UNIVERSIDAD DE MÁLAGA

ESCUELA DE INGENIERÍAS INDUSTRIALES

Departamento: Ingeniería de sistemas y automática
Área de conocimiento: Ingeniería de sistemas y automática

PROYECTO/TRABAJO FIN DE MÁSTER

RECONOCIMIENTO DE HERRAMIENTAS EN VÍDEOS DE ROBÓTICA QUIRÚRGICA Y EVALUACIÓN AUTOMÁTICA DE LA TAREA

MÁSTER EN INGENIERÍA INDUSTRIAL

MÁLAGA, 30 de Octubre de 2023

AUTOR: José Cabrera Villa

TUTORA: Irene Rivas Blanco

CO-TUTORA: Mª Carmen López Casado

RESUMEN

Este TFM se enmarca dentro del grupo de investigación de robótica médica de la Universidad de Málaga. El objetivo general de este trabajo es la evaluación automática en vídeos de robótica quirúrgica. En concreto, se analizarán vídeos pertenecientes a una base de datos de maniobras quirúrgicas realizadas con la plataforma da Vinci Research Kit (dVRK) en el marco de una colaboración entre el grupo de investigación de robótica médica de la Universidad de Málaga y el Instituto de Biorobótica de la Scuola Superiore Sant'Anna de la Universidad de Pisa [13]. Para ello, se utilizarán técnicas de Deep Learning para el reconocimiento de objetos en la imagen, y se empleará lógica proposicional como sistema de inferencia para el reconocimiento de las acciones básicas.

PALABRAS CLAVE:

Deep Learning, robótica médica, redes neuronales, segmentación por color, lógica proposicional.

ABSTRACT

This Master's Thesis is imagend within the medical robotics research group at the University of Malaga. The general objective of this work is the automatic evaluation of surgical robotics vídeos. Specifically, vídeos belonging to a database of surgical maneuvers performed with the da Vinci Research Kit robot will be analyzed, as part of a collaboration between the medical robotics research group at the University of Malaga and the BioRobotics Institute of Scuola Superiore Sant'Anna at the University of Pisa [13]. For this purpose, Deep Learning techniques will be used for object recognition in the image, and the use of first-order logic as an inference system for recognizing basic actions will be analyzed.

KEY WORDS:

Deep Learning, medical robotics, neural Networks, color segmentation, propositional logic.

Índice

Índice de figuras

Índice de tablas

Índice de pseudocódigos

1. Introducción

En el campo de la robótica quirúrgica, la capacidad de analizar las imágenes endoscópicas es esencial para poder entender el escenario quirúrgico y poder, por tanto, ser capaz de razonar y tomar decisiones durante una intervención. El reconocimiento de herramientas quirúrgicas es una de las tareas más estudiadas en el análisis de imágenes quirúrgicas, ya que los instrumentos representan el mecanismo de interacción entre el cirujano o cirujana y el escenario. Además, el tipo de instrumento utilizado en cada momento de la intervención proporcionan una información muy relevante acerca de la fase de la intervención y la maniobra que se está llevando a cabo. Por tanto,la capacidad de identificar y rastrear las herramientas utilizadas durante los procedimientos quirúrgicos es esencial para avanzar en el desarrollo de sistemas inteligentes que permitan tanto automatizar tareas como supervisar la intervención.

Para el desarrollo de este trabajo fin de máster se ha hecho uso de una base de datos de maniobras de entrenamiento realizadas con el sistema robótico da Vinci Research Kit (dVRK), una plataforma de investigación compuesta a partir de componentes del robot comercial da Vinci. El objetivo de este trabajo es implementar un sistemas de reconocimiento de objetos en vídeos de la base de datos, así como desarrollar un algoritmo de evaluación automática de una de las maniobras, empleando el tiempo y el número de errores cometidos como parámetros de evaluación.

1.1. Antecendentes y motivación

Hoy en día, la cirugía mínimamente invasiva, o cirugía laparoscópica, se ha convertido en la práctica habitual en numerosas intervenciones quirúrgicas, con grandes beneficios tanto estéticos como de recuperación para los pacientes. Sin embargo, este tipo de intervenciones introduce nuevos retos para el personal médico, que van desde la falta de visión directa y de sensación táctil hasta la limitación en el movimiento de las herramientas. La cirugía robótica ha permitido superar muchas de estas limitaciones así como mejorar las habilidades de los cirujanos gracias a instrumental más preciso y más intuitivo de manejar. A pesar de los elevados costes asociados, la demanda de la cirugía robótica sigue creciendo de forma exponencial, y los sistemas de salud tanto públicos como privados están apostando por integrar sistemas robóticos en sus procedimientos [1].

La eficiencia de estos sistemas frente a la cirugía convencional está ampliamente demostrada [2]. Sin embargo, a día de hoy este tipo de sistemas se limitan a replicar los movimientos que realiza un cirujano o cirujana en una consola de teleoperación. El uso de robots colaborativos, que puedan realizar cierto tipo de tareas de forma autónoma, permitiría reducir la carga mental del cirujano, permitiéndole concentrarse en las maniobras más complejas, mientras un asistente robótico realiza tareas auxiliares de apoyo. Además, en intervenciones convencionales, este tipo de tareas, que suelen ser muy repetitivas y tediosas, las realiza un cirujano/a auxiliar. Por tanto, la automatización de ciertas tareas permitiría liberar a este segundo cirujano/a, que podría dedicarse a otras tareas más complejas o estar disponible para otra intervención.

En los últimos años, las metodologías de automatización de tareas basadas en Deep Learning han crecido exponencialmente en todos los campos de la robótica, y la robótica quirúrgica no es una excepción. Este tipo de técnicas se fundamentan en el aprendizaje automático a partir un gran conjunto de datos, generalmente en forma de imágenes. En los últimos años, la comunidad científica ha realizado un gran esfuerzo por crear grandes bases de datos de intervenciones quirúrgicas, que puedan ser utilizadas para avanzar en el ámbito de la automatización en este tipo de entornos [3]. Así, existen numerosos trabajos que han desarrollado técnicas para el análisis automático de imágenes endoscópicas para clasificación y segmentación de instrumental quirúrgico [4]-[6], así como para la detección de estructuras anatómicas más complejas, como la segmentación del hígado [7],[9] o la detección de pólipos en imágenes colonoscópicas. Las técnicas de Deep Learning también se han utilizado para analizar las maniobras quirúrgicas con objeto de automatizar partes de los procedimientos quirúrgicos, y de crear sistemas capaces de supervisar los procedimientos y tomar cierto tipo de decisiones [10]-[12].

Este trabajo fin de máster se ha realizado en el grupo de investigación de Robótica Médica del Departamento de Ingeniería de Sistemas y Automática de la Universidad de Málaga. Investigadores de este grupo han desarrollado una base de datos de maniobras quirúrgicas realizadas con el robot dVRK, durante un proyecto realizado en colaboración con el Instituto de Biorobótica de la Scuola Superiore Sant'Anna de Pisa [13]. Esta base de datos está formada por un conjunto de vídeos de 3 maniobras básicas de entrenamiento.

1.2. Objetivos

El objetivo global de este trabajo es el reconocimiento de herramientas en vídeos de robótica quirúrgica y la evaluación automática de una tarea concreta. En particular, se analizarán vídeos pertenecientes a una base de datos de maniobras de entrenamiento realizadas con el sistema dVRK en el marco de una colaboración entