el subproceso de normalización de los datos previo a la clasificación se podrá valorar añadir al espacio de características otras nuevas variables resultado de la operación de las coordenadas espaciales entre varios fotogramas. Si además tenemos en cuenta la “cantidad de imágenes por segundo” a la que estamos ejecutando la captura, nos permite realizar una aproximación de la velocidad (dx/dt , dy/dt) con la que se ejecutan los gestos.

Es importante resaltar aquí, que en el problema de clasificación se abordará como un paso posterior e independiente al de la segmentación y extracción de la postura. Somos conscientes de que otros (Molchanov et al., 2016) han presentado arquitecturas basadas en redes neuronales complejas que abordan ambos pasos de manera simultánea. Sin embargo, se ha comprobado que estas arquitecturas son muy demandantes computacionalmente y que o bien no están diseñadas para su ejecución en tiempo real, o bien requieren de una mayor potencia computacional de la prevista en los supuestos de este trabajo, en el que se plantea la integración en vehículos aéreos autónomos de dimensiones acotadas. Es por ello, que al dividir el proceso en tres etapas, nos permite una mejora en la eficiencia del sistema para su operación en tiempo real, al adecuarse más a la capacidad computacional de un sistema embebido.

Uno de los puntos potencialmente más conflictivos, será el de elaborar un “conjunto de entrenamiento” de los gestos a identificar para poder llevar a cabo el paso de la clasificación, debido a la no existencia de conjuntos de datos preparados para tal fin. Para solventar este problema, y tras analizar los buenos resultados obtenidos por otros autores (Song et al., 2011) llevando a cabo procesos similares, se propone generar un pequeño “dataset” de los gestos deseados específicos, etiquetados a mano durante el momento de la captura, que tenga en consideración diferentes iluminaciones y tamaños corporales. Sobre este conjunto de datos se llevarán a cabo diferentes estrategias offline que permitan aumentar su tamaño, por medio de pequeñas rotaciones u otras transformaciones que en ningún caso comprometan la información del gesto, como podría ser una simetría.

Respecto a la solución completa, no se han encontrado en catálogos o manuales información de sistemas “*Commercial Off-The-Shelf - COTS*” similares. Sin embargo, los principales fabricantes están ya implementado soluciones de este tipo en las plataformas más grandes y avanzadas de sus portafolios. Por ejemplo, en la imagen que se muestra a continuación se aprecia a un señalero con uniformidad militar de la Marina de los EEUU haciendo “señales de rampa” a un vehículo no tripulado modelo “*Boeing MQ-25 Stingray*”, que realizó su vuelo inaugural a finales del año 2019.



Figura 10. Señalero ante un UAV “Boeing MQ-25 Stingray”. Fuente: [Boeing. Domino Público](#)

Por otro lado, tampoco se han encontrado numerosas referencias académicas que aborden este objetivo específico, es decir, el del reconocimiento de las “señales de rampa”. Destaco las contribuciones realizadas por los dos siguientes trabajos (Choi et al., 2008) y (Singh et al., 2005). Ambos son artículos relativamente antiguos y su estrategia para abordar el problema se basaban en la captura por medio de una cámara RGB y clasificación posterior de los gestos mediante “*Método de subsecuencia común más larga*” (Choi et al., 2008) y la “*Transformada de Radon*” (Singh et al., 2005).

También encontramos una publicación científica del MIT (Song et al., 2011) financiada por fondos de la Marina de EEUU que aborda el reconocimiento de las señales utilizadas en los portaaviones de la OTAN. El enfoque se basa en sensores de profundidad y el uso de la técnica “*Motion History Images - MHI*”. En este trabajo destaca además, el uso de dos sistemas para clasificar gestos tanto de la mano, como del cuerpo entero.

Sin embargo, como ha quedado de manifiesto si que existen numerosos trabajos y herramientas disponibles para apoyar el desarrollo de este trabajo basados en la interacción “hombre-máquina”, la visión artificial, el reconocimiento de gestos y las técnicas de clasificación supervisadas.

Así pues, se considera que el reto planteado es de actualidad e interés para resolver un problema conocido de un sector emergente con impacto en nuestra sociedad. Además, se ha demostrado que el empleo de estrategias basadas en las nuevas herramientas y técnicas disponibles para el procesamiento de imagen y aprendizaje profundo, más precisas y no dependientes de hardware adicional o sensores de profundidad, presentan una innovación objetiva respecto a las publicaciones anteriores sobre la misma temática y que a diferencia de estas, el resultado debidamente escalado, podría ser de aplicación industrial directa, al respetar la solución propuesta los supuestos identificados en cuanto a tamaño, peso y capacidad computacional disponible en un pequeño ordenador embebido, para ser ejecutada la solución en tiempo real a bordo de una plataforma aérea no tripulada de mediano tamaño.

3. Objetivos y metodología de trabajo

Una vez presentado el dominio del problema así como tras identificar y evaluar el conocimiento del campo aplicable al mismo, y teniendo siempre presente que este trabajo se circunscribe a la realización de un “*Trabajo Fin de Máster*” para la obtención de la titulación del Máster de Inteligencia Artificial cuyo plan de estudios especifica una dedicación de 12 ECTS, presentamos a continuación el alcance real del proyecto identificado por sus objetivos general y subobjetivos específicos.

3.1. Objetivo general

Disponer de un conjunto formado por una cámara estándar RGB y una unidad embebida de procesamiento de tamaño reducido y capacidad de procesamiento limitado que pudiera ser instalado en vehículos aéreos no-tripulados, que permita identificar y clasificar con un porcentaje de acierto de más del 80% un número limitado de gestos en tiempo real.

3.2. Objetivos específicos

Para conseguir el alcance identificado, será necesario alcanzar un nivel de desarrollo satisfactorio en los siguientes campos:

1. Explorar el estado del arte sobre el reconocimiento de gestos y aplicarlo al dominio de la identificación de las señales ejecutadas por el personal de tierra en las operaciones aeroportuarias.
2. Identificar el conjunto de hardware mínimo necesario que cumpliendo los requisitos identificados, posibilite la ejecución del software generado en el proyecto y a su vez permitiese ser instalado en una plataforma aérea de pequeño tamaño.
3. Diseñar una solución que se ejecute en tiempo real, clasificando las posturas que captura la cámara de video con una cadencia no superior a los 2 segundos de retraso, en alguna de las 5 categorías identificadas.
4. Generar un Dataset propio de entrenamiento con los gestos a identificar debidamente etiquetados. Este deberá estar compuesto al menos por datos extraídos de dos hombres y una mujeres en edad adulta, que ejecutarán cada uno de los 4 categorías de gesto al menos 100 veces.
5. Cuantificar y evaluar la adecuación a los requisitos identificados y el rendimiento del sistema, debiendo tener este un porcentaje de acierto en la clasificación superior al 80%.

3.3. Metodología del trabajo

Para el presente proyecto se plantea un desarrollo siguiendo un modelo “en Espiral”, en las que se repetirán fases consecutivas, e incrementales en dificultad, de: planificación, ejecución y evaluación de los resultados. Esta metodología, fruto de una combinación entre el modelo tradicional “Lineal” y el modelo “Iterativo”, se cree que se ajusta muy bien al presente trabajo debido principalmente a:

- Restricción temporal fuerte, marcada por las consecutivas fechas de entrega, lo que permite disponer de entregables a lo largo de toda la vida del proyecto.
- Incertidumbre asociada a un proyecto de investigación en el que de partida no se puede identificar la ruta que conducirá al éxito de las varias alternativas identificadas.
- No existe un “cliente directo” o “interesado en el producto” resultante que pueda ser involucrado en la definición de requisitos o pueda beneficiarse de experimentar con versiones tempranas del sistema.

La ventaja que proporciona esta metodología de trabajo es que es un modelo evolutivo con una concepción del proyecto de tipo experimental “de abajo hacia arriba” lo que permite iterar en ciclos crecientes en dificultad, generando entregables en cada iteración. Esto ayuda a identificar riesgos desconocidos antes de tiempo, y enfrentar los desarrollos más complejos como una colección de proyectos más simples y acotados, con la seguridad de que antes de iniciar un desarrollo más complejo, ya se dispone al menos de las evidencias necesarias para justificar el tiempo y trabajo invertido.

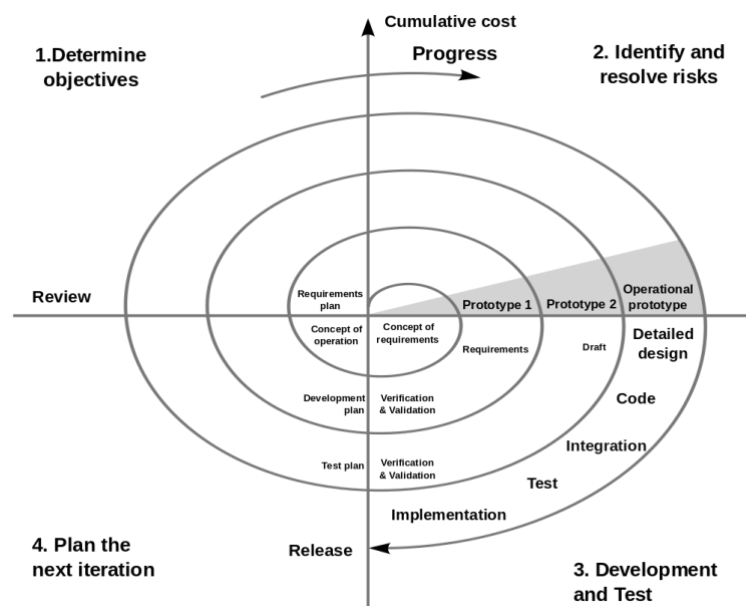


Figura 11. Modelo de desarrollo en Espiral. Fuente: [Boehm, Dominio Público](#)

En cada nuevo ciclo de trabajo se repetirán las siguientes subfases:

- **Planificación:** Una vez alcanzados los objetivos del anterior ciclo y decida el alcance o meta del siguiente, se deberá hacer una planificación que tenga en cuenta el alcance deseado, el plazo de tiempo disponible, los entregables deseados y los riesgos principales que se deberán enfrentar.
- **Implementación:** Una vez decidido el plan de ataque se ejecutarán todas las tareas identificadas, generando los resultados y entregables.
- **Evaluación:** Antes de continuar hacia la siguiente iteración se deberá evaluar los resultados obtenidos, plazo restante para la siguiente entrega, estado de la solución respecto al objetivo final y dificultad del siguiente paso, para decidir si merece la pena seguir con la experimentación de nuevos métodos o pasar al cierre.

Tras la ejecución del último ciclo se pasará a una ultima subfase de “Cierre” en la que se terminarán de recolectar las evidencias y de preparar los entregables finales que deberán poner de evidencia los resultados obtenidos así como el trabajo experimental invertido.

3.4. Cronograma de fases y tareas principales.

Durante la ejecución del proyecto se han ejecutado las siguientes fases:

[FASE1] ***Análisis del “Estado del Arte”:***

- Comprensión del dominio del problema.
- Comprensión de las técnicas y herramientas aplicables a su resolución.
- Selección de literatura más relevante.
- Desarrollo de entregables del proyecto.

[FASE 2] ***Gestión del Alcance:***

- Definir objetivo general y específicos.
- Recopilar requisitos.
- Adquirir equipos y herramientas del proyecto
- Elaboración plan de trabajo.
- Desarrollo de entregables del proyecto.

[FASE 3] *Formación específica para el proyecto:*

- Técnicas y herramientas genéricas para Visión Artificial (Open CV)
- Estudio detallado de las técnicas y herramientas identificadas resolución.
- Preparación del hardware seleccionado, y pruebas unitarias.
- Desarrollo de entregables del proyecto.

[FASE 4] *Creación Dataset específico:*

- Definir plan de ejecución.
- Desarrollo de la herramienta de captura y etiquetado.
- Toma de muestras.
- Aumento del volumen de muestras aplicando técnicas a posteriori (“Data Augmentation Techniques”)
- Procesamiento, limpieza y preparación del Dataset.
- Desarrollo de entregables del proyecto.

[FASE 5] *Implementación de la solución. Metodología incremental:*

- Desarrollo esqueleto básico del software.
- Implementación “Convolutional Pose Machine” sobre imagen en tiempo real.
- Estrategias para el pre-procesamiento y limpieza de los datos capturados.
- Implementar y evaluar rendimiento con diferentes técnicas de clasificación basadas en el espectro espacial:
 - Método de Ensemble: Random-Forest.
 - Red Neuronal “Feedforward” independiente como clasificador.
 - “Transfer Learning” sobre capa pre-entrenada profunda.
- Evaluar implementación de técnicas basadas en el espectro espacial y temporal:
 - Técnicas basadas en “Dynamic Time Warping”.
 - Red Neuronal Recurrente: LSTM.

- 2 x CNN paralelas, una para la captura de espectro espacial y otra temporal.
- Red Neuronal Convolutiva en 3D: 3D-CNN
- Desarrollo de entregables del proyecto.

[FASE 6] **Cierre de proyecto:**

- Análisis de la adecuación del resultado final a los requisitos de partida.
- Discusión de los resultados.
- Elaboración del “Artículo Científico”.
- Cierre de los entregables.
- Preparación de la defensa.

3.5. Materiales y métodos.

En esta sección se hace una relación de los recursos, hardware y software, que se han empleado en la elaboración de este proyecto, a fin de que los resultados expuestos puedan ser replicados.

RECURSOS HARDWARE:

- ✓ **Dev KIT NVIDIA Jetson Nano**®. Pequeño ordenador embebido específicamente diseñado para la ejecución de tareas que impliquen uso de técnicas en “Inteligencia Artificial”. Sus características principales son:



Figura 12. Características principales Jetson Nano. Fuente: [NVIDIA](#)® Tech Ref.

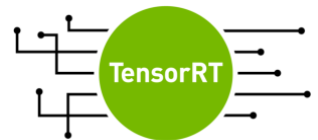
- ✓ **Webcam C920HD Logitech®:** Cámara RGB de alta resolución con salida USB 2.0. Sus características principales son:



Figura 13. Características principales C920HD. Fuente: Cortesía [Logitech®](#).

RECURSOS SOFTWARE:

- ✓ **NVIDIA TensorRT®:** JetPack 4.3.SDK para el desarrollo de aplicaciones que requieran de aprendizaje profundo de alto rendimiento. Incluye un API así como optimizadores especialmente desarrollados para “Deep Learning”, consiguiendo baja latencia. Está desarrollado en CUDA®. Integra nativa mente los principales Frameworks de trabajo. Logo TensorRT. Fuente: [NVIDIA®](#)



- ✓ **TensorFlow 2.0:** Conjunto de librerías de programación pensadas para facilitar el desarrollo e implementación eficiente de aquellas tareas de computación numérica que se beneficien de ser resueltas mediante la aplicación de grafos. Es “open-source” y destaca su uso en aplicaciones de Machine Learning, especialmente para “Deep Learning”. Logo TensorFlow. Fuente: [TensorFlow](#).



- ✓ **Tensor RT Pose Estimation:** Proyecto open-source que implementa una “Convolutional Pose machine” pre-entrenada especialmente optimizada para su ejecución en TensorRT y dispositivos NVIDIA. Permite la identificación en tiempo real de posturas de un humano devolviendo las coordenadas de los puntos clave del cuerpo humano en formato MSCOCO (Lin et al., 2014), dataset con el que ha sido entrenada la red. Proyecto se encuentra alojado en GitHub: https://github.com/NVIDIA-AI-IOT/trt_pose
- ✓ **Python 3.8.2:** Todo el código de este proyecto se ha desarrollado en este lenguaje de programación de código abierto. Existe un gran consenso en el ámbito científico y técnico del sector para su uso en “IA”. Python. Fuente: [Python](#)



4. Descripción de los requisitos del sistema.

4.1. Casos de uso del Sistema.

Los casos de uso del sistema, o también denominados conceptos de operación, persiguen describir como el sistema final, integrado en la plataforma, debería ser utilizado por los usuarios.

En este sentido hemos identificado varios escenarios posibles en los que el sistema podría ser utilizado, aportando un valor añadido al usuario. En todos ellos es importante resaltar que el sistema ha sido concebido para ser integrado en un sistema más complejo y de mayor tamaño. Por lo tanto para un caso de uso real, debemos tener en cuenta que harán falta interaccionar con otros equipos tales como paneles de alarma o sistemas de control.

Los tres escenarios más probables en los que la utilización de un sistema de este tipo podría aportar valor y como operaría se describen a continuación:

- *Vehículos aéreo NO tripulados* En este tipo de plataformas en las que no hay tripulantes humanos a bordo, el sistema permitiría al vehículo operar en los mismos espacios que el tráfico tripulado al permitirle adaptarse a la señalética visual ya existente y diseñada para uso entre humanos. Este tipo de sistemas facilitarían la introducción de dichos vehículos autónomos en aeropuertos y aeródromos. En este caso, el sistema actuaría como subsistema de carga de pago o de tipo “payload”. Estaría conectado al “Mission Control Computer - MCC” al cargo de las acciones de control y guiado durante la rodadura, y por medio de un protocolo de comunicaciones (ICD) previamente acordado entre ambos sistemas le enviaría información periódica sobre los gestos detectados, para que el “MCC” actuase acorde sobre los actuadores. En esa interfaz de comunicaciones se debería definir una cabecera que permitiese identificar los diferentes mensajes, un campo con una marca temporal que ayudase a identificar mensajes caducados así como un “Código de Redundancia Cíclica - CRC” que permitiese a ambas partes comprobar la integridad del mensaje transmitido. Por otro lado, en los campos reservados para la transmisión de la información, se debería incluir un campo que codificase el gesto detectado, otros con información adicional (e.g. etiqueta de velocidad de ejecución) así como unos campos de alarma que permitan al sistema de visión comunicar su estado al “MCC” avisándole de fallos o situaciones en las que no se encuentre operativo.
- *Vehículos aéreo tripulados:* En este tipo de plataformas, en las que si hay uno o varios tripulantes humanos en cabina, el sistema funcionaría aportándoles información adicional que ayudase a mejorar su conciencia situacional, aliviando su carga de trabajo e incluso alertando de posibles acciones que conlleven un riesgo inminente. En este caso, el sistema funcionaría como parte del subsistema de ayuda al rodaje en pista, haciendo labores de supervisión y monitoreo. Cuando detectase la

presencia de un señalero en su campo de visión haciendo gestos, alertaría por medio de alguna señal acústica o luminosa a los pilotos hasta que estos confirmasen con una acción que lo tiene en contacto visual. En esta segunda fase, el sistema aportaría información sobre el gesto reconocido, ayudando a los pilotos a decidir que información está codificando el operario de tierra en caso de duda o problemas de visibilidad. Por último, en caso de existir una discrepancia entre el gesto identificado y la acción de control del piloto humano (e.g. Se detecta gesto de “Stop” pero el Piloto no detienen el avión) el sistema alertaría nuevamente a los tripulantes con una alarma sonora o luminosa, hasta que estos confirmasen que están al tanto. Nuevamente, sería necesario establecer un interfaz de comunicación entre el sistema de visión y los otros sistemas que permitan recibir información del estado de la aeronave y envía notificaciones de alerta.

- *Vehículos terrestres de servicio*: En esta categoría catalogamos a todas aquellas plataformas terrestres que operan en el entorno aeroportuario en labores de “handling o asistencia en tierra”. Existen diferentes tipos y con misiones tan diversas como el facilitar el traslado de aeronaves (“tractores de arrastre”), movimiento de pasajeros (“jardineras o autobuses”) o movimiento de mercancías (“porta-maletas o catering”). Dado el elevado número de vehículos de este tipo que operan diariamente en las calles de rodadura y el gran número de incidentes en los que se ven involucrados (golpes a aviones, invasión de pistas, retrasos por fallo de orientación etc..) una solución basada en la integración de sistemas de este sentido podría tener un gran interés en la industria, al avanzar hacia el desarrollo de plataformas autónomas o servir como sistemas de supervisión y alerta en aquellos vehículos conducidos por un humano.

4.2. Requisitos Generales del Sistema.

Las características principales del sistema se pueden clasificar en las siguientes categorías atendiendo a su tipología. Cabe señalar, que al tratarse de una prueba de concepto, muchos de los requisitos generales aquí identificados deberían ser revisados una vez decida la integración del sistema en una plataforma final.

✓ [1] ***REQUISITOS FUNCIONALES:***

[1.1] El sistema será capaz de reconocer y clasificar correctamente al menos el 80% de “las señales de rampa” especificadas.

[1.2] El sistema deberá funcionar en tiempo real, con una cadencia no superior a 2 segundos desde que se inicia el gesto.

[1.3] El sistema será capaz de identificar al menos 5 tipos de gestos diferentes:

- [1.3.1] Stop.
- [1.3.2] Avance
- [1.3.3] Giro Derecha
- [1.3.4] Giro izquierda
- [1.3.5] No-Gesto. Cuando no detecte alguno de los anteriores.
- [1.4] El sistema basará su funcionamiento en el uso de información puramente visual, capturada por una cámara RGB.
- [1.5] El sistema deberá ser capaz de funcionar correctamente en aquellas condiciones ambientales (iluminación diurna, campo de visión etc.) en las que el propio ojo humano pueda reconocer el gesto.
- [1.6] El sistema deberá ser capaz de funcionar incluso cuando el operador no esté totalmente enfrentado a la cámara, pero su posición si permita detectar el gesto para un observador humano.
- [1.7] El sistema deberá ejecutarse a una velocidad/frecuencia tal que permita la captura y clasificación de los gestos en su velocidad normal de operación en al vida real.
- [1.8] El sistema deberá tener un tamaño y peso que permitan ser instalado en vehículos autónomos no tripulados de categoría específica (MTOW >150Kg).
- [1.9] El sistema estará compuesto de dos dispositivos independientes y diferenciados:
 - [1.9.1] Cámara RGB: Dispositivo de captura de imagen. Situado exterior plataforma.
 - [1.9.2] Unidad Procesamiento: Tareas reconocimiento y clasificación. Interior.
- [1.10] El sistema no podrá consumir más energía que todo el resto de aviónica (<10W@12V).
- [1.11] El sistema deberá mantener una temperatura que no perjudique la operación del resto de sistemas (<45°C)
- [1.12] El sistema funcionará de manera automática, detectando y clasificando los gestos.
- [1.13] El sistema permitirá el envío de información a otros subsistemas por medio de un canal Serie o Ethernet, con un protocolo de comunicaciones a determinar.
- [1.14] La comunicación será de tipo periódica, a una frecuencia de al menos 5 Hz.

[1.15] El sistema permitirá su operación en modo demostración. Este modo se caracterizará por:

[1.15.1] Estar conectado a un monitor de ordenador.

[1.15.2] Poder ser controlado por un usuario mediante teclado.

[1.15.3] Permitir acceso por SSH.

[1.15.4] Mostrar en pantalla la imagen capturada

[1.15.5] Mostrar en pantalla la etiqueta del gesto clasificado.

✓ **[2] *REQUISITOS -NO- FUNCIONALES:***

[2.1] El sistema usará como red convolucional para el reconocimiento de gestos la arquitectura ResNET-18.

[2.2] El sistema usará como pesos de la red un modelo pre entrenado contra “MS COCO” dataset.

[2.3] El sistema usará un “esqueleto” compuesto de solo 10 de los 18 puntos singulares o “keypoints” disponibles, que son los que se corresponden con las extremidades y tronco superior.

[2.4] Cada punto singular contendrá al menos la siguiente información:

[2.4.1] Identificador de articulación/keypoint.

[2.4.2] Coordenadas espaciales en 2 dimensiones X e Y referidas al punto 0.

[2.5] El sistema deberá procesar esta información a al menos 10 FPS.

[2.6] La clasificación del gesto se basará en una técnica de aprendizaje supervisado, basada en la información obtenida “frame a frame”.

[2.7] Existirá un sistema de filtrado que evite cambios “rápidos” de etiqueta entre frames consecutivos.

[2.8] Con el fin de entrenar el clasificador se generará un dataset propio con las siguientes características:

- [2.8.1] Contendrá datos de los 5 gestos: stop, avance, giro-derecha/izquierda y “none”.
- [2.8.2] Los gestos serán ejecutados por al menos 2 hombres y 1 mujer.
- [2.8.3] Cada gesto se repetirá al menos 100 veces.
- [2.8.4] Se aplicará “CPM” sobre cada fotograma durante la captura.
- [2.8.5] Los datos serán anonimizados y no permitirán identificar al usuario.
- [2.8.6] No se guardaran las imágenes RGB.
- [2.8.7] Cada fotograma será etiquetado con la etiqueta del gesto al que pertenece.
- [2.9] El dataset generado se publicará de forma abierta.
- [2.10] El sistema será escalable de tal manera que puedan añadirse nuevos gestos en versiones posteriores.

✓ **[3] *RESTRICCIONES TÉCNICAS:***

- [3.1] El sistema se desarrollará en Python 3.X.
- [3.2] El sistema empleará NVIDIA® TensorRT, JetPack 4.3.
- [3.3] El sistema usará como cámara un webcam RGB con foco automático.
- [3.4] La cámara dispondrá de conexión por USB.
- [3.5] La cámara tendrá al menos una resolución de 720p / 30FPS.
- [3.6] El sistema usará como unidad de procesamiento un sistema embebido NVIDIA® Jetson nano.
- [3.7] El SW se ejecutará sobre un SO. Linux TEGRA (LT4)
- [3.8] El sistema se alimentara a 5v@4A máximo.

4.3. Arquitectura del sistema.

En esta sección, introduciremos la arquitectura final que compondrá nuestro sistema tanto desde le punto de vista hardware como de software, mediante la identificación y asignación de funciones a cada elemento que componen el flujo de procesamiento del método desarrollado.

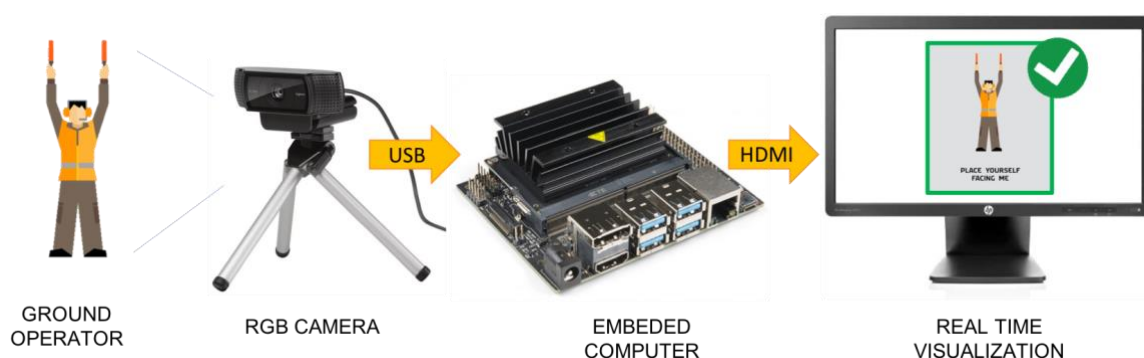


Figura 14. *Arquitectura HW embebida*. Fuente: Elaboración propia.

Tal y como se ha venido presentando, el “setup” que usaremos en este proyecto consta de 3 elementos principales:

- ✓ **Cámara RGB:** Estará montada en un trípode enfocando hacia el sujeto. Se conectará por USB al ordenador embebido.
- ✓ **Ordenador Embebido:** Ejecutará los diferentes softwares empleados, tanto para la generación del Dataset como para la demostración en tiempo real. Recibirá el video digital desde la cámara por USB, y mostrará información visual en el monitor por medio de un puerto HDMI o por una acceso SSH mediante otro dispositivo.
- ✓ **Monitor/PC externo:** Este elemento es auxiliar y solo tiene sentido durante las pruebas de validación y durante la demostración para mostrar los resultados de una forma eficiente. En un contexto de aplicación real, no se contaría con este elemento.

Este mismo montaje se utilizará para dos propósitos diferentes durante el proyecto, pero ejecutará software diferentes en cada caso:

- **Generación Dataset:** Este mismo sistema se empleará para capturar y etiquetar los gestos realizados por los diferentes voluntarios, que tendrá como objetivo generar un conjunto de datos de entrenamiento y validación que serán la base para el modelo de clasificación. En este caso el software que se ejecutará en el ordenador embebido permitirá la captura y etiquetado de fotogramas mediante una interfaz de usuario sencilla, en la que se le podrá indicar el gesto que se está ejecutando. Es importante señalar de nuevo que el conjunto de datos que se guardarán no serán las imágenes sino directamente las coordenadas de las articulaciones tras procesar los fotogramas capturados por nuestra red “CPM”.

- **Operación normal:** Esta misma arquitectura se empleará posteriormente durante al ejecución normal del sistema y su demostración. En este caso, además de realizar las funciones de captura e identificación de las coordenadas de las articulaciones por medio del método “*Convolutional Pose Machine*”, el sistema también llevará acabo la clasificación del gesto por medio de la técnica de aprendizaje supervisado seleccionada. En este caso, ya no será necesaria la intervención del operador, sino que el sistema funcionará de manera automática tras su puesta en funcionamiento procediendo a la segmentación, identificación y clasificación del gesto. La información sobre la etiqueta inferida así como otros datos útiles, tales como el número de frames por segundo procesados, se añadirán sobre la propia imagen de video capturada en tiempo real y se mostrarán todos ellos en el monitor auxiliar.

Atendiendo a la arquitectura software del programa en su operación normal, podemos identificar 6 funciones iterativas claramente diferenciadas con objetivos específicos cada una, pero cuyas entradas/salidas estarán vinculadas según se muestra en el diagrama de flujo de la siguiente imagen:

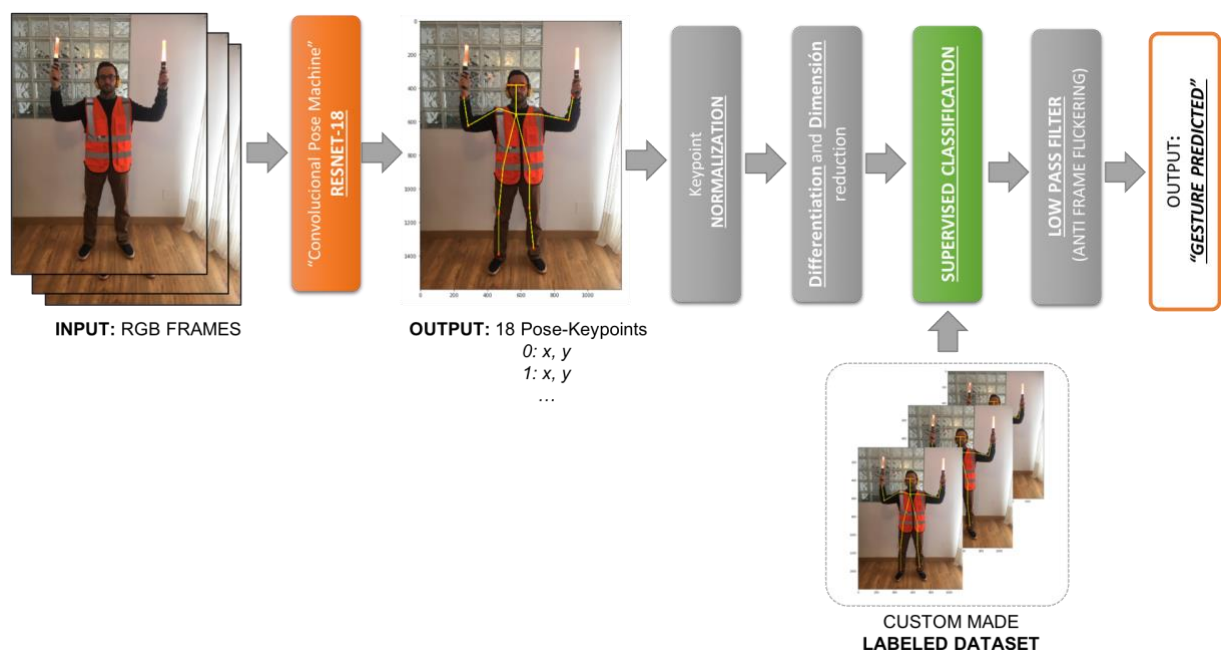


Figura 15. Arquitectura SW empleada. Fuente: Elaboración propia.

1. **Captura de imagen:** Esta función engloba las tareas de captura de video. Quedarían aquí incluidos los trabajos de pre-procesamiento de la imagen iniciales. Cabe destacar que si bien se utiliza una cámara de video, el proceso se realizará de manera discreta sobre las imágenes extraídas como “fotogramas” del propio video.
2. **“Convolutional Pose Machine – ResNET-18”.** Mediante la aplicación de una red neuronal pre-entrenada y de uso abierto se llevará acabo identificación de las

estructuras articulares parametrizadas del cuerpo humano. Con este paso, transformaremos la imagen en una serie de variables numéricas que representan los movimientos de las articulaciones, transformando el conjunto de “features” del dominio de los pixels-imágenes al de vectores espaciales, simplificando los cálculos posteriores. La red escogida ha sido ResNET-18 (He et al., 2016), una efectiva red neuronal de 18 capas de profundidad que sigue una novedosa arquitectura de tipo “Residual” que incorpora una serie de conexiones directas entre capas no consecutivas. Debido a estas características especiales, este tipo de redes son fáciles de optimizar y con buenas propiedad predictivas, proporcionando un equilibrio de rendimiento y eficiencia para ser ejecutado el en hardware embebido propuesto.

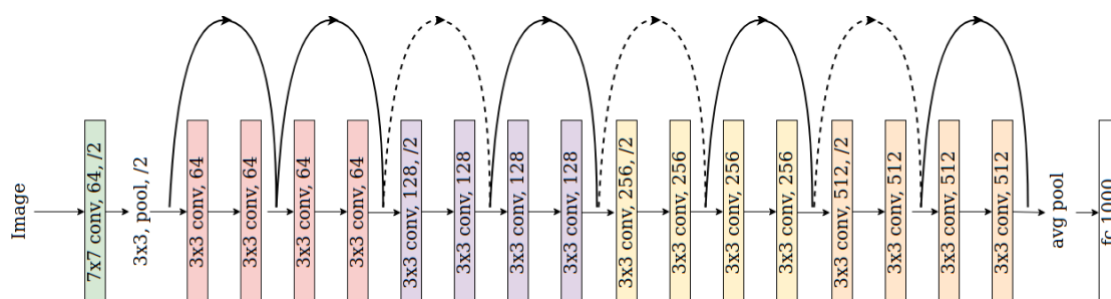


Figura 16. Esquema ResNET-18. Fuente: He et al., 2016.

3. **Normalización:** Los puntos singulares obtenidos en el paso anterior están en ejes “imagen” y por lo tanto antes de seguir con el proceso se deben normalizar, para hacerlos invariantes a la escala o la traslación. De esta manera hacemos que las coordenadas no sean dependientes de la posición que ocupó la persona respecto de la cámara. Para ello se transformarán todas las coordenadas de tal manera que el nuevo origen de referencia sea el punto “0” que se corresponde con el cuello.
4. **Diferenciación y Reducción de la dimensionalidad:** Tal y como se ha venido indicando, para los gestos que este proyecto intenta clasificar, solo los puntos correspondientes a las extremidades y tronco superior son relevantes, pudiendo desestimar el resto de información. En este paso también se podría calcular la deriva como la variación o diferencia del desplazamiento de cada coordenada respecto al fotograma anterior, dividiendo así u vez por la velocidad de obturación, es decir, el número de frames por segundo.
5. **Clasificación:** Englobamos aquí las funciones finales que permitirán clasificar el gesto capturado por la cámara, y traducido mediante “CPM” en coordenadas, en una

de las categorías previamente definidas. Se propondrán y compararán diferentes métodos, escogiendo aquel que mejor se adapte a las restricciones propias del problema, y permita una ejecución rápida. Como conjunto de entrenamiento y validación emplearemos un dataset de elaboración propia y etiquetado de forma manual con solo los gestos requeridos.

6. **Filtro Paso Bajo:** La parte final de esta etapa ejecutará las lógicas de ponderación y filtros paso-bajo que permitan mantener una salida continua entre las diferentes categorías, minimizando efectos de parpadeo por clasificación en categorías diferentes de frames consecutivos.

5. Diseño, desarrollo y evaluación de la solución técnica implementada .

Si bien en el segundo capítulo, tras el análisis del estado del arte ya se esbozaba el conjunto de técnicas preliminarmente seleccionadas, ha sido en la última sección del capítulo cuarto, en el que se ha presentado la arquitectura final empleada tanto hardware, pero principalmente software. De especial interés para entender este quinto capítulo, es la asignación de funciones de alto nivel anteriormente presentada y que identifica de manera clara y precisa, tanto la estrategia seguida, como el rol que desempeñará cada función de la herramienta.

En este quinto capítulo abordaremos de una manera más detallada la implementación técnica específica seguida en este proyecto, así como los problemas detectados y soluciones planteadas, a fin de alcanzar el objetivo del proyecto, identificado por sus requisitos.

Comenzaremos haciendo una presentación de la herramienta software final, introduciendo cada una de sus secciones y funcionalidades principales, que servirán para entender y seguir el resto de explicaciones.

A continuación, abordaremos la elaboración del dataset, analizando la captura, número de muestras y pre-procesamiento de los datos en bruto que permiten obtener este entregable final, disponible para la reproducción y validación de los resultados de este trabajo.

Concluido este trabajo, presentaremos y justificaremos las decisiones tomadas con el objetivo de adaptar nuestro dataset a la forma y características requeridas para el entrenamiento de los modelos de aprendizaje supervisado. En este sentido, se presentarán y compararán dos modelos diferentes, analizando con especial cuidado la elección de hiperparámetros de cada uno de ellos. Los modelos propuestos se basan en una implementación “Random Forest” y un modelo basado en una “Red Neuronal Artificial Feedforward”.

Tras el anterior ejercicio de comparación de las características “offline” de ambos modelos sobre las muestras de nuestro conjunto de datos, se pondrán a prueba en un entorno de ejecución en tiempo real, en el que se evaluará las capacidades de ambos para llevar a cabo la tarea propuesta. Se prestará especial atención a la capacidad para inferir sobre un conjunto de datos no vistos anteriormente, y sobre todo, buscando el modelo más optimizado que permita que el sistema en su conjunto responda con agilidad y fluidez, características clave para la consecución de los objetivos del proyecto.

5.1. Descripción de la herramienta software desarrollada.

En esta sección presentaremos la herramienta de software desarrollada durante la ejecución del proyecto. Esta interfaz gráfica de usuario (GUI), si bien carece de sentido en operación en entorno real, es de gran utilidad para la fase de experimentación.

Siguiendo esta filosofía de herramienta para el desarrollo y mantenimiento, se decidió finalmente llevar acabo su implementación sobre un Jupyter Notebook, en lugar de codificar una aplicación autoejecutable en Python directamente.

Si bien las opciones disponibles para adaptar y customizar “GUIs” sobre este entorno de trabajo son limitadas, mediante el empleo de librerías tales como “*ipyWidgets*” se puede acceder a un reducido, pero útil, conjunto de objetos que permiten crear botones, menús desplegables o áreas de texto.

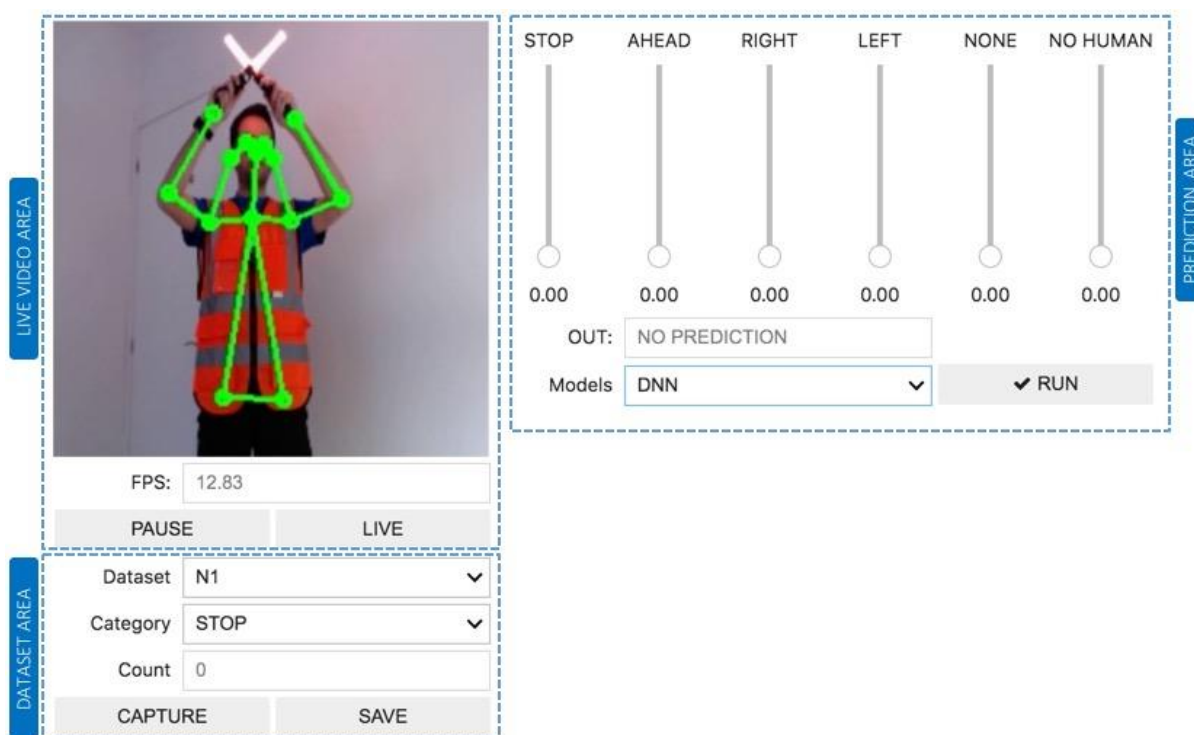


Figura 17. GUI desarrollada en el proyecto. Fuente: Elaboración Propia.

La interfaz de usuario desarrollada consta de una única pestaña que contiene todos los elementos para mostrar información e interactuar con el operador. Atendiendo a la misión de cada uno de los elementos presentantes en la interfaz podemos distinguir 3 secciones o áreas temáticas principales, marcadas en la imagen anterior con sendos recuadros azules:

- ✓ **“LIVE-VIDEO” AREA:** Esta sección es la encargada de gestionar la captura en tiempo real de las imágenes de la cámara. Está compuesta a su vez por otros 3 elementos
 - **Imagen:** Un recuadro de 224x224 que muestra la salida RGB en tiempo real que la cámara capturada a 30 FPS. Para dicha captura se emplea la librería “USB Camera” que permite un nivel de abstracción y manipulación de objetos muy útil. Sobre dicha imagen además se superpone la salida del modelo “CPM” que contiene las coordenadas identificadas del esqueleto estimado, compuesta por las articulaciones o puntos característicos del cuerpo.
 - **Campo “FPS”:** Bajo la imagen se muestra el tiempo calculado que tarda en ejecutarse el bucle principal del programa medido en “frames por segundo”. Esto es de gran interés para identificar el retardo máximo que se produce entre que se ejecuta el gesto y sistema devuelve la clase estimada. Como se verá más adelante, esta medida nos permite identificar que ciertos modelos no optimizados toman demasiado tiempo en ejecutarse, aumentado por tres el tiempo medio entre iteraciones consecutivas.
 - **PAUSE / LIVE:** Estos dos botones nos permiten detener y reanudar la salida de la cámara, así como la ejecución del “CPM” sobre la imagen capturada. Es útil en tareas de mantenimiento para evitar estar procesando imágenes no deseadas.
- ✓ **“DATASET” AREA:** Esta sección englobada todas las herramientas disponibles para la captura y etiquetado de las imágenes que usaremos para el entrenamiento y validación. Se compone de dos menús desplegables, un cuadro de texto y dos botones de acción.
 - **Menú “Dataset”:** Es un menú desplegable que permite seleccionar en que carpeta se guardarán los datos capturados. Muestra 4 opciones (N1, N2, N3 y N4) y es de utilidad para almacenar muestras de diferentes sujetos. Si la carpeta no existe, se crea al ser seleccionada.
 - **Menú “Category”:** Es un menú desplegable que contiene las cinco categorías empleadas en este proyecto. Esto permite introducir de antemano la etiqueta que vamos a capturar.



Figura 18. Detalle herramienta SW. Fuente: Elaboración propia.

- **Campo “Count”:** Este elemento informa sobre el número de muestras capturadas de cada categoría, para un usuario/dataset determinado. Se actualiza al cambiar entre las diferentes opciones de los menús disponibles.

- **Botón “Capture”**: Este botón permite tomar una “instantánea” de las coordenadas de los puntos singulares (18 coordenadas x-y) devueltos por el modelo “CPM”. Estos se añadirán a un array que contiene todas las muestras tomadas con anterioridad.
- **Botón “Save”**: Este botón permite ejecutar una copia no volátil del array que contiene todas las muestras por cada categoría y sujeto. Los datos se guardan en formato “NPY” (array binario serializado de NumPy). La copia se etiqueta de manera adecuada en la carpeta pertinente (e.g. “./N1/stop.npy”).
- ✓ **“PREDICTION” AREA**: Esta sección contiene los elementos relacionados con la elección del modelo de clasificación y su salida. Se compone a su vez de 4 elementos:
 - **Menú “Models”**: Se trata de un menú desplegable que permite seleccionar, entre los diferentes modelos entrenados, cual queremos que se ejecute.

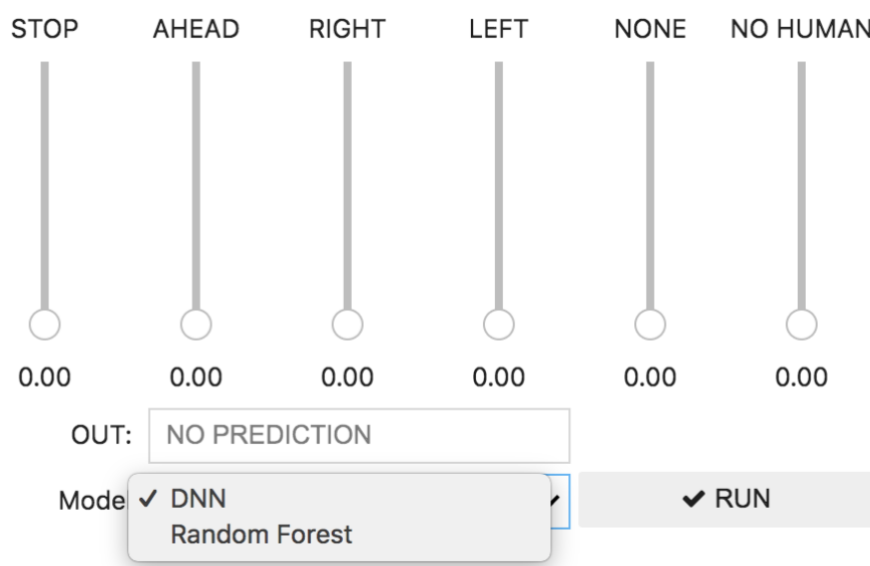


Figura 19. Detalle herramienta SW. Fuente: Elaboración propia.

- **Botón “RUN”**: Este botón permite detener o reanudar la ejecución del modelo de clasificación. Útil para tareas de mantenimiento o ajuste.
- **Campo “OUT”**: Este campo muestra la categoría más probable resultado de la predicción cuando el modelo está en ejecución. En caso de que la predicción se encuentre detenida, advertirá de este estado también.
- **Barras “Probabilidad”**: Este objeto se compone de 6 indicadores verticales de 0 a 1. Se utilizan para mostrar la probabilidad con que una de las categorías es predicha. Es de especial utilidad para comparar modelos e identificar qué movimiento producen mayor incertidumbre durante la etapa de inferencia. Destaca el uso de una sexta categoría que mide la probabilidad de que haya un “humano” en el área de visión. Este valor es resultado de la segmentación del propio “CPM” y se ejecuta antes del propio modelo de clasificación.

5.2. Generación del Dataset.

Como se ha venido explicando en esta memoria, la arquitectura software aquí propuesta se basa en la ejecución de dos modelos en serie:

- “CPM”: Un primer modelo convolucional pre entrenado que segmenta el ser humano de la imagen e infiere las coordenadas de las articulaciones.
- “ML”: Un segundo modelo que clasifica su salida en una de las 5 categorías previstas.

En esta sección abordaremos la estrategia seguida para la captura, pre-procesamiento y generación del Dataset que nos ayudará a entrenar este segundo modelo, permitiéndole clasificar el gesto capturado en una de las 5 categorías posibles.

Para ello emplearemos la “GUI” desarrollada sobre un “Jupyter Notebook” explicada en la sección anterior, ejecutándose sobre nuestro setup del proyecto (“*embed computer*”). Específicamente haremos uso de las herramientas que nos permiten capturar y etiquetar los distintos gestos realizados por los participantes.

Como se explicaba anteriormente, la herramienta está diseñada para capturar muestras de hasta 4 sujetos de manera independiente. Esto es de gran utilidad, ya que permite conducir análisis independientes sobre cada subconjunto, pudiendo analizar la respuesta del modelo al introducir nuevos ejemplos o descartar muestras, sin dañar el resto de capturas.

A su vez, la herramienta nos permite seleccionar de antemano de que “categoría” serán los gestos que vamos a capturar. Este paso es clave, ya que nos permitirá disponer de un conjunto de datos etiquetados sin un gran esfuerzo posterior.



Figura 20. Equipación utilizada durante la captura. Fuente: Elaboración propia.

Los datos se capturaron en ambientes e iluminaciones diversas, pero tal y como justificamos anteriormente en este trabajo, al utilizar una primera red convolucional pre-entrenada “CPM”, que nos permite transformar el problema desde el dominio de las imágenes/píxeles, al dominio de las coordenadas de las articulaciones de cada sujeto, el ambiente deja de jugar un papel fundamental siempre y cuando, el modelo sea capaz de identificar correctamente las diversas articulaciones. Esto es clave, ya que nos va a permitir obtener buenos resultados en la fase de inferencia, aunque el sistema se esté ejecutando en un ambiente no visto con anterioridad.

Una vez instalado el setup, se invita a entrar en la habitación a cada uno de los sujetos de forma individual, y se procede a explicarles los detalles del proceso y su implicación en el mismo. Para ilustrar cada uno de los gestos, se utilizó un video que muestra dichos gestos ejecutados por un profesional. De esta forma, se intentaba minimizar el posible sesgo derivado de ver al autor del trabajo ejecutar los gestos. Por otro lado, se les pedía que durante la captura se movieran por todo el campo de visión de la cámara y que rotaran su cuerpo, de tal manera que algunas extremidades quedasen más ocultas, o que algunos puntos (e.g. cadera, rodillas) aparecieran y desaparecieran, con la idea de que el clasificador final aprendiese a dar poca importancia a estos, centrándose en los efectores superiores.

Por otro lado, se pidió a los voluntarios llevar el equipo característico de trabajo descritos en el capítulo segundo, compuesto por una chaleco de alta visibilidad y 2 varitas luminosas. Justificamos esta decisión en: el interés por determinar si el uso de elementos reflectantes impedía la normal operación de la primer red convolucional, y a que al realizar los gestos con las “varitas de señalización” el cuerpo adopta una postura ligeramente diferente a si solo se emplean las manos.

```
#Mostramos uno de los datos
print("Longitud:", len(stopDataset[0]))
print("Sample:\n", stopDataset[0])

Longitud: 18
Sample:
[[0.47519749 0.27839831]
 [0.49778932 0.25115982]
 [0.45600948 0.25830549]
 [0.5321008  0.27554995]
 [0.43454909 0.28871176]
 [0.57479811 0.40900895]
 [0.402803   0.41639048]
 [0.6976065  0.31388822]
 [0.28192973 0.32799914]
 [0.61405438 0.16229226]
 [0.38951302 0.14774534]
 [0.55479592 0.8695296 ]
 [0.40808836 0.88541853]
 [0.         0.         ]
 [0.         0.         ]
 [0.         0.         ]
 [0.         0.         ]
 [0.48751718 0.4090299 ]]
```

Figura 21. Ejemplo información capturada por cada muestra. Fuente: Elaboración propia.

Es importante resaltar nuevamente que la información que se captura y se guarda, no son las imágenes sino los vectores de coordenadas que contienen las posiciones estimadas de las articulaciones. Estos es: 18 puntos singulares, con 2 coordenadas espaciales X e Y por cada uno. De esta manera, además de generar un dataset mucho más pequeño y con información optimizada, se asegura la privacidad de los voluntarios que han ayudado en la elaboración del mismo, no guardando bajo ningún caso fotos o información personal de ningún tipo.

Las características principales del dataset quedan resumidas en los siguientes puntos:

- ✓ Han participado 3 sujetos en edad adulta:
 - 2 varones
 - 1 mujer
- ✓ El Dataset completo consta de 4857 ejemplos etiquetados.
- ✓ Todos los sujetos han repetido cada gesto al menos 300 veces:
 - 'STOP': 971 muestras
 - 'AHEAD': 973 muestras
 - 'RIGHT': 973 muestras
 - 'LEFT': 970 muestras
 - 'NONE': 970 muestras
- ✓ La información capturada se ha guardado como un array (18, 2), por cada gesto y cada sujeto de manera independiente.

Una vez capturados y guardados todos los gestos, damos paso al preprocesado de datos. Para ello, empleamos un nuevo “Jupyter Notebook” que podemos ejecutar en un ordenador de propósito general, no como todo el trabajo previo, que se había realizado sobre la propia unidad embebida. El primer paso es cargar todos los arrays previamente guardados al espacio de trabajo, y agruparlos en un único objeto por cada tipo de gesto. A continuación hacemos un nuevo recuento y buscamos posibles entradas “NaN” que puedan afectar el entrenamiento de los modelos posteriores.

```
#Analizamos sus características:
print('STOP: shape\t', stopDataset[0].shape, '\tLong:', len(stopDataset), '\tNº NaN:', np.isnan(stopDataset).sum())
print('AHEAD: shape\t', aheadDataset[0].shape, '\tLong:', len(aheadDataset), '\tNº NaN:', np.isnan(aheadDataset).sum())
print('RIGHT: shape\t', rightDataset[0].shape, '\tLong:', len(rightDataset), '\tNº NaN:', np.isnan(rightDataset).sum())
print('LEFT: shape\t', leftDataset[0].shape, '\tLong:', len(leftDataset), '\tNº NaN:', np.isnan(leftDataset).sum())
print('NONE: shape\t', noneDataset[0].shape, '\tLong:', len(noneDataset), '\tNº NaN:', np.isnan(noneDataset).sum())
```

STOP: shape	(18, 2)	Long: 971	Nº NaN: 0
AHEAD: shape	(18, 2)	Long: 973	Nº NaN: 0
RIGHT: shape	(18, 2)	Long: 973	Nº NaN: 0
LEFT: shape	(18, 2)	Long: 970	Nº NaN: 0
NONE: shape	(18, 2)	Long: 970	Nº NaN: 0

Figura 22. Ejemplo Dataset capturado. Fuente: Elaboración propia.

El siguiente paso será generar una colección de archivos en formato “.csv”, mucho más exportables y utilizable en otras herramientas que no puedan manejar arrays de Numpy. Para ello, antes debemos de generar y añadir una lista con las cabeceras que identifican cada coordenada con la parte del cuerpo a la que hacen referencia, tanto para las coordenadas “X” como las “Y”.

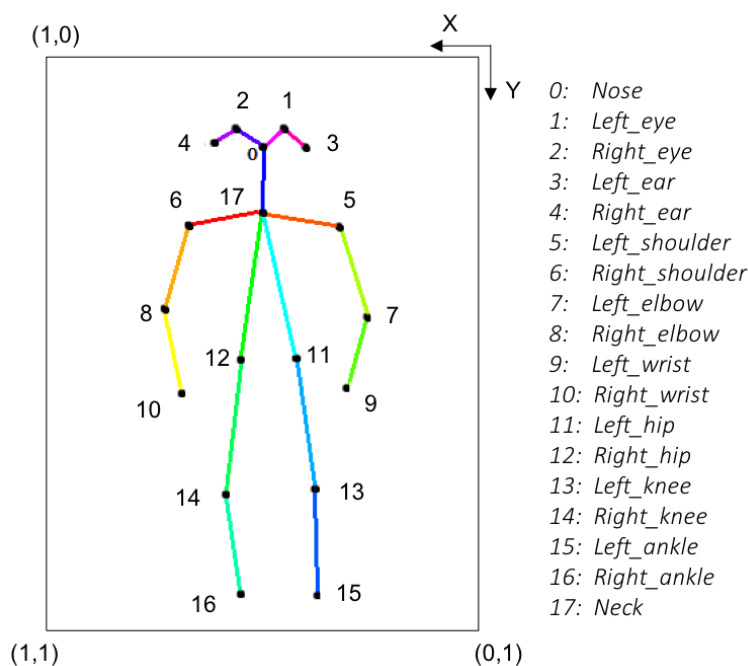


Figura 23. Relación “keypoints” y articulaciones . Fuente: Elaboración propia.

Por último generamos y guardamos los 5 archivos que componen nuestro dataset final, en formato “.csv” introduciendo una primer fila como cabecera. Este conjunto se publicará en un repositorio “GitHub” público, de tal manera que los resultados de este trabajo puedan ser reproducidos o ampliados por otros interesados.

stop.csv					
	noseX	noseY	left_eyeX	left_eyeY	right_eyeX
1	0.4751974940299988	0.2783983051776886	0.49778932332992554	0.25115981698036194	0.45600947737693787
2	0.47904425859451294	0.2716333270072937	0.500649631023407	0.24567170441150665	0.4594314992427826
3	0.4751324951648712	0.2660582959651947	0.49948740005493164	0.24731293320655823	0.45695000886917114
4	0.4771643280982971	0.2760014235973358	0.5003176927566528	0.2504390478134155	0.45732882618904114
5	0.47596102952957153	0.262143075466156	0.4991268813610077	0.24339225888252258	0.45733410120010376
6	0.47135305404663086	0.27115949988365173	0.49411246180534363	0.24752917885780334	0.4509400725364685
7	0.47522836923599243	0.2756428122520447	0.49846452474594116	0.25102323293685913	0.4564017951488495
8	0.47534820437431335	0.2737363278865814	0.4996498227119446	0.2479945719242096	0.4548669755458832
9	0.47787362337112427	0.27689510583877563	0.5013399720191956	0.24816536903381348	0.45731231570243835
10	0.482445627450943	0.2746840715408325	0.5035560727119446	0.25073346495628357	0.4634073078632355

Figura 24. Ejemplo archivo “CSV” generado . Fuente: Elaboración propia.

5.3. Preparación de los datos.

Una vez que tenemos listo nuestro dataset, llega el momento de examinar su contenido con más detalle con el objetivo de entender mejor el tipo de información que contiene y adaptarlo para su uso durante la siguiente etapa del proyecto: el entrenamiento de modelos de aprendizaje supervisado.

Como ya habíamos adelantado previamente, cada muestra está compuesta por 18 puntos de dos coordenadas cartesianas, en formato de cómo flotante de doble precisión (float64). Dichos puntos ya están normalizados, es decir, su valor es un número decimal entre 0 y 1. Las coordenadas aumentan su magnitud cuanto más a la izquierda y más próximo a la parte inferior de la imagen estén. Es muy importante resaltar que las coordenadas de los puntos singulares no detectados durante la captura (e.g. pies) se han rellenado con un valor 0.

Se plantea la siguiente estrategia para la preparación de los datos previos al entrenamiento:

1. Generamos vector equivalente por cada dataset con sus etiquetas:

Hasta ahora nuestro dataset no contiene una etiqueta por cada fila/muestra, sino que las distinguimos por el tipo de archivo (e.g. “stop.csv”- solo contiene muestras de ‘STOP’). Necesitamos generar un vector de la misma longitud que cada una de las muestras por cada conjunto, que nos ayude a identificar gestos de manera individual. Para ello utilizaremos una codificación numérica, por la que a cada una de las 5 categorías le asignamos un valor del 0 al 4.

Usaremos el siguiente código:

```
0 -> STOP
1 -> AHEAD
2 -> RIGHT
3 -> LEFT
4 -> NONE

#Añadimos array de ETIQUETAS
labels = np.zeros(len(stopDataset)) #Añadimos tantos 0 como datos de STOP
labels = np.append(labels, np.full((len(aheadDataset)), 1)) #Añadimos tantos 1 como datos de AHEAD
labels = np.append(labels, np.full((len(rightDataset)), 2)) #Añadimos tantos 2 como datos de RIGHT
labels = np.append(labels, np.full((len(leftDataset)), 3)) #Añadimos tantos 3 como datos de LEFT
labels = np.append(labels, np.full((len(noneDataset)), 4)) #Añadimos tantos 4 como datos de NONE

print(labels) #Mostramos todas las etiquetas
print("%i total examples for training." % len(labels)) #Y su suma total

[0. 0. 0. ... 4. 4. 4.]
4857 total examples for training.
```

Figura 25. Generación de etiquetas . Fuente: Elaboración propia.

2. Unificamos todos los dataset en uno solo :

Añadimos un conjunto de muestras detrás de otro, de tal manera que obtengamos un único array que contengan tantas filas como muestras tenga nuestro conjunto de datos. Es muy importante que las muestras se añadan siguiendo el mismo orden que se ha dado a las etiquetas, es decir, primero las muestras de 'STOP', luego las de 'AHEAD' etc..

```
#Juntamos todos los datos en un único dataset
dataset = np.append(stopDataset, aheadDataset, axis=0)
dataset = np.append(dataset, rightDataset, axis=0)
dataset = np.append(dataset, leftDataset, axis=0)
dataset = np.append(dataset, noneDataset, axis=0)

print("Tamaño Dataset:", len(dataset))

Tamaño Dataset: 4857
```

Figura 26. Unificación en un único Dataset . Fuente: Elaboración propia.

3. Mezclamos aleatoriamente el orden de las muestras:

Con el objetivo de obtener una muestra homogénea con una distribución aleatoria de ejemplos de cada tipo procedemos a mezclar el dataset. Es importante señalar que en ningún momento perdemos la relación entre muestras y sus etiquetas asignadas.

```
#Mezclados los datos manteniendo una correspondencia entre etiqueta y feature
from sklearn.utils import shuffle

X, y = shuffle(dataset, labels)
print(y)

[4. 0. 0. ... 0. 0. 3.]
```

Figura 27. Mezclado aleatorio de datos. Fuente: Elaboración propia.

4. Convertimos etiquetas a formato "categórico".

Para poder utilizar estos datos en la fase de entrenamiento debemos convertir las etiquetas [0,1,2,3,4] al formato categórico o también llamado "One-hot". Esto da como resultado un vector, poco denso y en su mayoría de ceros, de dimensión el número clases posibles, en el que solo el campo de la clase correspondiente tiene valor 1. Por ejemplo, en este caso la clase '2' ('AHEAD') se codificaría como [0 0 1 0 0].

5. División Entrenamiento / Test:

Para poder hacer una análisis objetivo de la capacidad de predicción del modelo con datos que nunca antes haya visto se decide dividir el conjunto de muestras en dos partes: el 80% se usarán para entrenamiento/validación. Guardamos el 20% restante de los datos para utilizarlos como test al final del proceso.

6. *Serialización de los datos:*

Hasta ahora, todas nuestras muestras tiene una dimensión (18,2), es decir, una matriz de 18 filas y dos columnas, una para la coordenada “X” y otra para la coordenada “Y”. Para poder pasar las muestras por nuestros modelos debemos “serializarlas” o “linealizarlas”, es decir, transformar a un único vector de 36 filas y una única columna, por entrada.

```
#Ajustamos tamaño del array para Red neuronal
print("Entrada:")
print(" Train:", X_train.shape, len (y_train))
print(" Test:", X_test.shape, len (y_test))

#Convertimos 18 keypoints *2 coordenadas X,y -> 36
X_train = X_train.reshape(len(X_train), 36)
X_test = X_test.reshape(len(X_test), 36)

print("\nSalida:")
print(" Train:", X_train.shape, len (y_train))
print(" Test:", X_test.shape, len (y_test))
```

```
Entrada:
Train: (3885, 18, 2) 3885
Test: (972, 18, 2) 972

Salida:
Train: (3885, 36) 3885
Test: (972, 36) 972
```

Figura 28. Serialización de los datos. Fuente: Elaboración propia.

5.4. Modelos para la clasificación automática.

Una vez que los datos han sido procesados para su explotación, pasamos a la evaluación y entrenamiento de los modelos de clasificación. Nos centraremos en aquellos modelos basados en aprendizaje supervisado, exclusivamente.

Justificamos esta decisión en que si bien un modelo de clasificación no supervisado basado en agrupamiento, como pudiera ser “K-MEANS” que ha sido ampliamente utilizado en la literatura consultada (Ribó et al., 2016), posibilita la adición de nuevos gestos más fácilmente, no creemos que esto sea de utilidad para la aplicación aquí descrita. Como queda recogido en el segundo capítulo de la presente memoria, nuestro dominio del problema recoge la existencia de un catálogo consensuado internacionalmente que describe 21 gestos, no contemplando variaciones o adiciones nuevas.

Por todo ello, consideramos que dadas las mejores prestaciones de clasificación y menor incertidumbre, un modelo supervisado sería la elección acertada, partiendo del supuesto de que disponemos de un conjunto de datos suficientemente grande, preciso y no sesgado que nos permitirá llevar acabo la fase de entrenamiento necesario del modelo.

Superada esta primera decisión, nos encontramos nuevamente ante un amplio abanico de clasificadores que poder explorar. Tras sopesar ventajas e inconvenientes de varios métodos propuestos teniendo en cuenta las características y limitaciones de nuestro conjunto de datos, y sobre todo apoyándonos en los interesantes resultados obtenidos por el grupo de investigación de la UC3M (Castillo et al., 2019) tras evaluar y comprar más de 50 modelos para la clasificación de gestos, escogemos finalmente llevar a cabo dos implementaciones: un modelo basado en un “ensamble de árboles de decisión”, y un modelo basado en un red neuronal artificial densamente conectada de poca profundidad.

5.4.1. Random Forest.

Este método de tipo “bagging”, se basa en un ensamble de un gran número de árboles de decisión y fue presentado por *Leo Breiman y Adele Cutler* en el año 2001 (Breiman, 2001). Desde entonces, ha sido ampliamente utilizado y validado en muy diversas aplicaciones, debido a su capacidad predictora, facilidad de entrenamiento e implementación.

El principal problema de los modelos basados en un único árbol de decisión es la baja capacidad predictiva, el ruido y su inestabilidad. Al combinar varios árboles de decisión y obtener la predicción final como la más votada de todas las clases predichas por cada árbol individual conseguimos un modelo con mejor capacidad de generalización y predicción, incluso con conjuntos de datos no demasiado grandes.

En este método, cada nuevo árbol se construye con una fracción aleatoria tanto de las muestras disponibles como de las variables del problema. Al aumentar el número de árboles se aumentan las posibilidades de que aquellas variables más descriptoras aparezcan varias veces en diferentes árboles. Esta característica del método es esencial para explicar su buen desempeño, ya que si bien el sesgo del modelo se ve incrementado respecto al de un único árbol, la varianza final se reduce en gran medida, dando como resultado un método versátil y potente.

Por otro lado, el hecho de que en cada sub-árbol se deje un conjunto de muestras sin utilizar (denominadas “out-of-the-bag”), permite que se entrene de manera secuencial sin necesidad de utilizar validación cruzada o “leave-one-out”, ya que el efecto obtenido es similar. Este es un excelente indicador de la capacidad de aprendizaje del modelo. Cuando el error calculado con las muestras no utilizadas se estabiliza indica que podemos detener el entrenamiento.

Como en cualquier modelo de aprendizaje supervisado, la elección correcta de hiperparámetros juega un papel fundamental. Según la literatura consultada son dos los parámetros principales a optimizar: el número de árboles y el número de variables. Si bien

otros parámetros tales como el número de muestras por cada hoja, o la profundidad máxima del árbol también pueden ayudar a obtener un buen modelo evitando un sobreajuste excesivo, en la práctica no se ha apreciado cambios significativos al variar su valor.

Para analizar el impacto de cada uno de estos hiperparámetros en la mejora de las capacidades predictivas del modelo, vamos a evaluar la raíz del error cuadrático medio o “RMSE”, ya que es una métrica fácil de calcular y que recoge el promedio de los errores de predicción para diferentes combinaciones.

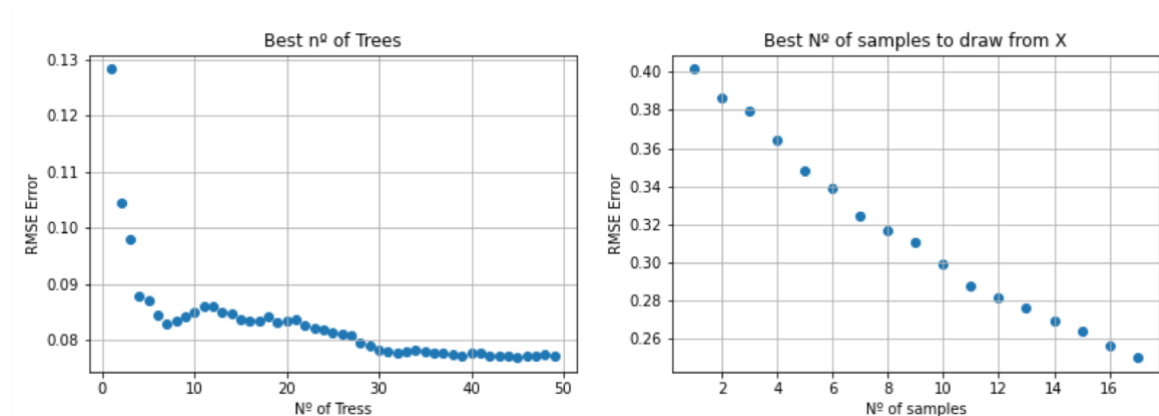


Figura 29. Hiperparámetros Random Forest. Fuente: Elaboración propia.

Para el caso del número de variables a extraer para entrenar cada estimador base se concluye que el error decrece aumentando el número de variables presentes en cada estimador. Finalmente se decide dejar el valor por defecto, que se fija igual al número máximo de variables independientes disponibles. Por otro lado, si que se observa una clara dependencia del error del conjunto de test en función del número de árboles máximo permitido. Finalmente, se considera fijar este parámetro en 45 árboles, al estabilizarse el error a partir de esta medida, tal y como se aprecia en la figura anterior.

```
# Random Forest
from sklearn.ensemble import RandomForestRegressor

rand_forest = RandomForestRegressor(random_state=0, n_estimators=45)
rand_forest.fit(X_train,y_train);
y_predicha = rand_forest.predict(X_test)

rmse_RFR = sqrt(mean_squared_error(y_test, y_predicha))
r_RFR = r2_score(y_test, y_predicha)

print("RMSE RandomForestRegressor:", rmse_RFR)
print("R^2 score RandomForestRegressor:", r_RFR)

RMSE RandomForestRegressor: 0.0768466043459443
R^2 score RandomForestRegressor: 0.9625752578761441
```

Figura 30. Ajuste del modelo mediante RF. Fuente: Elaboración propia.

Una vez analizado la combinación de hiperparámetros que mejor se ajusta a las características de nuestro conjunto de datos llega el momento de entrenar nuestro modelo. Una de las ventajas de este método es que permite construir de una forma sencilla gráficas que muestren la importancia de cada variable, como el resultado agregado para todos los árboles de la mejora obtenida en el criterio de corte por dicha variable.

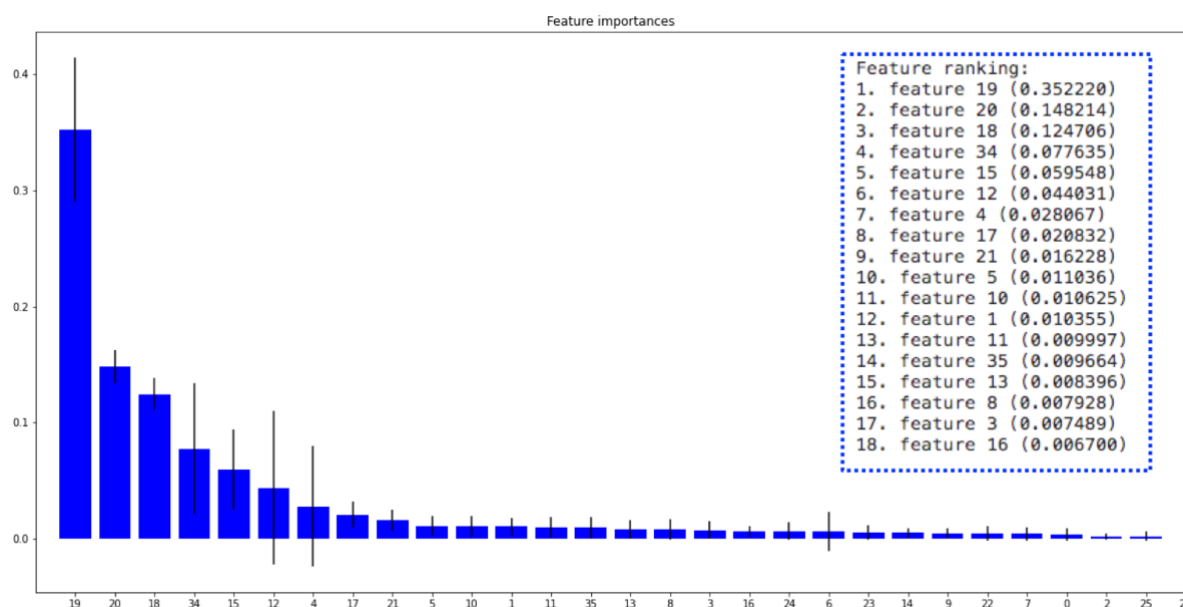


Figura 31. Importancia de cada variable en RF. Fuente: Elaboración propia.

Del anterior gráfico llama la atención la distribución exponencial, que pone de relieve el papel fundamental de tres de las características frente al resto, prácticamente inutilizadas. Haciendo una observación más detallada, se observa que estas variables (19, 20 y 18) corresponden a las coordenadas de la muñecas, lo cuál es del todo lógico ya que son las partes del cuerpo que más se desplazan en cada uno de los gestos, por lo tanto, más ayudan a su diferenciación.

Como el objetivo de la presente sección es la de comparar diferentes modelos de clasificación para nuestro conjunto de datos, es importante escoger las métricas que emplearemos para ello. Estos indicadores deben ser lo más completos posibles, y reflejar la capacidad del modelo tanto para predecir con precisión como para distinguir entre errores de tipo falso positivo y falso negativo.

En el caso de los algoritmos de clasificación, no hemos encontrado mejor herramienta que la propia “Matriz de Confusión” para llevar acabo este análisis, ya que además de permitirnos mostrar en un tabla la relación entre “predicción” y “realidad”, nos permite calcular con facilidad otras muchas métricas. De todas estas, basaremos nuestro análisis en dos de ellas:

- ✓ Accuracy o ratio de éxito: Métrica simple de comprender aunque a veces incompleta. Nos informa sobre la proporción entre las predicciones correctas que ha hecho el modelo sobre el total de predicciones.
- ✓ F-Score: Esta métrica combina “Precisión” y “Recall” utilizando la media armónica. Es una forma de evaluar las soluciones de compromiso adoptadas por el modelo durante la clasificación. Es ampliamente utilizado en la literatura (Castillo et al., 2019) debido a que simplifica el rendimiento del modelo completo a una única métrica.

Finalmente, en la siguiente imagen se muestran los resultados del modelo basado en Random Forest. Realmente, las marcas obtenidas son excelentes, con una ratio de éxito del 98%. Analizando los errores cometidos, destacan varios fallos de clasificación entre los gestos de parada ('STOP') y los de avance ('AVANCE'). Esto es lógico si tenemos en cuenta que ambos gestos se basan en el movimiento de ambos brazos cerca de la cabeza, y que no estamos codificando más información que la espacial.

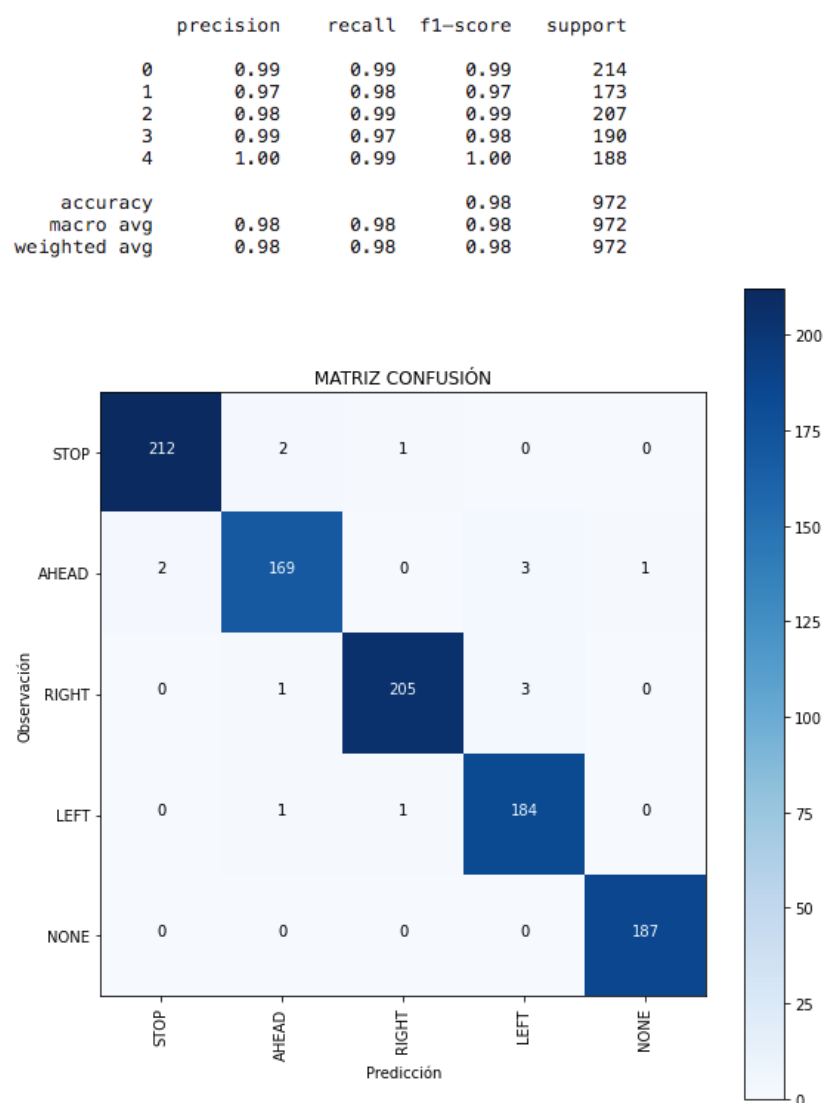


Figura 32. Matriz de confusión y métricas RF. Fuente: Elaboración propia.

5.4.2. Red Neuronal Artificial: Perceptrón Multicapa.

A continuación se abordará el diseño, elección de hiperparámetros, entrenamiento y validación de un modelo basado en una red neuronal artificial multicapa y densamente conectada.

Las arquitecturas basadas en redes neuronales artificiales nos permiten, gracias a sus excelentes propiedades como aproximadores universales, modelar una relación entre el conjunto de variables de entrada y las señales de salida, a pesar del ruido o complejidad del patrón presente. Su gran versatilidad, acompañado de la mayor disponibilidad de datos para su entrenamiento y mejora en las capacidades computacionales actuales, han impulsado su uso en multitud de tareas de aprendizaje automático, sobresaliendo en tareas de clasificación de todo tipo.

Cabe reforzar la idea de que este es el segundo modelo basado en una red neuronal artificial profunda que se emplea en el presente proyecto. El primer modelo ejecutado, al que nos hemos venido refiriendo como “Convolutional Pose Machine”, se encarga de identificar las coordenadas de las extremidades del cuerpo y específicamente en nuestro caso concreto se basa en una red convolucional pre entrenada y altamente optimizada, siguiendo una arquitectura de tipo “Residual” de 18 capas (NVIDIA® TensorRT Pose Estimation).

El segundo modelo que vamos a presentar y describir a continuación, sigue una arquitectura completamente diferente y se ejecutará tomando como entradas la salida del anterior modelo, es decir, las coordenadas cartesianas que identifican dichos puntos singulares del cuerpo humano. Su topología será la de una red neuronal pre alimentada y densamente conectada, es decir, en la que todas las neuronas de una capa se conectan con todas las neuronas de la siguiente. La información seguirá un recorrido secuencial desde la capa de entrada hasta la de salida. A diferencia del anterior modelo que había sido preentrenado sobre un dataset de gran tamaño, esta red será entrenada por completo usando nuestro conjunto de datos generado previamente. La misión de este modelo será, dado un vector de entrada con las coordenadas del cuerpo, calcular la probabilidad de pertenencia a una de las 5 clases que componen nuestro problema de clasificación.

Existen numerosas adaptaciones de estas redes neuronales artificiales en función de la tarea y tipo de datos a procesar. La capacidad de una red para aprender y predecir correctamente está directamente relacionada con el número de capas presentes, el reparto de neuronas por cada capa, su conexionado interno, la función de activación o el optimizador empleado durante la etapa de entrenamiento, entre otros aspectos a tener en cuenta. Si bien por norma general se puede decir que redes más grandes, es decir aquellas con mayor profundidad y

número de neuronas por capa, pueden modelar problemas más complejos permitiendo identificar patrones más sutiles y fronteras de decisión más complejas, también es cierto que un exceso en este sentido puede conducir a modelos poco optimizados, difíciles de entrenar y con tendencia al sobreajuste, lo que impide que sean capaces de extrapolar adecuadamente ante el presencia de nuevos datos no vistos con anterioridad.

Volviendo a la descripción de la arquitectura de nuestra red neuronal, podemos identificar 3 tipos de capas principales:

- ✓ **Entrada:** Cada una de las neuronas de esta capa debe ser capaz de procesar una de las variables del conjunto de datos, por lo que en nuestro caso específico, la capa de entrada deberá contener 36 neuronas.
- ✓ **Salida:** El número de neuronas de la capa de salida viene determinado por el número de resultados posibles del clasificador. En nuestro caso está formada por una capa de 5 neuronas con función de activación “softmax”, lo que nos proporciona como resultado del modelo un vector normalizado con las probabilidades de pertenencia a cada una de las clases mutuamente excluyentes.
- ✓ **Ocultas:** Las capas ocultas son las capas intermedias de nuestra red por las que la información va fluyendo desde las capas de entrada hasta las de salida. El número de neuronas de cada capa es un hiperparámetros del modelo cuya elección se abordará más adelante.

Cada neurona de la capa tiene una función de activación que define su salida, que a su vez se propagará hacia las capas posteriores. Esta función permite introducir la no linealidad en las capacidades de representación de la red, y desempeña un papel destacado. No es el objetivo de este trabajo hacer una evaluación detallada de los diferentes hiperparámetros posibles y su potencial optimización, sino encontrar una primer solución de compromiso que permita evaluar el sistema en su conjunto. Por este motivo, se ha escogido para todas las neuronas de la red una función de activación de tipo “Rectified Linear Unit” (RELU), al ser esta una función de activación ampliamente usada y que ha demostrado dar excelentes resultados en diferentes situaciones.

El optimizador es otro de los argumentos importantes a definir para conseguir una respuesta eficiente. Esto es lógico si atendemos a que el proceso de aprendizaje es de base un problema de optimización global en el que se deben calcular los pesos y sesgos que minimicen la función de pérdida, o “loss”, que cuantifica lo lejos que está la salida de la red de la respuesta esperada. Para la función de pérdida, dada la naturaleza de nuestro problema y la manera en la que hemos preparado las etiquetas, hemos escogido “categorical cross entropy”. En el caso del optimizador, nuevamente nos hemos decantado de partida por un optimizador de tipo “ADAM” (Adaptative Momnet Estimation) que basado en el mismo principio que el cálculo del

descenso por el gradiente, introduce los conceptos avanzados de “momento” y “RMSProp”, lo cuál se traduce en un método más robusto y de convergencia más rápida.

A nivel de algoritmo del propio modelo de entrenamiento, y no tanto ya de estructura o topología, debemos mencionar dos hiperparámetros importantes a tener en cuenta: el “número de épocas” y el “tamaño de los lotes de entrenamiento por cada iteración”. El primero de ellos, nos ayuda a definir el número de veces en las que todos los datos de nuestro conjunto de entrenamiento deben pasar por la red. Este valor es clave para evitar un sobreajuste de los mismos. El otro, nos permite definir el tamaño de los “mini lotes” de datos de nuestro conjunto que usaremos en cada iteración completa.

```
## MODELO_1
model_1= tf.keras.models.Sequential([
    #DNN 1
    tf.keras.layers.Dense(128, activation='relu', input_shape=(36,)),
    tf.keras.layers.Dropout(0.5),
    #DNN 2
    tf.keras.layers.Dense(128, activation='relu'),
    tf.keras.layers.Dropout(0.5),
    #DNN 3
    tf.keras.layers.Dense(y.shape[1], activation='softmax', name = "LastLayer")
])

#Summary
model_1.summary()

#Compile
model_1.compile(optimizer='Adam',
                loss='categorical_crossentropy',
                metrics=['acc'])

#Entreno
batch_size = 32
epochs=300

history1=model_1.fit(X_train, y_train,
                    epochs=epochs,
                    batch_size=batch_size,
                    validation_split=0.2,
                    verbose=0)
```

Model: "sequential"

Layer (type)	Output Shape	Param #
dense (Dense)	(None, 128)	4736
dropout (Dropout)	(None, 128)	0
dense_1 (Dense)	(None, 128)	16512
dropout_1 (Dropout)	(None, 128)	0
LastLayer (Dense)	(None, 5)	645
Total params: 21,893		
Trainable params: 21,893		
Non-trainable params: 0		

Figura 33. Resumen modelo basado en “ANN”. Fuente: Elaboración propia.

Como se ve en la imagen anterior, nuestro mejor modelo está compuesto por dos capas ocultas, densamente conectadas. Entre capas de neuronas se han intercalado sendos filtros de regularización basados en “Dropout” que permite desactivar el 50% de las neuronas de manera aleatoria entre iteraciones del ciclo de aprendizaje exclusivamente. Esta técnica ha demostrado unos excelentes resultados en aplicaciones similares, al prevenir que el modelo termine memorizando completamente los datos de entrenamiento, perdiendo su capacidad de generalización.

En resumen, en la fase de entrenamiento de nuestro modelo deberemos calcular un valor para los 21893 parámetros (pesos y sesgos) que componen nuestra red.

Finalmente, antes de comenzar con el entrenamiento se configura el método para reservar un 20% de los datos de entrenamiento para validación. Esto permite evaluar el rendimiento del modelo sobre ese conjunto de datos que no ha sido utilizado por cada iteración.

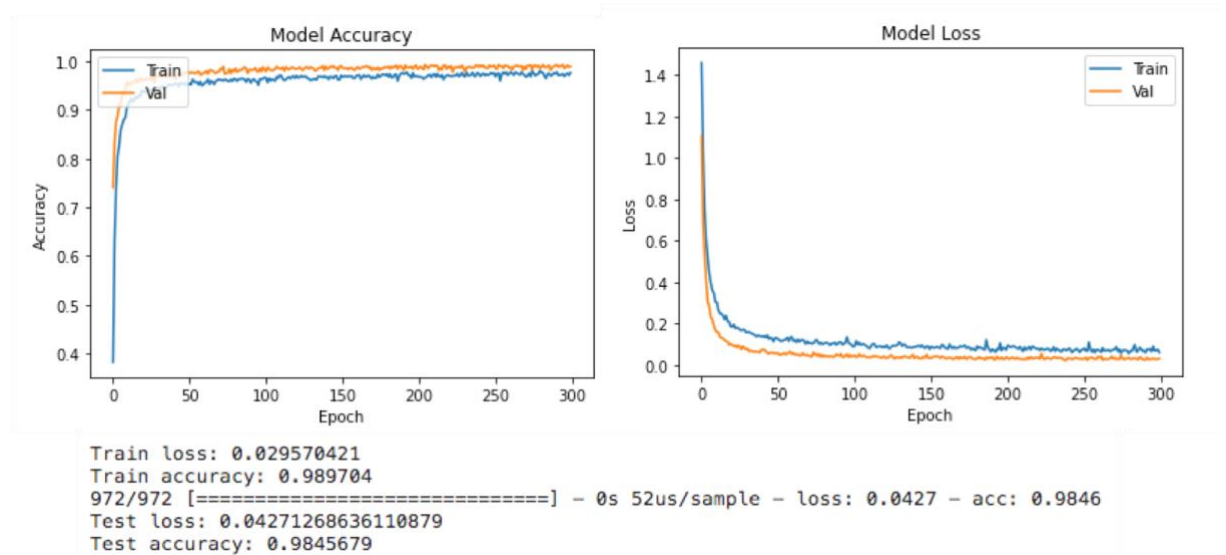


Figura 34. Resultados del modelo tras entrenamiento. Fuente: Elaboración propia.

Tras algunas iteraciones se observa que no tiene utilidad ampliar el número de iteraciones en la fase de entrenamiento a más de 300 épocas. A partir de este momento, podemos considerar que la función de error se ha estabilizado entorno a su valor mínimo. Como puede apreciarse en la figura anterior, los resultados preliminares son prometedores. Hemos alcanzado una “ratio de éxito” del 98% sobre el conjunto de prueba, que no había visto la red con anterioridad. Además, observamos como la gráfica con los datos de validación se mantiene estable y con una respuesta similar a la generada con los datos de entrenamiento, lo que indica que no se está produciendo un sobreajuste de la red.

Llegados a este punto, nos planteamos si un ajuste de los hiperparámetros presentados anteriormente tendría alguna consecuencia positiva, que ayudase a incluso mejorar el rendimiento alcanzado hasta ahora. Tras varias pruebas, no se aprecia una mejora considerable a pesar de iterar sobre varios modelos. A modo de ejemplo presentamos los resultados comparativos de nuestro modelo original (Modelo-1), con un modelo similar en el que el número de neuronas de todas las capas ha sido reducido a 32 (Modelo-2), y otro en el que se ha aumentado este valor hasta las 256 neuronas por capa (Modelo-3).

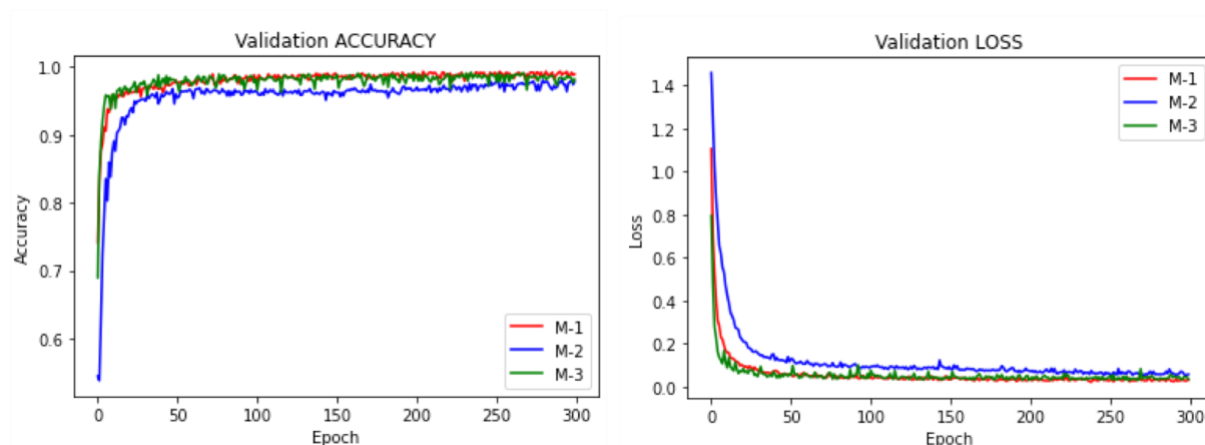


Figura 35. Resultados comparación 3 modelos. Fuente: Elaboración propia.

Como se aprecia en la gráfica del Modelo-2, la reducción del número de parámetros entrenables (2405 “train-params”) tiene el efecto de un ligero empeoramiento del rendimiento. Por otro lado, el Modelo-3 (142341 “train-params”), no supone un considerable aumento de las prestaciones. En ninguno de los casos, el coste computacional o tiempo de ejecución destaca, siendo las diferencias entre ellos pequeñas, y atribuibles al número de neuronas presentes en cada red. Por todo ello, optamos por la primera opción presentada, como nuestro mejor modelo basado en una red neuronal artificial.

A continuación, y con el fin de poder acometer una comparación objetiva entre este modelo, y el anteriormente presentado basado en “Random Forest”, calculamos nuevamente la “Matriz de Confusión” y con ella calculamos, los indicadores principales de rendimiento.

Analizando la tabla observamos que el modelo tiene una rendimiento sobresaliente. El ratio de éxito, tal y como ya habíamos avanzado es del 98%. De nuevo destaca que el mayor número de errores se debe a haber clasificado varios gestos etiquetados como ‘AHEAD’ en la categoría de ‘STOP’. Tal y como ya habíamos analizado en el caso del modelo basado en “Random Forest”, esto podría deberse a que en ambos gestos las manos están próximas a la cabeza y por encima del cuello, lo que puede producir cierta confusión.

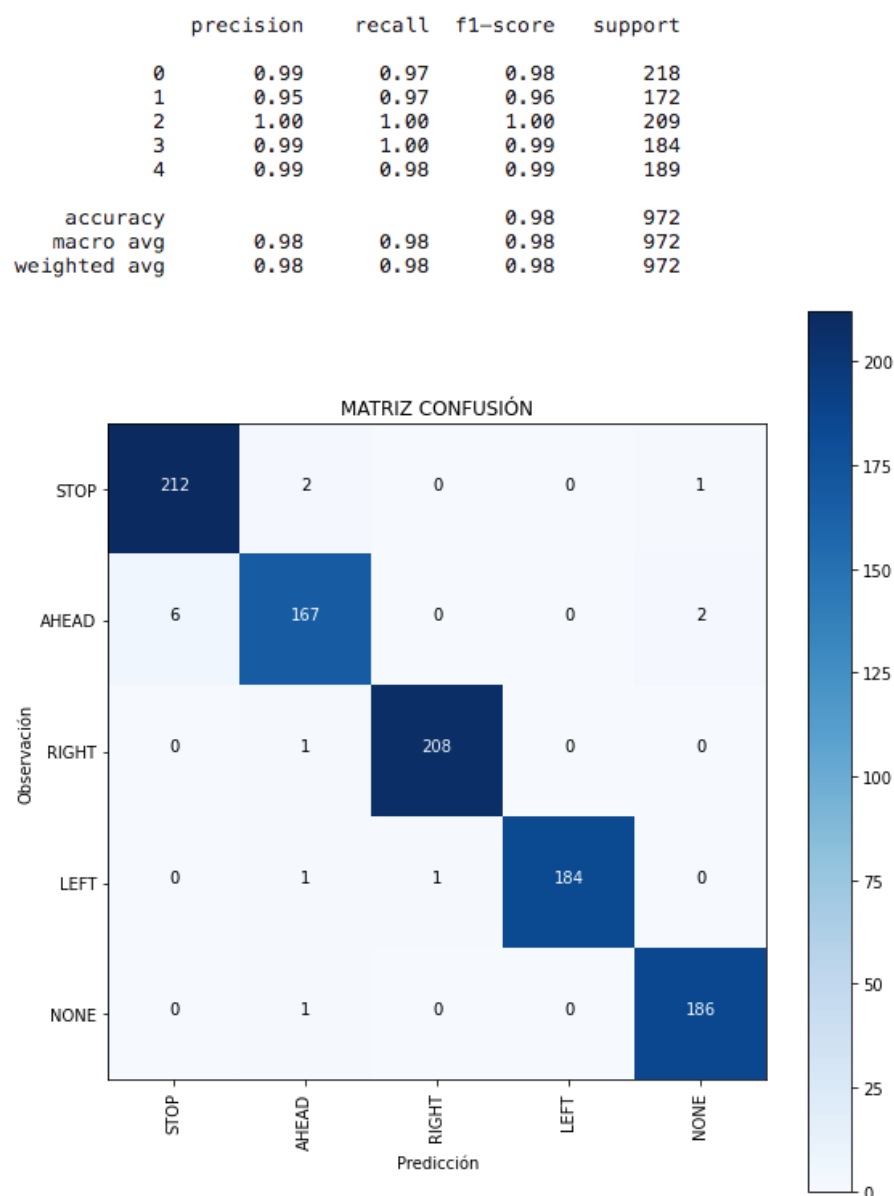


Figura 36. Matriz de confusión y métricas ANN. Fuente: Elaboración propia.

5.5. Evaluación de la solución en entorno real.

Una vez implementados y entrenados nuestros modelos, llega el momento de hacer una comparación objetiva entre ambos a fin de identificar cuál de ellos presenta las mejores características para la clasificación en tiempo real de los gestos llevados a cabo por un señalero.

Analizando el “Ratio de éxito” y la “Matriz de confusión” sobre los datos que habíamos decidido reservar de nuestro conjunto de entrenamiento, no se aprecia una clara ventaja de un modelo sobre otro, presentando ambos unas prestaciones similares sobre el mismo conjunto de datos.

Para llevar acabo la evaluación en tiempo real, debemos mover nuestros modelos ya entrenados al dispositivo embebido que ejecutará el procesamiento de imagen y clasificación en tiempo real.

En el caso del modelo basado en “Random Forest”, utilizamos la herramienta recomendada por la propia librería “*Scikit-Learn*”, denominada “*Pickle*”. Esta herramienta permite guardar en disco un archivo con extensión “.sav” que contiene nuestro modelo como un objeto de Python. Utilizando un procedimiento análogo es posible volver carga dicho modelo desde el archivo local. En el caso de la red neuronal, es incluso más simple ya que desde el propio “framework *Tensorflow 2.0*” podemos exportar el modelo con los pesos de la red en un archivo en formato “.h5”.

De nuevo, volvemos a trabajar sobre el “Jupyter Notebook” sobre el que habíamos implementado nuestra interfaz gráfica de usuario. Una vez cargados ambos modelos sobre la carpeta correspondiente del directorio del programa, podemos comenzar la evaluación. Para ello haremos uso de las herramientas contenidas en “área de predicción” que había sido explicada en la primera sección de este capítulo. Con el menú desplegable podemos seleccionar cual de los dos modelos queremos ejecutar, pudiendo pasar de uno a otro en tiempo real. Evaluaremos la respuesta del modelo observando tanto la clase predicha, que se mostrará en el campo ‘OUT’, como con las barras verticales que informarán de manera gráfica sobre la probabilidad de pertenencia a cada una de las cinco categorías.

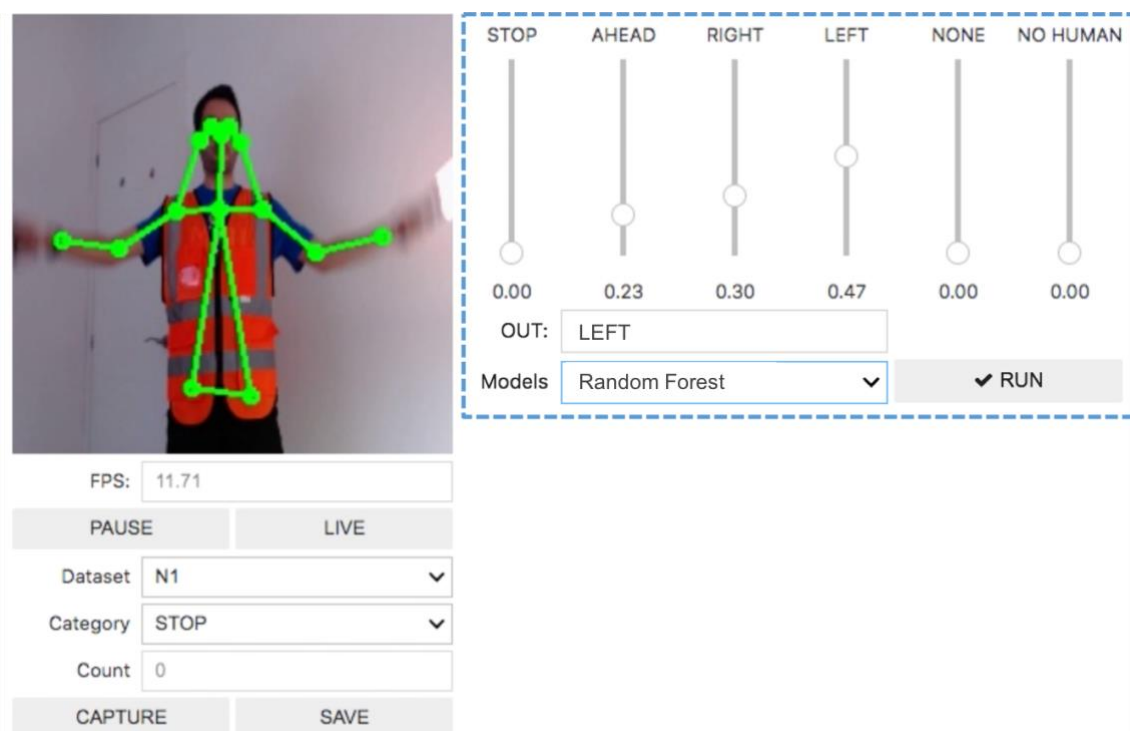


Figura 37. Prueba de ejecución en tiempo real. Fuente: Elaboración propia.

Para la evaluación se utilizará un cuarto voluntario, varón y de complejión similar al resto de voluntarios pero que no había participado en la generación del conjunto de datos de entrenamiento. Nuevamente se le equipa con el chaleco de alta visibilidad y las baritas de señalización. Se le pide que interactúe con el sistema, desplazándose por su campo de visión y ejecutando los diferentes gestos tal y como un operario lo haría en la vida real.

Comenzando por el modelo “Random Forest”, observamos un buen comportamiento. El modelo es ligero, y la herramienta es capaz de ejecutarlo a una frecuencia de 12Hz, dando una agilidad de respuesta muy adecuada para la tarea. Aunque la clase más probable es correcta en la mayoría de los casos, se observa que el modelo predice con un alto nivel de incertidumbre y que las otras clases también tiene probabilidades bastante elevadas, en algunos casos. En general, la respuesta global del sistema es correcta y cumple su cometido, pero se detecta un salida con menos ratio de éxito y mucho más ruido que los resultados obtenidos sobre el conjunto de datos usados para entrenamiento, validación y test. En los anexos se puede encontrar un enlace al video con los resultados de este modelo.

A continuación, cargamos el modelo de la red neuronal con los pesos ya precargados en el formato “.h5”. Aunque el modelo parece que predice correctamente, es demasiado pesado y el sistema no es capaz de ejecutarlo a más de 4 veces por segundo. El retardo que introduce, hace que sea demasiado complicado operar con el sistema de una manera ágil y dificulta su evaluación. Antes de descartar completamente su uso, y reflexionando sobre la buena tasa de ejecución del modelo convolucional que estamos ejecutando como primer paso sobre el mismo hardware embebido, y que fácilmente es capaz de inferir un nuevo vector de coordenadas cada 12Hz, decidimos investigar más sobre qué herramientas a nuestra disposición existen que permitan llevar a cabo una optimización avanzada del modelo de nuestra red neuronal.

Al disponer nuestra unidad en embebido de una GPU NVIDIA®-Maxwell de 128 núcleos, encontramos que el fabricante pone a nuestra disposición una API denominada “Tensor-RT”. Esta herramienta desarrollada en C++ permite justamente optimizar modelos de redes neuronales entrenadas en “TensorFlow” para una mejor ejecución durante la etapa de inferencia utilizando las GPUs de la marca. Para ello es necesario llevar a cabo un paso intermedio en el que procesamos el modelo generado “.h5” para obtener un nuevo archivo optimizado con extensión “.pb”



Figura 38. Paso adicional para optimizar modelo para GPU. Fuente: NVIDIA®.

En la documentación de la herramienta se explica que “*Tensor-RT*” lleva a cabo varias transformaciones y optimizaciones sobre el grafo de la red neuronal, eliminando todas las capas no utilizadas y fusionándolas donde sea posible. La fusión horizontal mejora el rendimiento al combinar capas que acepten como entrada el mismo tipo de tensor fuente y que apliquen las mismas operaciones con parámetros similares.

Al evaluar el nuevo modelo optimizado sobre nuestra aplicación, descubrimos gratamente que la herramienta de optimización ha hecho su trabajo, y el sistema es capaz de mover el modelo optimizado a una frecuencia de 12Hz, permitiendo una interacción ágil y una clasificación rápida por parte de la herramienta. Comprobamos que a diferencia del modelo basado en “Random Forest”, nuestra red neuronal artificial tiene una ratio de acierto mucho mayor, y una menor incertidumbre en sus predicciones, estando la gran mayoría de veces la clase seleccionada con una probabilidad superior al 80%, convirtiéndose en el modelo que mejor resultados ha proporcionado en la evaluación en entorno de real. Igualmente, en los anexos se puede encontrar un enlace al video con los resultados de la ejecución en tiempo real de este modelo.

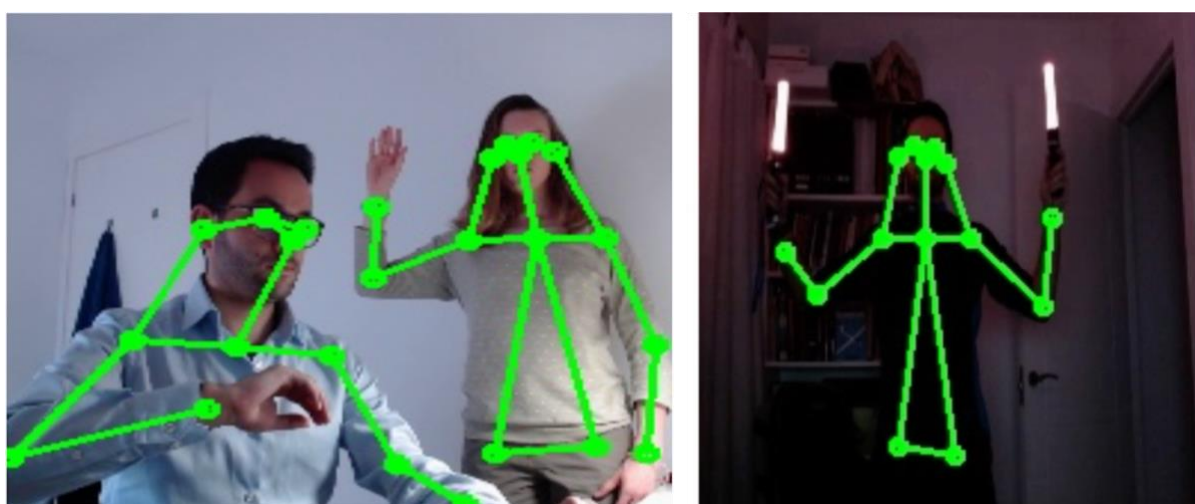


Figura 39. Pruebas con varios individuos y baja iluminación. Fuente: Elaboración Propia.

Finalmente se prueba la herramienta de manera extensiva, llevando al límite el sistema, tanto en la ejecución de los gestos como en las propias condiciones de iluminación en las que se realiza la captura. Sorprende muy favorablemente, dado el pequeño conjunto de datos con los que se ha entrenado la red, la elevada tasa de acierto del sistema y la casi nula dependencia con el entorno. Esto último es debido principalmente al uso de esa primera red convolucional pre entrenada y que lleva a cabo la primera segmentación y extracción de los puntos singulares. El sistema es capaz incluso de detectar e inferir de manera correcta e independiente a varias personas en el campo de visión. En este caso, la predicción del gesto la realiza sobre el individuo identificado más próximo al eje de coordenadas.

6. Benchmarking: Arquitectura basada en CNN.

Una vez analizados los buenos resultados obtenidos por la herramienta desarrollada en el presente proyecto, no queremos dejar de poner en valor las ventajas que la arquitectura escogida supone respecto a otras opciones posibles, en cuanto a su robustez, no dependencia del entorno y escalabilidad. Por otro lado, identificar igualmente posibles puntos débiles y desventajas de ésta frente a otras opciones, con el objetivo de tender hacia una solución final que integrase y explotase las fortalezas de todas ellas.

En especial, cuando se abordó el primer análisis del problema de la identificación en tiempo real de las señales de rampa en el entorno del aeropuerto, una de las preguntas que se plantearon fue si una topología exclusivamente basada en redes convolucionales profundas, a la que se le pasase directamente como parámetro de entrada la matriz de píxeles que conforman la imagen capturada por la cámara, podría ser suficientemente buena para llevar a cabo una tarea de clasificación del gesto en una de las 5 categorías propuestas.

6.1. Topología de la red.

Desde un primer momento, dado el reducido conjunto de datos de entrenamiento disponible y potencial sesgo de este, se tuvo en cuenta que cualquier solución en este sentido debería pasar por la utilización de una arquitectura pre entrada. En la actualidad existen numerosos modelos disponibles para las tareas de clasificación de imágenes. Estos modelos, entrenados con grandes colecciones de imágenes han sido puestos a prueba en numerosas competiciones de clasificación dando excelentes resultados, como es el caso de la mencionada *“Imagenet Large Scale Visual Recognition Challenge”* (ILSVRC) en el que se utiliza un dataset con 1000 categorías de imágenes diferentes.

Entre las diferentes arquitecturas posibles, nuevamente en este trabajo nos decantamos por una red de tipo “Residual”. Como ya habíamos presentado anteriormente este tipo de redes se basan en la incorporación de una serie de atajos entre capas que permiten conectar neuronas de capas no consecutivas. La salida de la nueva capa será una combinación tanto de la capa inmediatamente anterior como de la información que le llega mediante este otro canal de las capas previas. Los autores demuestran (He et al., 2016) que esta topología de red hace que el modelo sea mucho más óptimo, encontrando una solución de compromiso entre velocidad de aprendizaje y ratio de éxito. Hay varias arquitecturas basadas en la topología ResNet, que se diferencian en el número de capas ocultas presentes en la red, desde 18 a 512 capas.

Partiendo de los requisitos del proyecto, que determinan que la solución debería poder ejecutarse en tiempo real , en un sistema empotrado y a una frecuencia tal que permitiese una respuesta ágil, nos decantamos por el uso de la red de menor profundidad: ResNet-18.

Por medio de la técnica conocida como “Aprendizaje por Transferencia” podemos partir de una base convolucional ya entrenada contra un vasto conjunto de imágenes, como es el caso de “*Imagenet*”, pero modificar la última capa densamente conectada de la red para adaptarla a nuestro problema específico de clasificación de 5 gestos, en lugar de las 1000 categorías del modelo original. Esto nos permite reaprovechar todos los pesos y sesgos calculados de la red de las 17 capas previas, que son las encargadas de la extracción de características de bajo nivel de la imagen.

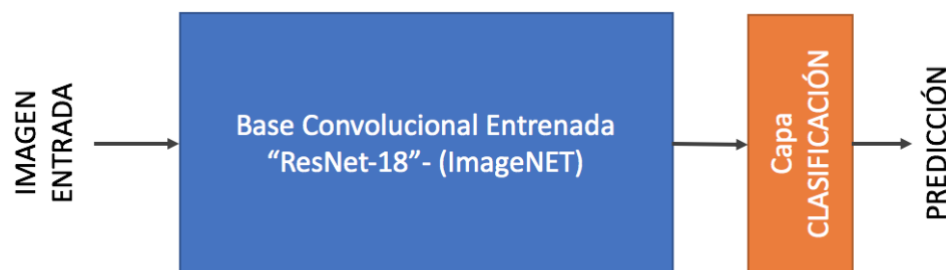


Figura 40. Idea conceptual “Transfer Learning”. Fuente: Elaboración Propia.

La última capa por lo tanto será una capa densamente conectada, que unirá cada una de las 512 neuronas con una de las 6 clases propuestas en este problema con una función de activación “softmax”. Esta última capa si debe ser entrenada para que el modelo relacione las características extraídas en las capas anteriores con una de las categorías posibles.

6.2. “Data Augmentation Techniques” .

De nuevo nos enfrentamos al problema de no contar con un conjunto de entrenamiento disponible que reúna las características que necesitamos para la clasificación de las 5 categorías de gesto llevadas a cabo por el personal de tierra del aeropuerto. Adicionalmente, se decide añadir una nueva categoría extra designada como ‘No-Human’ para clasificar los fotogramas en los que no se detecta una persona en el campo de visión. Cabe la pena mencionar que este dato lo extraíamos de la segmentación del primer modelo “CPM” en la arquitectura original, y por eso no aparecía como un salida del clasificador del segundo

modelo. Sin embargo, en este caso, al disponer exclusivamente de una única red debemos tenerla en cuenta, a fin de obtener resultados comparables entre ambas implementaciones.

Nuevamente se decide elaborar un dataset propio, que servirá tanto de entrenamiento como de validación. Siguiendo un procedimiento similar al descrito al capítulo anterior se capturaron 450 imágenes de las seis categorías.



Figura 41. Ejemplos del Dataset generado. Fuente: Elaboración Propia.

Al ser este ejercicio una “experiencia piloto” para analizar si un sistema basado en esta arquitectura tendría potencial, se decidió no invertir demasiado tiempo en generar un gran conjunto. Por esta razón tampoco se contó con voluntarios, como si se había hecho en el capítulo anterior, por las implicaciones derivadas de la protección de datos al tener que guardar las imágenes sin ejecutar alguna técnica de anonimizado que permitiese eliminar rasgos identificables.

Para poder disponer de un mayor número de muestras para el entrenamiento del modelo se implementaron herramientas de “*Data Augmentation*”. Estas técnicas se basan en la aplicación de una serie de transformaciones sobre los mini lotes de imágenes que se cogen del dataset en cada iteración de la fase de entrenamiento. Por medio de unos hiperparámetros se puede establecer el alcance de estos cambios tales como pequeñas rotaciones o cambios de color.

Es muy importante que dichas acciones no den como resultado un dato no coherente con el dominio del problema. Por ejemplo, en nuestro caso particular de reconocimiento de gestos, si permitimos llevar a cabo simetrías verticales u horizontales, podemos encontrarnos con muestras en las que la cabeza del operador esté hacia abajo, o que la etiqueta de derecha o izquierda ya no concuerde con el gesto real, lo que indudablemente produciría una respuesta equivocada del modelo.

```

import torchvision.transforms as transforms
from dataset import ImageClassificationDataset

TASK = 'air_signals'

CATEGORIES = ['STOP', 'AHEAD', 'RIGHT', 'LEFT', 'NONE', 'NO_HUMAN']
DATASETS = ['N1', 'N2', 'N3', 'N4']

TRANSFORMS = transforms.Compose([
    transforms.ColorJitter(0.2, 0.2, 0.2, 0.2),
    transforms.Resize((224, 224)),
    transforms.ToTensor(),
    transforms.Normalize([0.485, 0.456, 0.406], [0.229, 0.224, 0.225])
])

datasets = {}
for name in DATASETS:
    datasets[name] = ImageClassificationDataset(TASK + '_' + name, CATEGORIES, TRANSFORMS)

print("{} task with {} categories defined".format(TASK, CATEGORIES))

air_signals task with ['STOP', 'AHEAD', 'RIGHT', 'LEFT', 'NONE', 'NO_HUMAN'] categories defined

```

Figura 42. Ejemplos de “Data Augmentation Techniques”. Fuente: Elaboración Propia.

6.3. Entrenamiento de la red .

A diferencia de los anteriores modelos en los que se ha utilizado el “*framework Tensorflow 2.0*”, en esta experiencia hemos empleado “*PyTorch*” (v1.5.1), así como el paquete de librerías “*TorchVision*” (v0.6.1) que contiene una serie de herramientas y métodos de utilidad para el desarrollo de modelos de redes artificiales aplicados a problemas de visión artificial.

Justificamos este cambio de criterio en los siguientes puntos:

- ✓ “*PyTorch*” dispone de un modelo pre entrenado ResNet-18 sobre “*ImageNet*” ya disponible. “*Tensorflow*”, incorpora el modelo ResNet-50, que como hemos visto cuenta como muchas más capas. Si bien es posible implementar una red de este tipo de manera manual, nos decantamos por la primera opción por agilidad.
- ✓ Existe mucha documentación y ejemplos prácticos sobre “*Machine Learning*” aplicado a la familia de computadoras embebidas de NVIDIA®, lo que para las pruebas preliminares llevadas a cabo en las fases iniciales del proyecto, suponía una ventaja clara al permitir implementar y probar con rapidez varias alternativas.
- ✓ Al ser una prueba piloto, buscábamos invertir el menor tiempo posible, y partimos de un ejemplo de clasificación ya optimizado para su uso sobre una GPU NVIDIA® que está disponible en el manual de introducción del fabricante, que adaptamos para nuestro propio propósito (NVIDIA Getting Started With Jetson Nano Developer Kit).
- ✓ Por último, se quería ganar experiencia en este otro framework, ya que durante la ejecución del máster solo se había trabajado sobre “*Tensorflow 2.0*”

En el proceso de entrenamiento, es muy importante asegurar que los pesos de la base convolucional no se modifiquen, ya que esto tendría un efecto del todo contraproducente. Para ello podemos indicar al modelo que “bloquee” dichas capas, reduciendo el número de parámetros entrenables de la red, y solo centrarnos en modificar los parámetros de la última capa densa.

```
import torch
import torchvision

device = torch.device('cuda')

# RESNET 18
model = torchvision.models.resnet18(pretrained=True)
model.fc = torch.nn.Linear(512, len(dataset.categories))

model = model.to(device)
```

```
=====
Total params: 11,179,077
Trainable params: 11,179,077
Non-trainable params: 0
-----
Input size (MB): 0.57
Forward/backward pass size (MB): 62.79
Params size (MB): 42.64
Estimated Total Size (MB): 106.00
-----
```

Figura 43. Configuración del modelo basado en ResNet-18. Fuente: *Elaboración Propia*.

Como optimizador volvemos a optar por el algoritmo “Adam”, atendiendo a las mismas razones cubiertas en capítulos anteriores en cuanto a su rapidez de convergencia y eficiencia computacional. Igualmente, la función de pérdida se define como “*categorical cross entropy*”, ya que el problema no ha cambiado y nos enfrentamos de nuevo a un problema de clasificación de categorías auto excluyentes. A pesar de no rentrenar las capas de la base convolucional, debemos calcular un gran número de parámetros de la última capa densamente conectada. Como este entrenamiento se llevó a cabo en la propia GPU NVIDIA® del sistema embebido se decidió escoger un tamaño de “batch-size” pequeño, de no más de 8 muestras, para no sobrepasar la memoria disponible.

Al disponer de tan pocas muestras se tuvo que iterar varias veces sobre la combinación más correcta de valores, descubriendo que con pocas épocas de entrenamiento se producía un grave problema de sobreajuste. Con 10 épocas de entrenamiento se alcanza un valor de la función de pérdida de 0.012 y un ratio de éxito sobre el propio conjunto de datos, de más del 96%.

6.4. Evaluación de la predicción en tiempo real .

Para la etapa de inferencia, hacemos uso nuevamente de nuestra interfaz de usuario explicada en el capítulo anterior, debidamente adaptada para la evaluación del nuevo modelo. Nuevamente, conectamos la salida de la capa “softmax” a los 6 indicadores verticales, para poder tener una valoración rápida de cuál es la incertidumbre del modelo al predecir una categoría. La clase más probable, se visualiza en un campo de texto justo debajo de los indicadores verticales.



Figura 44. Validación en tiempo real del modelo. Fuente: Elaboración Propia.

La evaluación del modelo durante la etapa de inferencia en tiempo real arroja buenos resultados. El sistema responde con agilidad y el modelo es capaz de predecir con poca latencia cada una de las 6 categorías con una elevada tasa de acierto, a pesar de haber sido entrando con tan pocas imágenes, lo cuál nos sorprendió muy gratamente. Justificamos este resultado, en parte, por la simplicidad de las imágenes capturadas para el entrenamiento en el que el sujeto vestía con ropa oscura y se situaba sobre un fondo blanco, permitiendo una fácil segmentación. En los anexos se puede encontrar un enlace al video con los resultados de ejecución en tiempo real de este modelo también.

El problema aparece cuando el contexto de la imagen cambia sustancialmente, ya bien sea en la persona que ejecuta los gestos o por el propio fondo. El modelo generado no tiene capacidad de generalizar al haber sido entrenado con pocos ejemplos de cada caso, e imágenes simples que no recogen la complejidad real con un fondo muy cambiante.

Por lo que, aunque funcional en un contexto muy determinado, fuera de él este modelo no sería una solución real a la clasificación de gestos en tiempo real en un entorno muy variable. Para ello sería necesario generar un dataset de un tamaño varios órdenes de magnitud

superior al utilizado para esta prueba piloto, que incluyese operarios de diversas complexiones, vestimenta y sobretodo fondos diversos, a fin de que la red adquiriese la capacidad de discriminar información no esencial y terminase fijándose solo en los puntos clave del cuerpo humano.

Entendemos que esto no es práctico y que la solución propuesta y ampliamente defendida por este trabajo en capítulos anteriores, basada en el uso de una primera red convolucional pre entrenada y específicamente desarrollada para la segmentación de seres humanos y extracción de coordenadas del cuerpo, a la que posteriormente se le agrega una segunda etapa de clasificación centrada exclusivamente en el reconocimiento de cada gesto, es un método mucho más sencillo, óptimo y escalable.

6.5. Caso de uso: “Operación Nocturna” .

Hay un caso de uso en el que este segundo método, basado exclusivamente en una arquitectura convolucional para la clasificación de los gestos tomando como entrada directamente las propias imágenes puede ser muy interesante: las operaciones nocturnas.

Tal y como se abordó en el capítulo segundo, los operadores de rampa en el entorno aeroportuario utilizan unas varitas o linternas que cobran especial relevancia cuando desarrollan su trabajo en franjas del día con escasa visibilidad. En estas condiciones, en las que no se alcanza a distinguir con precisión los contornos de la persona que ejecuta los gestos los pilotos situados en sus cabinas siguen pudiendo identificar y reconocer que indicaciones les hacen desde tierra, gracias a la persistencia visual del ojo humano y las linternas empleadas.

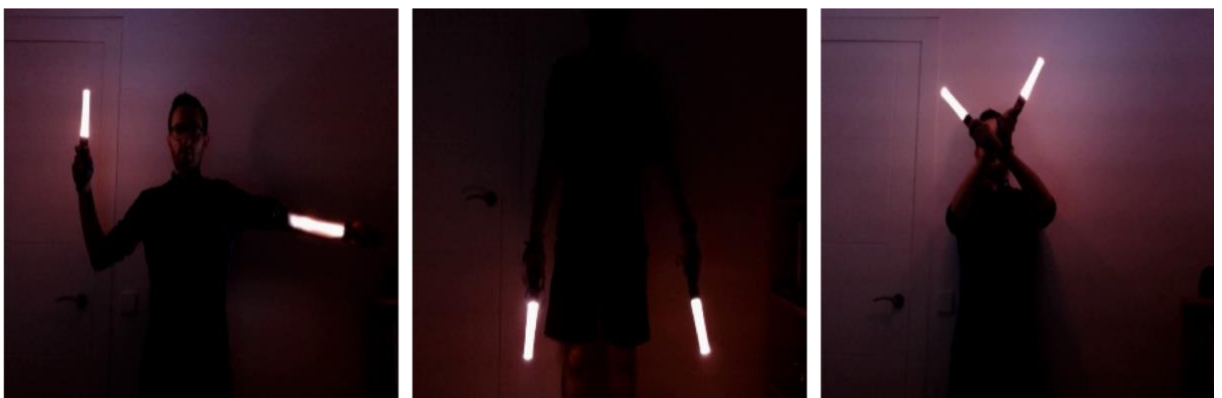


Figura 45. Datase generado para “Operación Nocturna”. Fuente: Elaboración Propia.

Si analizamos este problema con más detalle, observaremos que la estrategia seguida hasta ahora y que tan buenos resultados nos ha dado con una adecuada iluminación, tendría en

estas condiciones serios problemas para identificar las coordenadas que definen la posición de la extremidades. Sin embargo, un modelo basado en una arquitectura convolucional de clasificación de imágenes, al no depender ya la información del fondo ni de la propia fisonomía o vestimenta del operador, podría ser un perfecto aliado para ayudar al reconocimiento de estos gestos, identificando el patrón de luces dejado por las linternas.

Con el fin de poner a prueba esta hipótesis, se volvió a generar un tercer conjunto de datos de entrenamientos en condiciones de muy baja visibilidad. Esta vez solo se capturaron 50 muestras por cada gesto, lo que da como resultado un conjunto total de 300 imágenes etiquetadas. Tal y como se había descrito en el primer modelo, se aplicaron las herramientas de “*Data Augmentation*” para intentar ampliar el número de muestras diferentes que vería la red durante su entrenamiento.

La topología del modelo e hiperparámetros se dejaron también tal y como estaban en el primer caso, delegando el grueso de la extracción de características en la base convolucional pre entrenada basada en ResNet-18. Durante la fase de entrenamiento se identificó una tendencia incluso más alta al sobreajuste de la red. Esto es lógico si tenemos en cuenta que la gran mayoría de píxeles presentes en las pocas imágenes de entrenamiento no codifican prácticamente ninguna información, ya que son eminentemente negros. Con no más de 4 épocas se alcanza un ratio de éxito de 98% y una valor de la función de pérdida de 0.03. Se debe entonces detener el entrenamiento para evitar que nuestro modelo memorice los datos de entrenamiento .

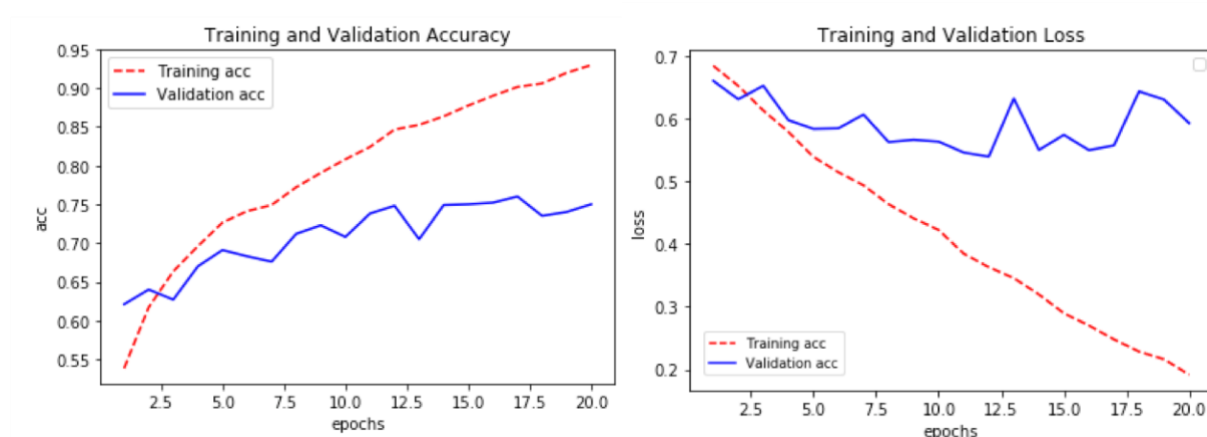


Figura 46. Sobreajuste característico de la red. Fuente: Elaboración Propia.

Durante la fase de evaluación se observó que el efectivamente el nuevo modelo es capaz de predecir las diferentes clases con un elevado ratio de éxito, y que a diferencia de su predecesor entrenado con imágenes con buena visibilidad, éste no se ve afectado por la variación del “operador” o “del fondo”, ya que al ser eminentemente oscuro, no marcan una

diferencia significativa en los patrones que la red había identificado como principales, y que entendemos que se basan en la posición que ocupan las linternas y su estela, en cada una de las capturas.

En los anexos se puede encontrar un enlace al video con los resultados del ensayo llevado a cabo.



Figura 47. Ensayo del modelo en operación nocturna. Fuente: Elaboración Propia.

7. Conclusiones y trabajo futuro

7.1. Conclusiones

Este trabajo se ha centrado en resolver el problema del reconocimiento visual e identificación de un número reducido de las “señales de rampa” que el personal de tierra del aeropuerto lleva a cabo mediante el movimiento específicos de sus brazos para codificar diferentes mensajes hacia los pilotos situados en la cabina, asistiéndoles en las maniobras de rodaje en pista y aparcamiento. Estos miembros del equipo de operaciones terrestres del aeropuerto, conocido generalmente como “señaleros, agentes de rampa o aircraft marshallers”, van equipados con identificadores visuales característicos: chaleco reflectante, paletas de colores o barras de luz en la oscuridad.

Tras una análisis específico del dominio del problema, que ha buscado dar una descripción más detallada sobre el tipo de operación y gestos involucrados, se ha procedido a hacer una defensa de la importancia e interés de una solución como la aquí propuesta. Para ello se han recopilado datos que ponen de relevancia el elevado número de incidentes, y consecuentes pérdidas económicas, durante las operaciones en el entorno del aeropuerto, identificando las causas principales en: poca visibilidad, espacios pequeños, alta concentración de aeronaves y vehículos de asistencia en tierra, y sobre todo, baja conciencia situacional de los pilotos situados en cabina.

Posteriormente, se ha puesto especial atención en la importancia de que, un sector emergente como es el de las plataformas aéreas no tripuladas, pueda integrarse en los espacios e infraestructura ya existentes y reservados a día de hoy al tráfico aéreo tripulado. Además del consecuente alivio de la inversión necesaria, supondría un fuerte ahorro de tiempo. En este sentido se ha explicado que muchos de los sistemas utilizados a día de hoy fueron concebidos para el uso entre humanos, jugando un papel preponderante toda la señalética visual: luces, colores y gestos, como los tratados en este trabajo. Concluimos, pues que no podremos abordar una verdadera integración de este sector en el entorno de operación del aeropuerto actual sin contar con las herramientas adecuadas de visión artificial y clasificación.

No podemos olvidarnos del resto de requisitos propios del sector y que van a condicionar cualquier solución adoptada. En este sentido, se ha analizado las restricciones más importantes que debería tener un sistema de estas características para poder ser de verdadera aplicabilidad industrial. Factores como el tamaño y el peso, pero también otros

como la capacidad computacional disponible, la latencia o la robustez del sistema frente a datos nuevos han de ser tenidos en cuenta. Todos estos factores que caracterizan y limitan el alcance de la solución propuesta se han recopilado en una serie de requisitos específicos y medibles.

Además de la aplicación objetivo, se han planteado otros posibles casos de uso de la misma tecnología, ya bien sea como elemento auxiliar en cabina ayudando a incrementar la conciencia situacional de los pilotos humanos, o instalada en los potenciales vehículos autónomos terrestres del propio aeropuerto, que llevarán a cabo labores de servicios tales como, transporte de equipajes, pasajeros o catering.

Una vez, comprendidas las implicaciones del problema a resolver y tras haber reunido el conjunto de requisitos que caracterizan y limitan el alcance de la solución, se ha llevado a cabo un pormenorizado estudio del arte, identificando aquellos campos de la técnica y trabajos más relevantes que pudieran ser aplicados para la resolución del presente reto. En este sentido, se ha podido comprobar que si bien no existe prácticamente literatura específica que aborde exactamente nuestro problema, si que hay un gran número de publicaciones en el campo de la interacción “hombre máquina”, y en concreto con el subcampo que trata el reconocimiento de gestos, que es de total aplicabilidad para nuestro caso. De todas las técnicas identificadas, se ha concluido que solo aquellas basadas en el procesamiento de imagen y que no dependan de elementos auxiliares avanzados, como pudieran ser aquellas basadas en sensores inerciales instalados en la ropa de los operarios o cámaras dotadas de sensores de profundidad, son las que más ventajas presentan al permitir el desarrollo de soluciones más flexibles, baratas y autónomas.

Dentro de las técnicas de reconocimiento y clasificación de imágenes se ha hecho una breve revisión histórica que concluye reconociendo las ventajas indiscutibles, en términos de potencia y versatilidad, que las nuevas soluciones basadas en redes neuronales artificiales ponen a nuestra disposición. Ya centrados en esta tecnología de inminente actualidad, se han comparado diferentes estrategias posibles para abordar el problema aquí planteado. En este sentido se han valorado las ventajas e inconvenientes de la utilización de topologías más complejas, como las redes convolucionales 3D para la clasificación de videos o las redes recurrentes para el reconocimiento de secuencias de gestos, frente a implementaciones más simples basadas en topologías convolucionales bidimensionales.

Llegados a este punto, y tras una valoración justificada que ha tenido en cuenta las características del problema a resolver, los requisitos esperados del sistema así como las

ventajas e inconvenientes de cada una de las técnicas y herramientas del estado del arte identificadas, se ha presentado la estrategia técnica a seguir por este proyecto. En este sentido, se ha hecho una descripción de la arquitectura software y hardware propuesta. En especial, se ha presentado un plan para superar la limitación de no disponer de un conjunto de datos ya existente que contuviese los gestos a clasificar debidamente etiquetados.

Desde el punto de vista de hardware, se han identificado las características más destacables de los componentes principales del sistema: cámara RGB y ordenador embebido.

Por el lado del software, se ha optado por una topología basada en dos modelos en serie:

- “*CPM - Convolutional Pose Machine*”: Se trata de un modelo basado en una topología ResNet-18 pre entrenada contra una basta colección de datos, capaz de extraer desde una imagen la segmentación del ser humano y dar como salida del modelo las coordenadas de las articulaciones principales del cuerpo.
- “*Clasificador*”: se trata de un modelo ajeno e independiente del paso anterior, cuya tarea es la clasificación de las coordenadas extraídas del paso previo en una de las cinco categorías identificadas.

Para la implementación de este clasificador se optó por el aprendizaje supervisado, y para ello se tuvo que desarrollar un conjunto de datos propios con más de 4500 muestras, mediante la ayuda de 3 voluntarios que repitieron varias veces cada clase de gesto. Se han descrito específicamente todos los trabajos necesarios para procesar estos datos y dejarlos preparados para su uso en la fase de entrenamiento de los modelos. El dataset generado se ha alojado en un repositorio público para que los resultados del presente trabajo puedan ser reproducidos.

En cuanto al clasificador, se han propuesto e implementado dos modelos diferentes:

- *Random Forest*: Modelo basado en el ensamble de árboles de decisión, con una excelente capacidad de generalización y predicción, incluso con conjuntos de datos no demasiado grandes.
- *ANN*: Modelo basado en una red artificial densamente conectada y no muy profunda, con excelentes propiedades como aproximador universal.

Mediante una porcentaje de los datos reservados para tal fin, se han evaluado las prestaciones de ambos modelos, utilizando para ello las métricas de “Matriz de confusión”, “Ratio de éxito” y “F-Score”. En este sentido, ambos modelos han arrojado un excelente resultado, alcanzando una valoración similar en todas ellas y un ratio de acierto del 96%.

A continuación, se han puesto a prueba en un entorno de inferencia en tiempo real, ejecutadas en el ordenador embebido escogido, con datos no vistos con anterioridad. Si bien ambos

modelos han funcionado correctamente, el modelo basado en la red artificial sobresale respecto al otro, dando una respuesta menos ruidosa y con menor incertidumbre en la predicción. Sin embargo, es justo destacar que para el modelo de red neuronal fue necesario ejecutar una acción intermedia para su optimización sobre GPU. Sin este paso, el modelo no optimizado era de media 3 veces más lento que su homólogo basado en “Random Forest”. Esto es fundamental si no se dispone del hardware adecuado para la etapa de inferencia, ya que probablemente sería más conveniente evitar el uso del modelo “ANN”.

Por último, y aunque la arquitectura expuesta ha dado excelentes resultados, se ha presentado otra solución diferente, basada esta vez en una red convolucional clásica de clasificación, preentrenada sobre un dataset de gran tamaño, a la cual se le ha sustituido la última capa, para añadir el clasificador de nuestro propio problema. A pesar de partir de una red en la que la mayoría de parámetros ya han sido calculados de partida, es necesario generar un nuevo conjunto de datos que permita entrenar dicha nueva capa clasificadora. En este sentido, además de la generación de un nuevo dataset, se han empleado técnicas de aumentación de datos que permiten modificar ligeramente las muestras que verá la red, con el objetivo de introducir cierta variabilidad y mejorar la tendencia al sobreajuste del modelo.

Es importante recalcar que, a diferencia de la arquitectura anterior en la que había dos modelos en serie con roles claramente diferenciados, en esta nueva arquitectura se presenta una única red cuyo parámetro de entrada es directamente la matriz de píxeles que componen cada imagen capturada por la cámara RGB, y su salida la probabilidad de pertenencia a cada una de las clases.

Tras implementar y entrenar el nuevo modelo, se consiguen buenos resultados en la fase de evaluación del sistema en tiempo real. Sin embargo, debido al reducido número de imágenes empleadas en la fase de entrenamiento, el modelo está muy sobreajustado y no es capaz de generalizar ante nuevos datos, no comportándose de manera adecuada cuando el operador o el fondo distan mucho de las imágenes de entrenamiento.

Visto el resultado anterior y aunque la solución que nos brinda la primera arquitectura es mucho más robusta y menos dependiente del fondo, se identifica un claro caso de uso en el que un modelo como el de la segunda arquitectura propuesta, sería de gran utilidad: la operación nocturna. Se pone a prueba esta idea, generando un tercer dataset con imágenes con poca luminosidad en las que el operario emplea las linternas características de la profesión para señalar los gestos en la oscuridad.

Tras la fase de entrenamiento se confirma la hipótesis evaluando el modelo en tiempo real: al ser la imagen eminentemente oscura, las características del fondo o el operario son poco informativas, y la red se centra en extraer características de la posición que ocupan las linternas en cada gesto, haciendo este modelo muy útil para reconocer gestos en los que debido a la baja intensidad de luz, el modelo basado en “CPM” no pueda llevar a cabo la segmentación del ser humano sobre la imagen capturada.

Con la información técnica recogida en esta memoria, el repositorio público que contienen el dataset generado y los códigos fuentes desarrollados, así como con los enlaces a los videos de las diferentes pruebas ejecutadas durante la validación, podemos concluir que existen las evidencias necesarias para justificar que este proyecto ha alcanzado el objetivo de partida, y que la solución aquí propuesta cumple con los requisitos técnicos y operativos que se perseguían.

Queda, además, contestada la pregunta que se planteaba al inicio del proyecto: es posible, y en este proyecto se ha llevado a cabo un validador tecnológico que lo demuestra, el empleo de técnicas de inteligencia artificial y visión por ordenador para la correcta identificación y clasificación de las señales de rampa llevadas a cabo por el personal de tierra en las operaciones en el entorno del aeropuerto.

7.2. Líneas de trabajo futuro

Aunque se han conseguido los objetivos del trabajo, si queremos convertir el validador tecnológico actual en una solución industrial fiable, debemos abordar nuevos retos técnicos en aras de una mejora en la robustez y sobretodo en la escalabilidad del sistema, mejorando tanto el dataset como las herramientas utilizadas.

Una clara línea de mejora sería el desarrollo de una solución que combinase e integrase ambas arquitecturas presentadas. Esto dotaría al sistema final de la capacidad de poder trabajar en diferentes situaciones, a pesar de que la iluminación no sea la adecuada. Aunque este desarrollo merece una reflexión más sosegada, se podría pensar en una primera capa de procesamiento que valorase la cantidad de píxeles oscuros presentes en cada imagen. A medida que este valor va aumentando se podrían ponderar con una figura de mérito las predicciones de ambos modelos, dando progresivamente más peso al modelo basado en la clasificación puras de imágenes y que tan bien ha respondido en ambientes nocturnos.

Se han quedado otras muchas pruebas interesantes sin poder ser ejecutadas. Una de las más prometedoras y que podría ayudar a mejorar los resultados del clasificador de gestos de la primer arquitectura propuesta, para ambos modelo ensayados, es la agregación de “features” adicionales a las 18 coordenadas calculadas por “CPM”. En este sentido, creemos que añadir variables que computen la velocidad con la que se están desplazando los puntos singulares podría ser la clave para distinguir entre gestos en aquellos casos en los que los modelos actuales están fallando. Para llevar a cabo este cálculo se debería tener en cuenta las coordenadas extraídas del fotograma anterior y el tiempo transcurrido entre capturas, elementos esenciales para poder calcular una derivada discreta.

Por último, también debemos reconocer que, desde un punto de vista operacional, el sistema actual no sería capaz de extraer información sobre la velocidad a la que se está ejecutando el gesto clasificado. Es decir, si el operario quisiese comunicar que el viraje hacia la izquierda debe ser más lento o más rápido, podría cambiar el ritmo o cadencia con la que mueve su brazo. Sin embargo, a día de hoy, el sistema solo reconocería que la indicación se corresponde con un “viraje a izquierdas”, sin poder extraer información adicional sobre la velocidad recomendada para dicho giro.

Una solución que proponemos es el cálculo del ángulo formado por el brazo y su antebrazo utilizando las coordenadas que obtenemos del modelo “CPM”. Con esas variables podríamos estimar una frecuencia de oscilación media del brazo por medio de una “FFT”. Finalmente, se podría codificar ese valor puramente cuantitativo por medio de lógica difusa, en información cualitativa de tal manera que nos avisase de si el movimiento parece “más rápido o más lento” de lo habitual.

8. Bibliografía

- Breiman, L. (2001). Random forests. *Random Forests*, 1–122. <https://doi.org/10.1201/9780367816377-11>
- Cao, Z., Hidalgo Martinez, G., Simon, T., Wei, S.-E., & Sheikh, Y. A. (2019). OpenPose: Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. <https://doi.org/10.1109/tpami.2019.2929257>
- Cao, Z., Simon, T., Wei, S. E., & Sheikh, Y. (2017). Realtime multi-person 2D pose estimation using part affinity fields. *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, 2017-Janua*, 1302–1310. <https://doi.org/10.1109/CVPR.2017.143>
- Castillo, J. C., Alonso-Martín, F., Cáceres-Domínguez, D., Malfaz, M., & Salichs, M. A. (2019). The Influence of Speed and Position in Dynamic Gesture Recognition for Human-Robot Interaction. *Journal of Sensors*, 2019. <https://doi.org/10.1155/2019/7060491>
- Choi, C., Ahn, J. H., & Byun, H. (2008). Visual recognition of aircraft marshalling signals using gesture phase analysis. *IEEE Intelligent Vehicles Symposium, Proceedings*, 853–858. <https://doi.org/10.1109/IVS.2008.4621186>
- Civil Aviation Authority (CAA). (1996). Visual aids handbook. *Aids*, 10(6), 690–691. <https://doi.org/10.1097/00002030-199606000-00024>
- Dalal, N., & Triggs, B. (2005). Histograms of oriented gradients for human detection. *Proceedings - 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR 2005*. <https://doi.org/10.1109/CVPR.2005.177>
- Demarco, K. J., West, M. E., & Howard, A. M. (2014). Underwater human-robot communication: A case study with human divers. *Conference Proceedings - IEEE International Conference on Systems, Man and Cybernetics, 2014-Janua*(January), 3738–3743. <https://doi.org/10.1109/smc.2014.6974512>
- Donahue, J., Hendricks, L. A., Rohrbach, M., Venugopalan, S., Guadarrama, S., Saenko, K., & Darrell, T. (2017). Long-Term Recurrent Convolutional Networks for Visual Recognition and Description. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(4), 677–691. <https://doi.org/10.1109/TPAMI.2016.2599174>
- European Union Aviation Safety Agency. (2020). *Artificial Intelligence Roadmap: A Human-*

- Centric Approach to AI in Aviation. February, 1–33.*
<https://www.easa.europa.eu/newsroom-and-events/news/easa-artificial-intelligence-roadmap-10-published>
- Fang, H. S., Xie, S., Tai, Y. W., & Lu, C. (2017). RMPE: Regional Multi-person Pose Estimation. *Proceedings of the IEEE International Conference on Computer Vision*.
<https://doi.org/10.1109/ICCV.2017.256>
- Hara, K., Kataoka, H., & Satoh, Y. (2018). Can Spatiotemporal 3D CNNs Retrace the History of 2D CNNs and ImageNet? *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 6546–6555.
<https://doi.org/10.1109/CVPR.2018.00685>
- He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2016-Decem*, 770–778. <https://doi.org/10.1109/CVPR.2016.90>
- ICAO. (2005). *Annex 2 - Rules of the Air - Tenth Edition* (Issue November). <http://www.icao.int>
- Joo, H., Liu, H., Tan, L., Gui, L., Nabbe, B., Matthews, I., Kanade, T., Nobuhara, S., & Sheikh, Y. (2015). Panoptic studio: A massively multiview system for social motion capture. *Proceedings of the IEEE International Conference on Computer Vision*.
<https://doi.org/10.1109/ICCV.2015.381>
- Kanazawa, A., Black, M. J., Jacobs, D. W., & Malik, J. (2018). End-to-End Recovery of Human Shape and Pose. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. <https://doi.org/10.1109/CVPR.2018.00744>
- Kapuscinski, T., Oszust, M., Wysocki, M., & Warchol, D. (2015). Recognition of hand gestures observed by depth cameras. *International Journal of Advanced Robotic Systems*, 12.
<https://doi.org/10.5772/60091>
- Kim, J. H., Thang, N. D., & Kim, T. S. (2009). 3-D hand motion tracking and gesture recognition using a data glove. *IEEE International Symposium on Industrial Electronics, ISIE*, 1013–1018. <https://doi.org/10.1109/ISIE.2009.5221998>
- Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). ImageNet classification with deep convolutional neural networks. *Advances in Neural Information Processing Systems*.
- Lin, T. Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., & Zitnick, C. L. (2014). Microsoft COCO: Common objects in context. *Lecture Notes in Computer*

- Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 8693 LNCS(PART 5), 740–755. https://doi.org/10.1007/978-3-319-10602-1_48
- Martin, J. B., & Moutarde, F. (2019). Real-Time Gestural Control of Robot Manipulator Through Deep Learning Human-Pose Inference. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 11754 LNCS, 565–572. https://doi.org/10.1007/978-3-030-34995-0_51
- Molchanov, P., Yang, X., Gupta, S., Kim, K., Tyree, S., & Kautz, J. (2016). Online Detection and Classification of Dynamic Hand Gestures with Recurrent 3D Convolutional Neural Networks. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2016-Decem, 4207–4215. <https://doi.org/10.1109/CVPR.2016.456>
- Pullen, P., & Seffens, W. (2018). Machine learning gesture analysis of yoga for exergame development. *IET Cyber-Physical Systems: Theory & Applications*, 3(2), 106–110. <https://doi.org/10.1049/iet-cps.2017.0027>
- Raheja, J. L., Minhas, M., Prashanth, D., Shah, T., & Chaudhary, A. (2015). Robust gesture recognition using Kinect: A comparison between DTW and HMM. *Optik*. <https://doi.org/10.1016/j.ijleo.2015.02.043>
- Ribó, A., Warchol, D., & Oszust, M. prz edu pl. (2016). An approach to gesture recognition with skeletal data using dynamic time warping and nearest neighbour classifier. *International Journal of Intelligent Systems and Applications*, 8(6), 1–8. <https://doi.org/10.5815/ijisa.2016.06.01>
- Schneider, P., Memmesheimer, R., Kramer, I., & Paulus, D. (2019). Gesture Recognition in RGB Videos Using Human Body Keypoints and Dynamic Time Warping. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 11531 LNAI, 281–293. https://doi.org/10.1007/978-3-030-35699-6_22
- Shannon, C. E. (1997). The Mathematical Theory of Communication. *M.D. Computing*. <https://doi.org/10.2307/410457>
- Simonyan, K., & Zisserman, A. (2014a). Two-stream convolutional networks for action recognition in videos. *Advances in Neural Information Processing Systems*.
- Simonyan, K., & Zisserman, A. (2014b). VGG-16. *ArXiv Preprint*.

<https://doi.org/10.1016/j.infsof.2008.09.005>

- Singh, M., Mandal, M., & Basu, A. (2005). Visual gesture recognition for ground air traffic control using the radon transform. *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS*, 2850–2855. <https://doi.org/10.1109/IROS.2005.1545408>
- Song, Y., Demirdjian, D., & Davis, R. (2011). Tracking body and hands for gesture recognition: NATOPS aircraft handling signals database. *2011 IEEE International Conference on Automatic Face and Gesture Recognition and Workshops, FG 2011*, 500–506. <https://doi.org/10.1109/FG.2011.5771448>
- Tomaszewska, J., Zieja, M., Woch, M., & Krzysiak, P. (2018). Statistical analysis of ground-related incidents at airports. *Journal of KONES*, 25(3), 467–472. <https://doi.org/10.5604/01.3001.0012.4369>
- Viola, P., & Jones, M. (2001). Rapid object detection using a boosted cascade of simple features. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. <https://doi.org/10.1109/cvpr.2001.990517>
- Waldherr, S., Romero, R., & Thrun, S. (2000). Gesture based interface for human-robot interaction. *Autonomous Robots*, 9(2), 151–173. <https://doi.org/10.1023/A:1008918401478>
- Wei, S. E., Ramakrishna, V., Kanade, T., & Sheikh, Y. (2016). Convolutional pose machines. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2016-Decem*, 4724–4732. <https://doi.org/10.1109/CVPR.2016.511>
- Zaman Khan, R. (2012). Hand Gesture Recognition: A Literature Review. *International Journal of Artificial Intelligence & Applications*, 3(4), 161–174. <https://doi.org/10.5121/ijaia.2012.3412>
- Zhou, B., Andonian, A., Oliva, A., & Torralba, A. (2018). Temporal Relational Reasoning in Videos. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 11205 LNCS, 831–846. https://doi.org/10.1007/978-3-030-01246-5_49

Anexos

Anexo I. Enlaces al material complementario del proyecto.

- *Repositorio GitHub Público del proyecto:* “Dataset” y “Source code”.
https://github.com/astromaf/ramp_hand_signals_recognition
- *Video Nº1:* Ejecución en tiempo real arquitectura basada en: CPM+ANN.
<https://www.youtube.com/watch?v=gknhEQL33qg>
- *Video Nº2:* Ejecución en tiempo real arquitectura basada en: CPM+RF.
<https://www.youtube.com/watch?v=VEt8DadGwLU&feature=youtu.be>
- *Video Nº3:* Ejecución en tiempo real arquitectura basada en: CCN (Night Operation).
<https://www.youtube.com/watch?v=aJ8TJzfSGjs&feature=youtu.be>

Anexo II. Artículo de investigación

En este anexo, de carácter obligatorio, se incluye un resume del trabajo realizado y los principales resultados obtenidos, siguiendo una plantilla de artículo de investigación.

Reconocimiento visual en tiempo real de las “señales de rampa” aplicado a UAVs para operaciones en tierra.

Miguel Ángel de Frutos Carro

Universidad Internacional de la Rioja, Logroño (España)

Julio de 2020

PALABRAS CLAVE

Aircraft Marshalling signals, Convolutional Pose Machines, Gesture Recognition, Real-time, UAV .

RESUMEN

Lo vehículos aéreos no tripulados ya son una realidad en nuestros días. El siguiente gran hito tecnológico y operativo al que se enfrenta el sector es el de la convivencia simultánea de plataformas tripuladas y no tripuladas en los espacios actuales existentes. Unos de los mayores retos se presenta en las operaciones terrestres, diseñadas originalmente para interacción entre humanos.

El objetivo principal de este trabajo es el diseño y validación de un software de reconocimiento visual e identificación en tiempo real, de un número limitado de las “señales de rampa” que el personal de tierra, lleva a cabo mediante el movimiento específicos de sus brazos, enfocado para su uso en UAVs u otras plataformas autónomas.

Se hará uso de las técnicas actuales basadas en el uso de redes neuronales pre-entrenadas para la predicción secuencial de la postura de un ser humano y de métodos adicionales para la clasificación de los gestos.

I. INTRODUCCIÓN

ESTE trabajo se ha centrado en resolver el problema del reconocimiento visual e identificación de un número reducido de las “señales de rampa” que el personal de tierra del aeropuerto lleva a cabo mediante el movimiento específicos de sus brazos para codificar diferentes mensajes hacia los pilotos situados en la cabina, asistiéndoles en las maniobras de rodaje en pista y aparcamiento. Estos miembros del equipo de operaciones terrestres del aeropuerto son conocido generalmente como “señaleros, agentes de rampa o aircraft marshallers”.

A continuación se ha hecho análisis específico del dominio del problema, que ha buscado dar una descripción más detallada sobre el tipo de operación y gestos involucrados. Una vez, comprendidas las implicaciones del problema a resolver y tras haber reunido el conjunto de requisitos que caracterizan y limitan el alcance de la solución, se ha llevado a cabo un pormenorizado estudio del arte, identificando aquellos campos de la técnica y trabajos más relevantes que pudieran ser aplicados para la resolución del presente reto. En este sentido, se ha podido comprobar que si bien no existe prácticamente literatura específica que aborde exactamente nuestro problema, si que hay un gran número de publicaciones en el campo de la interacción “hombre máquina”, y en concreto en el subcampo que trata el “reconocimiento de gestos”, que es de total aplicabilidad para nuestro caso.

Llegados a este punto, y tras una valoración justificada que ha

tenido en cuenta las características del problema a resolver, los requisitos esperados del sistema así como las ventajas e inconvenientes de cada una de las técnicas y herramientas del estado del arte identificadas, se ha presentado la estrategia técnica a seguir por este proyecto.

Desde el punto de vista de hardware, se han identificado las características más destacables de los componentes principales del sistema: cámara RGB y ordenador embebido. Por el lado software se ha optado por una topología basada en dos modelos en serie:

- **“CPM - Convolutional Pose Machine”**: Se trata de un modelo pre entrenado contra una basta colección de datos, capaz de extraer desde una imagen la segmentación del ser humano y dar como salida del modelo las coordenadas de las articulaciones principales del cuerpo.
- **“Clasificador”**: se trata de un modelo ajeno e independiente del paso anterior, cuya tarea es la clasificación de las coordenadas extraídas del paso previo en una de las cinco categorías identificadas.

Para la implementación de este clasificador se optó por el aprendizaje supervisado, y para ello se ha tenido que desarrollar un conjunto de datos propios con más de 4500 muestras, mediante la ayuda de 3 voluntarios que repitieron varias veces cada clase de gesto.

En cuanto al clasificador, se han propuesto e implementado dos modelos diferentes:

- *Random Forest*: Modelo basado en el ensamble de árboles de decisión.
- *ANN*: Modelo basado en una red artificial densamente conectada y no muy profunda.

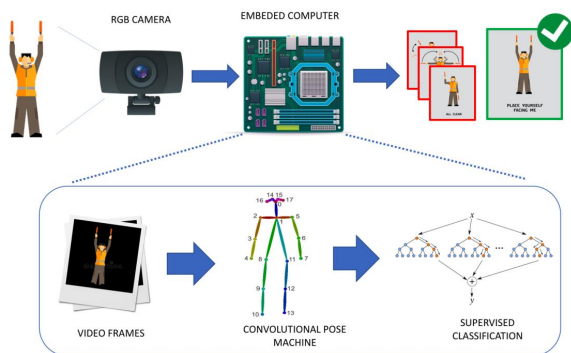


Fig. 1. La arquitectura de alto nivel de la solución queda definida por 2 elementos hardware (cámara y unidad de procesamiento) y tres funciones principales software (captura, “CPM” y clasificación).

Mediante una porcentaje de los datos reservados para tal fin, se han evaluado las prestaciones de ambos modelos, utilizando para ello las métricas de “Matriz de confusión”, “Ratio de éxito” y “F-Score”. En este sentido, ambos modelos arrojaron un excelentes resultados, alcanzando una valoración similar en todas las métricas y un ratio de acierto del 96%.

A continuación, se han puesto a prueba en un entorno de inferencia en tiempo real, ejecutadas en el ordenador embebido escogido, con datos no vistos con anterioridad. Si bien ambos modelos han funcionado correctamente, el modelo basado en la red artificial sobresale respecto al otro, dando una respuesta menos ruidosa y con menor incertidumbre en la predicción. Sin embargo, es justo destacar que para el modelo de red neuronal fue necesario ejecutar una acción intermedia para su optimización sobre GPU.

Por último, y aunque la arquitectura expuesta ha dado excelente resultados, se ha presentado otra solución diferente, basada esta vez en un red convolucional clásica de clasificación, pre entrenada sobre un dataset de gran tamaño, a la cual se le ha sustituido la última capa, para añadir el clasificador de nuestro propio problema. A pesar de partir de una red en la que la mayoría de parámetros ya han sido calculados de inicio, es necesario generar un nuevo conjunto de datos que permita entrenar dicha nueva capa clasificadora. En este sentido, además de la generación de un nuevo dataset, se han empleado técnicas de aumentación de datos que permiten modificar ligeramente las muestras que verá la red, con el objetivo de introducir cierta variabilidad y mejorar la tendencia al sobreajuste del modelo.

Es importante recalcar que, a diferencia de la arquitectura anterior en la que había dos modelos en serie con roles claramente diferenciados, en esta nueva arquitectura se presenta una única red cuyo parámetro de entrada es directamente la matriz de píxeles que componen cada imagen capturada por la cámara RGB, y su salida la probabilidad de pertenencia a cada una de las clases.

Tras implementar y entrenar el nuevo modelo, se consiguen

buenos resultados en la fase de evaluación del sistema en tiempo real. Sin embargo, debido al reducido número de imágenes empleadas en la fase de entrenamiento, el modelo está muy sobre ajustado y no es capaz de generalizar ante nuevos datos, no comportándose de manera adecuada cuando el operador o el fondo distan mucho de las imágenes de entrenamiento. Sin embargo, se identifica un claro caso de uso en el que un modelo como el de la segunda arquitectura propuesta, sería de gran utilidad: la operación nocturna. Se pone a prueba esta idea, generando un tercer dataset con imágenes con poca luminosidad en las que el operario emplea las linternas características de la profesión para señalar los gestos en la oscuridad.

Tras la fase de entrenamiento se confirma la hipótesis evaluando el modelo en tiempo real: al ser la imagen eminentemente oscura, las características del fondo o el operario son poco informativas, y la red se centra en extraer características de la posición que ocupan las linternas en cada gesto, haciendo este modelo muy útil para reconocer gestos en los que debido a la baja intensidad de luz, el modelo basado en “CPM”, de la arquitectura original, no pueda llevar a cabo la segmentación del ser humano sobre la imagen capturada.

Finalmente se hace una reflexión sobre los objetivos alcanzados, destacando los logros y contribuciones del proyecto, así como, una serie de propuestas de mejora que buscan dotar a la solución aquí presentada de mayor robustez y una mejor escalabilidad.

II. ESTADO DEL ARTE

○ *Dominio del problema: “Aircraft Marshalling”.*

“Señalero” o “Aircraft Marshaller” es la designación oficial [1] para referirse al miembro del equipo que forma parte del personal de tierra y que ayuda a los pilotos en la ejecución de determinadas maniobras cuando éstas entrañan un riesgo de incidente, siendo un complemento a las indicaciones que los controladores de superficie hacen llegar a los pilotos. Esta figura está también presente en el sector de la defensa, recibiendo el nombre de “personal de vuelo en cubierta” o “Shooters”, cuando realizan sus funciones en portaviones. El equipamiento habitual de estos operadores incluye: un chaleco de alta visibilidad, unos cascos de protección auditiva y dos “varitas o linternas” de maniobra.

Se conocen como “señales de rampa” al conjunto de gestos que el personal de tierra lleva a cabo mediante el movimiento específicos de sus brazos para codificar los diferentes mensajes hacia los pilotos situados en la cabina, desde donde por lo general disponen de una mala visión del espacio alrededor del avión durante la fase de rodadura y parking. Existen diferentes conjunto de gestos dependiendo de su ámbito de aplicación, así pues podemos encontrar ciertas discrepancias entre la señalética utilizada en la pista de un porta aviones de la OTAN [2] o las empleadas en los aeropuertos internacionales de uso civil [3].

La Organización de Aviación Civil, o por sus siglas en inglés ICAO, promulga una serie de recomendaciones de obligado cumplimiento para los países adheridos a ella, con el fin de establecer un estándar internacional en todos los aspectos referentes al transporte aéreo. Este conjunto es el más popular y extendido. Merece la pena señalar en este punto las características principales de los gestos:

- Solo intervienen las extremidades superiores.
- No son estáticos, pero la información espacial predomina a la temporal.
- Ambos brazos codifican información.
- Son visualmente independientes.

- *"Interacción Humano-Robot"*

La interacción entre hombre y máquina, generalmente referido en inglés como HRI ("Human-Robot Interaction") [4] es un campo de estudio multidisciplinar que aborda el diseño de mecanismos de comunicación más eficientes y efectivos entre humanos y máquinas, que permita a ambos colaborar en la ejecución de tareas.

El objetivo final es identificar y desarrollar aquellas técnicas que puedan permitir una comunicación entre los hombre y los robots de su entorno de tal manera que no sea necesario usar telecomandos específicos. Es un campo de gran interés y dificultad, en el que se estudia tanto la manera en la que los propios humanos nos comunicamos e interaccionamos entre nosotros, de manera explícita e implícita, así como los mecanismos y procesos necesarios para implementar habilidades semejantes en las máquinas dotadas de un cierto nivel de inteligencia. Esta comunicación puede usar uno o varios canales de manera simultánea, entrando en juego los sistemas de procesamiento del lenguaje natural, sistemas de visión artificial, reconocimiento y clasificación de gestos etc.

Desde un punto de vista industrial, dotar de capacidades de este tipo a las máquinas supondría una mejora en la eficiencia, al permitir una rápida coordinación y cooperación entre operadores, a la vez de un incremento en la seguridad del trabajo. En este sentido algunos autores [5] introducen las actuales limitaciones que, por ejemplo, se imponen en las líneas de producción a la presencia de operarios humanos cuando los brazos robots están funcionando para evitar accidentes. Un sistema capaz de interactuar de forma eficiente con el operador sería una ventaja en términos de seguridad.

El campo de los sistemas autónomos también está muy interesado en esta área y sus avances. El equipo de investigación en IA de NVIDIA® defiende en su trabajo [6] como una comunicación no verbal entre coche autónomo y humano, basada en el uso de determinados gestos, o expresiones fáciles incluso, conduciría a un aumento de la seguridad y confort de los pasajeros.

En este sentido, destaca también el trabajo del equipo de investigación del "Georgia Tech Institute" sobre comunicaciones bajo el agua entre submarinistas y vehículos de asistencia subacuática [7]. Es relevante en tanto que en este tipo de aplicaciones cualquier otro medio de comunicación, como la propia voz u otras formas de ondas, no son posibles al verse fuertemente atenuadas por el agua. Este trabajo analiza el problema y presenta una solución basada en la codificación de ciertas señales mediante un código de luces que permiten la comunicación con el robot subacuático.

El problema que el presente trabajo aborda se circunscribe dentro de este campo en tanto que intenta resolver la comunicación entre los operadores de rampa y los vehículos autónomos a su alrededor sin utilizar controles o mandos remotos. Esta interacción se clasificaría como comunicación explícita, en la medida en que los mensajes codificados son únicos e identificables de manera biunívoca, y al no jugar un papel importante la comunicación implícita más propia de valoraciones subjetivas o en las que se tengan que hacer asunciones basadas en los rasgos faciales o manera de realizar los gestos.

- *Reconocimiento de Gestos.*

El reconocimiento de gestos es un área profusamente estudiada durante las últimas dos décadas. Esto se debe a la gran cantidad de información que codificamos de manera habitual mediante gestos, y que complementan la comunicación verbal o directamente la sustituyen cuando esta no es posible, debido a limitaciones del canal [7] o de los propios agentes involucrados en el proceso (e.g. pérdida de capacidad auditiva).

Las aplicaciones actuales de las técnicas de "reconocimiento de gestos" son muchas y muy variadas, desde interacción con sistemas autónomos, domótica, video juegos o robots socio-sanitarios. Un campo de estudio muy habitual es el del reconocimiento de la "lengua de signos" que permite la comunicación natural por medios no orales a través de un canal visual mediante el uso de gestos con las manos. Una completa revisión del tema, así como las diferentes estrategias presentadas en los últimos años puede encontrarse en [8].

Una primera distinción que encontramos se puede referir a, si las técnicas y herramientas de reconocimiento se centran en zonas concretas del cuerpo, como podría ser el caso de las manos en el ejemplo expuesto anteriormente para la identificación y reconocimiento de "lengua de signos", o por el contrario procesan información del cuerpo entero. Esto es relevante en tanto que limita la precisión o nivel de detalle esperado. Así por ejemplo, un sistema como el presentado por [9] en el que únicamente se persigue clasificar posturas corporales no se analiza la posición de los dedos de la mano, mientras que en la aplicación presentada por [6] para la mejora de la conducción de coches autónomos, por el contrario, la posición de la mano y sus diferentes falanges predomina a la información corporal al estar sentado el conductor.

A continuación, podemos establecer una segunda clasificación entre las técnicas que nos permiten distinguir posturas o señas en las que la información espacial predomina sobre la temporal, de aquellas otras en las que juega un papel importante la propia dinámica del gesto, es decir, la velocidad de ejecución codifica también información. Las primeras se diferencian de las segundas por ser más simples y sencillas, al no tener que prestar atención a la dimensión temporal, que suele adquirir la forma de una secuencia de datos relacionados.

La clasificación de gestos dinámicos es más compleja al requerir establecer dicha relación en una secuencia. Un excelente reflexión sobre la influencia de la velocidad y la posición en el reconocimiento dinámico de gestos se puede encontrar en [4].

Encontramos diferentes propuestas a lo largo de la historia que abordan el problema de la secuencias de gestos encadenados. Destaca entre otros el uso de "Modelos ocultos de Markov-HMM" [10] o el "Problema de la subsecuencia común más larga" [11]. Sin embargo, sobre todos los demás destaca el uso de técnicas basadas en "Dynamic Time Warping". Esta técnica permite comparar dos secuencias con escalas de tiempos diferentes y determinar su grado de similitud. Junto con un clasificador que ayude a determinar cuál de las secuencias de referencia es la más similar permite conseguir resultados óptimos en la clasificación de gestos dinámicos. Son varios los autores que han seguido este enfoque tanto para gestos de cuerpo entero [9] como solo de la mano [12].

En la actualidad han surgido nuevas técnicas basadas en el uso de redes neuronales, especialmente aquellas destinadas a la clasificación de video, que tratan el problema de manera secuencial y no como una clasificación aislada de fotogramas. En este sentido encontramos diferentes enfoques que combinan múltiples redes. Destaca el trabajo de [13] que propone extraer las características de cada fotograma de video por medio de una red convolucional (CNN) para luego pasar la secuencia a una segunda red recurrente (RNN) independiente. Una arquitectura similar la encontramos en el trabajo de [14] pero en este caso, las "features" extraídas por la red convolucional se pasan a un "Multi Layer Perceptron" (MLP). Esta arquitectura se basa en la hipótesis de que una "MLP" podrá inferir las características temporales de la secuencia de forma natural, sin tener que saber que es una secuencia. Finalmente debemos mencionar el trabajo basado en redes convolucionales en tres dimensiones (3D-CNN), que son capaces de extraer características gráficas directamente de un conjunto de fotogramas con relación temporal entre ellos,

identificando una representación de bajo nivel del conjunto por medio de operaciones convolucionales en 3D. Este enfoque ha dado excelentes resultados como se puede ver en el trabajo presentado por el equipo de investigación de NVIDIA® [6], pero también presenta algunas desventajas importantes, tal y como analiza [15] respecto a las anteriores arquitecturas basadas en redes bidimensionales. Por un lado, las 3D-CNN son mucho más demandantes en términos computacionales, tanto de recursos de cálculo como de memoria, lo que a pesar de los grandes avances en el estado actual de la tecnología limita su uso para aplicaciones en tiempo real o en plataformas embebidas. Por otro lado, a día de hoy no existen tantas redes pre-entrenadas para la extracción de características en videos, como si existen numerosas y muy potentes redes entrenadas usando arquitecturas 2D-CNN.

Volviendo de nuevo al análisis del reconocimiento de gestos, por último, podemos nuevamente clasificar las técnicas de reconocimiento de gestos presentadas hasta la fecha en dos categorías: colaborativas o no colaborativas. Entendemos por técnicas colaborativas, o también invasivas, aquellas que basan su funcionamiento en el uso de hardware adicional que el usuario que ejecuta la acción debe llevar puesto con el objetivo de capturar el movimiento y luego enviar a la máquina. En este sentido podemos encontrar propuestas basadas en guantes [16] o chalecos que por medio de diferentes sensores, como sistemas inerciales o “encoders” de posición, miden el movimiento de las diferentes falanges o extremidades.

Más interesante para nuestro propósito son los trabajos llevados a cabo basados en aquellas técnicas no dependientes de sistemas hardware adicionales que los operadores de rampa deban llevar encima, más allá de los elementos pasivos que ya utilizan. En este sentido, debemos mirar hacia las técnicas basadas en “visión por ordenador” al basarse estas en información no colaborativa que el sistema debe capturar y analizar de manera independiente del emisor.

En este sentido encontramos una vasta colección de artículos que recogen numerosos enfoques a lo largo de los años basados en diversas técnicas de visión por ordenador, comenzando por el enfoque de Paul Viola y Michael Jones [17] basado en la función de “Ondícula de Haar” para la identificación de objetos basados en la detección en cascada de características simples. Posteriormente, le siguió el trabajo presentado por Dalal y Trigs [18] basado en el empleo de “Histogramas de gradientes orientados - HOG”. Ambas técnicas fueron revolucionarias y todavía se usan en diferentes aplicaciones debido a que son poco demandantes computacionalmente y son fácilmente accesibles, pero sin embargo presentan algunas desventajas de base, como la cantidad de falsos positivos o la incapacidad para detectar humanos en diferentes posturas si éstos no están perpendiculares a la cámara.

Enfoques más modernos se basan nuevamente, en el empleo de redes convolucionales profundas. Desde que en 2012 AlexNet [19] mostrase el potencial de las “CNN” para la clasificación de imágenes, son numerosas las aplicaciones en las que se han usado. De especial interés para el reconocimiento de gestos y posturas son las “Convolutional Pose Machines - CPM” [20] un conjunto de redes pre-entrenadas que permiten la identificación de las articulaciones principales del cuerpo, o zonas específicas como manos, pies o incluso rasgos faciales, mediante una serie de coordenadas o “punto singulares” que conforman un “esqueleto artificial”.

En la actualidad existen varias implementaciones basadas en diferentes arquitecturas ya entrenadas con distintos modelos que permiten la detección de múltiples personas en una misma imagen. El problema del reconocimiento de posturas se puede subdividir en dos problemas más pequeños: segmentación y reconocimiento de puntos singulares [21]. En este sentido

podemos hacer una distinción entre las diferentes implementaciones atendiendo al tipo de arquitectura. Si siguen un enfoque en dos pasos, podemos hablar de modelos “de arriba hacia abajo” [22], en las que primero se ejecuta una segmentación para identificar a todas las personas en la imagen para luego extraer las posturas, o por el contrario, siguen una arquitectura “de abajo arriba” [23] en las que primero se identifican los puntos singulares de la imagen y posteriormente se agrupan para formar “esqueletos” con sentido, dando como resultado la segmentación de múltiples personas.

Entre los diferentes modelos destaca el desarrollado por el laboratorio de “Percepción Computacional” del la universidad norteamericana de “Carnegie Mellon”, que ganó la competición “COCO Keypoint Challenge” en el año 2016. Los autores del trabajo [24] diseñaron y entrenaron una arquitectura basada en varias redes profundas que consta de dos etapas. La primera consta de 10 capas identifica un mapa de características de la imagen analizada. La segunda etapa, más compleja, consta de dos redes convolucionales en paralelo e infiere un mapa de confianza que codifica el grado de asociación entre dichos puntos singulares para formar los vectores que conformaran el esqueleto

Al igual que cualquier otro problema que se aborde mediante una arquitectura basada en redes profundas los resultados del modelo pre-entrenado dependerán en gran medida del conjunto de entrenamiento utilizado. Afortunadamente, debido al gran interés científico e industrial en este campo en los últimos años han surgido numerosos datasets de gran calidad entre los que cabe destacar “*Common Objects in Context - COCO*” [25].

Finalmente, cabe mencionar que no se han encontrado demasiadas referencias académicas que aborden el objetivo específico de este trabajo: el reconocimiento de las “señales de rampa”. Destaca las contribuciones realizadas por los dos siguientes trabajos [11] y [26]. Ambos son artículos relativamente antiguos y su estrategia para abordar el problema se basaban en la captura por medio de una cámara RGB y clasificación posterior de los gestos mediante “Método de subsecuencia común más larga” [11] y la “Transformada de Radon” [25].

También encontramos una publicación científica del MIT [2] financiada por fondos de la Marina de EEUU que aborda el reconocimiento de las señales utilizadas en los portaaviones de la OTAN. El enfoque se basa en sensores de profundidad y el uso de la técnica “Motion History Images - MHI”. En este trabajo destaca además, el uso de dos sistemas para clasificar gestos tanto de la mano, como del cuerpo entero.

Sin embargo, como ha quedado de manifiesto si que existen numerosos trabajos y herramientas disponibles para apoyar el desarrollo de este trabajo basados en la interacción “hombre-máquina”, la visión artificial, el reconocimiento de gestos y las técnicas de clasificación supervisadas.

III. OBJETIVOS Y METODOLOGÍA

El objetivo general será el de disponer de un conjunto formado por una cámara estándar RGB y una unidad embebida de procesamiento de tamaño reducido y capacidad computacional limitada que pudiera ser instalado en vehículos aéreos no tripulados, que permita identificar y clasificar con un porcentaje de acierto de más del 80% al menos 5 gestos en tiempo real, de los 20 gestos reconocidos por la OACI.



Fig. 2. Este proyecto propone que de los 20 gestos reconocidos por OACI solo se reconozcan 5 categorías, incluyendo una designada como “no-gesto”.

Se abordará el problema desde una perspectiva exclusivamente visual, no dependiendo por tanto de sensores adicionales presentes en la propia cámara (e.g. sensor de profundidad) o en los miembros del personal de tierra (e.g. “wearables”).

Para conseguir el alcance identificado, será necesario alcanzar un nivel de desarrollo satisfactorio en los siguientes campos, que conforman los objetivos específicos del proyecto:

1. Explorar el estado del arte sobre el reconocimiento de gestos y aplicarlo al dominio específico del problema abordado.
2. Identificar el conjunto de hardware mínimo necesario que cumpliendo los requisitos identificados, posibilite la ejecución del software generado en el proyecto y a su vez permitiese ser instalado en una plataforma aérea de pequeño tamaño.
3. Diseñar una solución que se ejecute en tiempo real, clasificando las posturas que captura la cámara de video con una cadencia no superior a los 2 segundos de retraso, en alguna de las 5 categorías identificadas.
4. Generar un Dataset propio de entrenamiento con los gestos a identificar debidamente etiquetados. Este deberá estar compuesto al menos por datos extraídos de dos hombres y una mujer en edad adulta, que ejecutarán cada uno de los 4 categorías de gesto al menos 100 veces.
5. Cuantificar y evaluar la adecuación a los requisitos identificados y el rendimiento del sistema, debiendo tener este un porcentaje de acierto en la clasificación superior al 80%.

Atendiendo a la metodología, para el presente proyecto se ha planteado un desarrollo siguiendo un modelo “en Espiral”, en las que se han repetido fases consecutivas, e incrementales en dificultad, de: planificación, ejecución y evaluación de los resultados. La ventaja que proporciona esta metodología de trabajo es que es un modelo evolutivo con una concepción del proyecto de tipo experimental “de abajo hacia arriba” lo que permite iterar en ciclos crecientes en incertidumbre, generando entregables en cada iteración. Esto ayuda a identificar riesgos desconocidos antes de tiempo, y enfrentar los desarrollos más complejos como una colección de experiencias previas más simples y acotadas, con la seguridad de que antes de iniciar un desarrollo más complejo, ya se dispone al menos de las evidencias necesarias para justificar el tiempo y trabajo invertido.

La arquitectura de alto nivel de la solución quedaría definida por los siguientes bloques funcionales:

1. *Captura de video.* Engloba las tareas de captura de video a una tasa de refresco y con una calidad tal que permita los posteriores análisis. Quedarían aquí incluidos los trabajos de pre-procesamiento de la imagen iniciales. Cabe destacar que si bien se utiliza una cámara de video, el proceso se realizará de manera discreta sobre las imágenes extraídas como “frames” del propio video.
2. *Identificación en tiempo real de “señalero”.* Busca diferenciar este individuo respecto al fondo (otros objetos, personas..).
3. *Aplicación de “Convolutional Pose Machine - CPM”.*

Mediante la aplicación de una red neuronal pre-entrenada se llevará a cabo la identificación de las estructuras articulares parametrizadas del cuerpo humano. Con ello transformamos el conjunto de “features” del dominio de los pixels-imágenes al de vectores espaciales, simplificando los cálculos posteriores.

4. *Clasificación de los gestos mediante técnicas de aprendizaje supervisado.* Englobamos aquí las funciones finales que permitirán clasificar el gesto capturado por la cámara, y traducido mediante “CPM” en coordenadas, en una de las categorías previamente definidas.

Uno de los puntos potencialmente más conflictivos, es la elaboración de un “conjunto de entrenamiento” de los gestos que queramos identificar para poder llevar a cabo el paso de la clasificación, debido a la no existencia de dataset preparados para tal fin. Para solventar este problema, se propone el desarrollo de un programa complementario y más sencillo que el anterior que permita las labores de captura y etiquetado de las muestras de entrenamiento. Dicha aplicación permitirá la captura en video de varios individuos de diferentes estaturas y complexión, repitiendo varias veces cada gesto, en escenarios de iluminación no similares. La información capturada se procesará mediante “CPM” y se almacenará directamente las coordenadas de las articulaciones. El asistente será el encargado de etiquetar mediante un acceso rápido de la propia aplicación cada una de las posturas que se guardan.

IV. CONTRIBUCIÓN

Tal y como se ha venido presentando, el “setup” de pruebas que usaremos en este proyecto consta de 3 elementos principales:

- *Cámara RGB:* Estará montada en un trípode enfocando hacia el sujeto. Se conectará por USB al ordenador embebido.
- *Ordenador Embebido:* Ejecutará los diferentes softwares empleados, tanto para la generación del dataset como para la demostración en tiempo real. Recibirá el video digital desde la cámara por USB, y mostrará información visual en el monitor por medio de un puerto HDMI o por una acceso SSH mediante otro dispositivo.
- *Monitor/PC externo:* Este elemento es auxiliar y solo tiene sentido durante las pruebas de validación y.

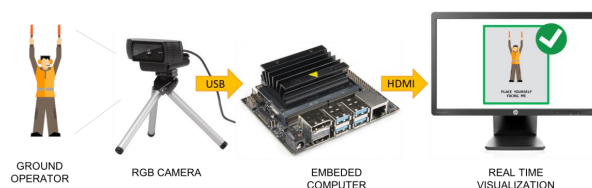


Fig. 3. Este proyecto propone que de los 20 gestos reconocidos por OACI solo se reconozcan 5 categorías, incluyendo una designada como “no-gesto”.

Este mismo montaje se utilizará para dos propósitos diferentes durante el proyecto: generar el dataset etiquetado primero, y el reconocimiento de gestos en tiempo real después.

Atendiendo a la arquitectura software del programa en su operación normal, podemos identificar 6 funciones iterativas claramente diferenciadas con objetivos específicos cada una, pero cuyas entradas/salidas estarán vinculadas según se muestra en el diagrama de flujo de la figura 4.

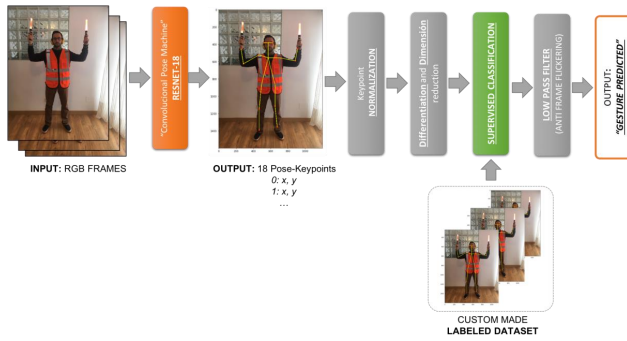


Fig. 4. Diagrama de flujo con los bloques funcionales principales de la arquitectura software propuesta.

1. *Captura de imagen*: Esta función engloba las tareas de captura de video. Quedarían aquí incluidos los trabajos de pre-procesamiento de la imagen iniciales.

2. *“Convolutional Pose Machine – ResNET-18”*: Mediante la aplicación de una red neuronal pre-entrenada se llevará a cabo identificación de las estructuras articulares parametrizadas del cuerpo humano. La red escogida ha sido ResNET-18 [27] una efectiva red neuronal de 18 capas de profundidad que sigue una novedosa arquitectura de tipo “Residual”.

3. *Normalización, Diferenciación y Reducción de la dimensionalidad*: Preparación de los datos obtenidos para la extracción de características.

4. *Clasificación*: Englobamos aquí las funciones finales que permitirán clasificar el gesto capturado por la cámara, y traducido mediante “CPM” en coordenadas, en una de las categorías previamente definidas. Se han evaluado dos modelos diferentes, uno basado en un ensemble de árboles de decisión y otro en una red artificial multicapa densa.

5. *Representación de la información*: La compondrían todas las funcionalidades para mostrar la información en pantalla.

Para las fases de generación de dataset y validación del sistema se ha desarrollado una aplicación gráfica de usuario, que permite el acceso rápido a las diferentes funcionalidades. Está dividida en 3 secciones atendiendo a su funcionalidad: área de ejecución en directo, área de predicción y área de dataset.

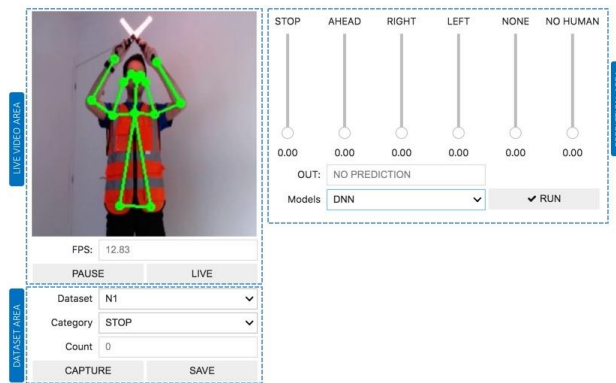


Fig. 5. Aplicación software desarrollada en el proyecto para la generación del dataset y validación de los modelos en tiempo real.

V. EVALUACIÓN Y RESULTADOS

El dataset generado en este proyecto está compuesto por 4857 ejemplos etiquetados. Han participado en su elaboración 2 varones y una mujer en edad adulta. Todos los sujetos han

repetido al menos 300 veces cada uno de los 5 gestos a clasificar. La información recopilada han sido las coordenadas de las articulaciones (18 puntos característicos, con dos coordenadas por punto), preservando en todo momento la privacidad de los voluntarios.

Una vez que los datos han sido procesado para su explotación, siguiendo los mismos procesos a los que se les someterá durante la etapa de inferencia, se procede a entrenar los dos modelos seleccionados.

Como el objetivo de la presente sección es la de comparar diferentes modelos de clasificación para nuestro conjunto de datos, es importante escoger las métricas que emplearemos para ello. Estos indicadores deben ser lo más completos posibles, y reflejar la capacidad del modelo tanto para predecir con precisión como para distinguir entre errores de tipo falso positivo y falso negativo. En el caso de los algoritmos de clasificación, no hemos encontrado mejor herramienta que la propia “Matriz de Confusión” para llevar a cabo este análisis, ya que además de permitirnos mostrar en un tabla la relación entre “predicción” y “realidad”, nos permite calcular con facilidad otras muchas métricas. De todas estas, basaremos nuestro análisis en dos de ellas: “Accuracy” y “F-Score”.

○ *Random Forest*:

Este método de tipo “bagging”, se basa en un ensemble de un gran número de árboles de decisión y fue presentado por Leo Breiman y Adele Cutler en el año 2001 [28]. Desde entonces, ha sido ampliamente utilizado y validado en muy diversas aplicaciones, debido a su capacidad predictora, facilidad de entrenamiento e implementación. Al combinar varios árboles de decisión y obtener la predicción final como la más votada de todas las clases predichas por cada árbol individual conseguimos un modelo con mejor capacidad de generalización y predicción, incluso con conjuntos de datos no demasiado grandes.

Como en cualquier modelo de aprendizaje supervisado, la elección correcta de hiperparámetros juega un papel fundamental. Para analizar el impacto de los más destacados se llevaron a cabo diferentes experiencias variando el número de árboles y el número de variables por árbol.

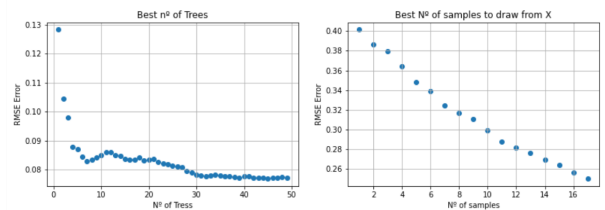


Fig. 6. Estimación experimental de hiperparámetros para modelo “Random Forest”.

Una vez analizada la combinación más adecuada de hiperparámetros, se procede al entrenamiento del modelo y a su evaluación. Se obtiene unos resultados excelentes, con un ratio de éxito del 98%.

○ *Red Neuronal Artificial*:

Las arquitecturas basadas en redes neuronales artificiales nos permiten, gracias a sus excelentes propiedades como aproximadores universales, modelar una relación entre el conjunto de variables de entrada y las señales de salida, a pesar del ruido o complejidad del patrón presente.

La topología empleada es una red de dos capas ocultas, con 128 neuronas cada una y función de activación “ReLU”. Entre capas se han intercalado sendas funciones de regularización

basadas en “Dropout” al 50%. La capa de salida tiene una neurona por cada una de las 5 categorías a clasificar, con función de activación “Softmax”. Como optimizador, tras varias iteraciones, se ha optado por “Adam”. El entrenamiento de los 21893 parámetros de la red se ha hecho en lotes de 32 muestras.

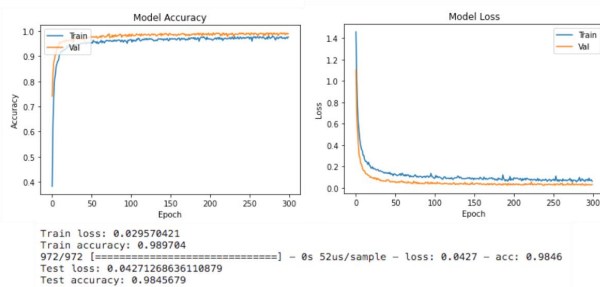


Fig. 7. Métricas de “precisión” y “función de error” obtenidas durante la fase de entrenamiento del modelo basado en ANN multicapa densa.

Tras algunas iteraciones se observa que no tiene utilidad ampliar el número de iteraciones en la fase de entrenamiento a más de 300 épocas. A partir de este momento, podemos considerar que la función de error se ha estabilizado entorno a su valor mínimo, habiendo alcanzado una “ratio de éxito” del 98% sobre el conjunto de prueba, que no había visto la red con anterioridad.

Evaluación en entorno real.

A continuación, se han puesto a prueba ambos en un entorno de inferencia en tiempo real, ejecutados sobre el ordenador embebido escogido, con datos no vistos con anterioridad. Si bien ambos modelos han funcionado correctamente, el modelo basado en ANN sobresale respecto al otro, dando una respuesta menos ruidosa y con menor incertidumbre en la predicción. Sin embargo, es justo destacar que para este modelo fue necesario ejecutar una acción intermedia para su optimización sobre GPU. Sin este paso, el modelo no optimizado era de media tres veces más lento que su homólogo basado en “Random Forest”.

Evaluación frente a modelo CNN.

Por último, y aunque la arquitectura expuesta ha dado excelente resultados, se ha evaluado otra solución diferente, basada esta vez en un red convolucional clásica, pre entrenada sobre un dataset de gran tamaño, a la cual se le ha sustituido la última capa, para añadir el clasificador de nuestro propio problema. Esto permite entender mejor el potencial del modelo defendido en este trabajo.

A pesar de partir de una red en la que la mayoría de parámetros ya han sido calculados de partida, es necesario generar un nuevo conjunto de datos que permita entrenar dicha nueva capa clasificadora. En este sentido, además de la generación de un nuevo dataset, se han empleado técnicas de aumentación de datos que permiten modificar ligeramente las muestras que verá la red, con el objetivo de introducir cierta variabilidad y mejorar la tendencia al sobre ajuste del modelo.

Es importante recalcar que, a diferencia de la arquitectura anterior en la que había dos modelos en serie con roles claramente diferenciados, en esta nueva arquitectura se presenta una única red cuyo parámetro de entrada es directamente la matriz de píxeles que componen cada imagen capturada por la cámara RGB, y su salida la probabilidad de pertenencia a cada una de las clases.

Tras implementar y entrenar el nuevo modelo, se consiguen buenos resultados en la fase de evaluación del sistema en tiempo real. Sin embargo, debido al reducido número de imágenes empleadas en la fase de entrenamiento, el modelo está muy sobre

ajustado y no es capaz de generalizar ante nuevos datos, no comportándose de manera adecuada cuando el operador o el fondo distan mucho de las imágenes de entrenamiento.

A pesar de que los resultados obtenidos no son satisfactorios, se identifica un claro caso de uso en el que un modelo como el de la segunda arquitectura propuesta, sería de gran utilidad: la operación nocturna. Se pone a prueba esta idea, generando un tercer dataset con imágenes con poca luminosidad en las que el operario emplea las linternas características de la profesión para señalar los gestos en la oscuridad.

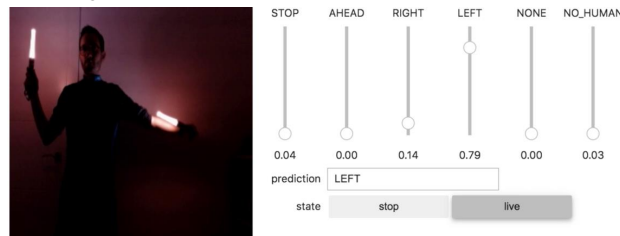


Fig. 8. Captura de las pruebas de validación del segundo modelo propuesto basado en arquitectura CNN para la identificación de las señales en operaciones de muy baja luminosidad.

Tras el entrenamiento se confirma la hipótesis evaluando el modelo en tiempo real: al ser la imagen eminentemente oscura, las características del fondo o el operario son poco informativas, y la red se centra en extraer características de la posición que ocupan las linternas en cada gesto, haciendo este modelo muy útil para reconocer gestos en los que debido a la baja intensidad de luz, el modelo basado en “CPM” no pueda llevar a cabo la segmentación.

VI. DISCUSIÓN

Este trabajo abordaba si sería posible el desarrollo de un sistema tal que, instalado en sistemas autónomos, permitiese la identificación en tiempo real de las “señales de rampa” que el personal de tierra del aeropuerto lleva a cabo. Específicamente se planteaba la hipótesis de poder utilizar para tal fin, técnicas y herramientas de procesamiento de imagen y aprendizaje supervisado, que permitiesen conseguir una solución eficiente, robusta y escalable.

Tras un pormenorizado análisis del dominio del problema y del estado del arte, se ha propuesto una plan de trabajo basado en unos componentes hardware, la elaboración de un conjunto de datos de entrenamiento y varias arquitecturas de modelo posible.

En este sentido se ha presentado dos arquitecturas. La primera, se basaba en la concatenación en serie de un primer modelo “CPM” seguido por un segundo modelo de clasificación pura. Sobre esta topología se han evaluado dos modelos diferentes de clasificador, uno basado “Random Forest” y otro en “ANN”, siendo este segundo el que mejores resultados ha dado en el entorno de experimentación real.

La segunda arquitectura propuesta se basaba en una única red CNN para la clasificación del gesto directamente desde el dominio de las imágenes. Aunque la primera arquitectura ha superado a esta segunda propuesta en robustez y escalabilidad, se ha identificado y validado como una potencial solución para la identificación de gestos en ambientes de escasa luminosidad como la operación nocturna, donde la primera arquitectura propuesta tiene mayores limitaciones para segmentar al operador.

Por lo tanto es posible, y en este proyecto se ha llevado a cabo un validador tecnológico que así lo demuestra, el empleo de técnicas de inteligencia artificial y visión por ordenador para la correcta identificación y clasificación de las señales de rampa llevadas a cabo por el personal de tierra.

VII. CONCLUSIONES

Aunque se han conseguido los objetivos del trabajo, si se desea convertir el validador tecnológico actual en una solución industrial fiable, debemos abordar nuevos retos técnicos en aras de una mejora en la robustez y sobretodo en la escalabilidad del sistema, mejorando tanto el dataset como las herramientas utilizadas.

Una clara línea de mejora sería el desarrollo de una solución que combinase e integrase ambas arquitecturas presentadas. Esto dotaría al sistema final de la capacidad de poder trabajar en diferentes situaciones, a pesar de que la iluminación no sea la adecuada. Aunque este desarrollo merece una reflexión más sosegada, se podría pensar en una primera capa de procesamiento que valorase la cantidad de píxeles oscuros presentes en cada imagen. A medida que este valor va aumentando se podrían ponderar con una figura de mérito las predicciones de ambos modelos, dando progresivamente más peso al modelo basado en la clasificación puras de.

REFERENCIAS

- [1] ICAO. (2005). *Annex 2 - Rules of the Air - Tenth Edition* (Issue November). <http://www.icao.int>
- [2] Song, Y., Demirdjian, D., & Davis, R. (2011). Tracking body and hands for gesture recognition: NATOPS aircraft handling signals database. *2011 IEEE International Conference on Automatic Face and Gesture Recognition and Workshops, FG 2011*, 500–506. <https://doi.org/10.1109/FG.2011.5771448>
- [3] Civil Aviation Authority (CAA). (1996). Visual aids handbook. *Aids*, 10(6), 690–691. <https://doi.org/10.1097/00002030-199606000-00024>
- [4] Castillo, J. C., Alonso-Martín, F., Cáceres-Domínguez, D., Malfaz, M., & Salichs, M. A. (2019). The Influence of Speed and Position in Dynamic Gesture Recognition for Human-Robot Interaction. *Journal of Sensors*, 2019. <https://doi.org/10.1155/2019/7060491>
- [5] Martin, J. B., & Moutarde, F. (2019). Real-Time Gestural Control of Robot Manipulator Through Deep Learning Human-Pose Inference. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 11754 LNCS, 565–572. https://doi.org/10.1007/978-3-030-34995-0_51
- [6] Molchanov, P., Yang, X., Gupta, S., Kim, K., Tyree, S., & Kautz, J. (2016). Online Detection and Classification of Dynamic Hand Gestures with Recurrent 3D Convolutional Neural Networks. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2016-Decem*, 4207–4215. <https://doi.org/10.1109/CVPR.2016.456>
- [7] Demarco, K. J., West, M. E., & Howard, A. M. (2014). Underwater human-robot communication: A case study with human divers. *Conference Proceedings - IEEE International Conference on Systems, Man and Cybernetics, 2014-Janua*(January), 3738–3743. <https://doi.org/10.1109/smc.2014.6974512>
- [8] Zaman Khan, R. (2012). Hand Gesture Recognition: A Literature Review. *International Journal of Artificial Intelligence & Applications*, 3(4), 161–174. <https://doi.org/10.5121/ijai.2012.3412>
- [9] Ribó, A., Warchol, D., & Oszust, M. prz edu pl. (2016). An approach to gesture recognition with skeletal data using dynamic time warping and nearest neighbour classifier. *International Journal of Intelligent Systems and Applications*, 8(6), 1–8. <https://doi.org/10.5815/ijisa.2016.06.01>
- [10] Kapuscinski, T., Oszust, M., Wysocki, M., & Warchol, D. (2015). Recognition of hand gestures observed by depth cameras. *International Journal of Advanced Robotic Systems*, 12. <https://doi.org/10.5772/60091>
- [11] Choi, C., Ahn, J. H., & Byun, H. (2008). Visual recognition of aircraft marshalling signals using gesture phase analysis. *IEEE Intelligent Vehicles Symposium, Proceedings*, 853–858. <https://doi.org/10.1109/IVS.2008.4621186>
- [12] Raheja, J. L., Minhas, M., Prashanth, D., Shah, T., & Chaudhary, A. (2015). Robust gesture recognition using Kinect: A comparison between DTW and HMM. *Optik*. <https://doi.org/10.1016/j.ijleo.2015.02.043>
- [13] Donahue, J., Hendricks, L. A., Rohrbach, M., Venugopalan, S., Guadarrama, S., Saenko, K., & Darrell, T. (2017). Long-Term Recurrent Convolutional Networks for Visual Recognition and Description. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(4), 677–691. <https://doi.org/10.1109/TPAMI.2016.2599174>
- [14] Zhou, B., Andonian, A., Oliva, A., & Torralba, A. (2018). Temporal Relational Reasoning in Videos. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 11205 LNCS, 831–846. https://doi.org/10.1007/978-3-030-01246-5_49
- [15] Hara, K., Kataoka, H., & Satoh, Y. (2018). Can Spatiotemporal 3D CNNs Retrace the History of 2D CNNs and ImageNet? *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 6546–6555. <https://doi.org/10.1109/CVPR.2018.00685>
- [16] Kim, J. H., Thang, N. D., & Kim, T. S. (2009). 3-D hand motion tracking and gesture recognition using a data glove. *IEEE International Symposium on Industrial Electronics, ISIE*, 1013–1018. <https://doi.org/10.1109/ISIE.2009.5221998>
- [17] Viola, P., & Jones, M. (2001). Rapid object detection using a boosted cascade of simple features. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. <https://doi.org/10.1109/cvpr.2001.990517>
- [18] Dalal, N., & Triggs, B. (2005). Histograms of oriented gradients for human detection. *Proceedings - 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR 2005*. <https://doi.org/10.1109/CVPR.2005.177>
- [19] Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). ImageNet classification with deep convolutional neural networks. *Advances in Neural Information Processing Systems*.
- [20] Wei, S. E., Ramakrishna, V., Kanade, T., & Sheikh, Y. (2016). Convolutional pose machines. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2016-Decem*, 4724–4732. <https://doi.org/10.1109/CVPR.2016.511>
- [21] Schneider, P., Memmesheimer, R., Kramer, I., & Paulus, D. (2019). Gesture Recognition in RGB Videos Using Human Body Keypoints and Dynamic Time Warping. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 11531 LNAI, 281–293. https://doi.org/10.1007/978-3-030-35699-6_22
- [22] Fang, H. S., Xie, S., Tai, Y. W., & Lu, C. (2017). RMPE: Regional Multi-person Pose Estimation. *Proceedings of the IEEE International Conference on Computer Vision*. <https://doi.org/10.1109/ICCV.2017.256>
- [23] Cao, Z., Simon, T., Wei, S. E., & Sheikh, Y. (2017). Realtime multi-person 2D pose estimation using part affinity fields. *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, 2017-Janua*, 1302–1310. <https://doi.org/10.1109/CVPR.2017.143>
- [24] Cao, Z., Hidalgo Martinez, G., Simon, T., Wei, S.-E., & Sheikh, Y. A. (2019). OpenPose: Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. <https://doi.org/10.1109/tpami.2019.2929257>
- [25] Lin, T. Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., & Zitnick, C. L. (2014). Microsoft COCO: Common objects in context. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 8693 LNCS(PART 5), 740–755. https://doi.org/10.1007/978-3-319-10602-1_48
- [26] Singh, M., Mandal, M., & Basu, A. (2005). Visual gesture recognition for ground air traffic control using the radon transform. *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS*, 2850–2855. <https://doi.org/10.1109/IROS.2005.1545408>
- [27] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2016-Decem*, 770–778. <https://doi.org/10.1109/CVPR.2016.90>
- [28] Breiman, L. (2001). Random forests. *Random Forests*, 1–122. <https://doi.org/10.1201/9780367816377-11>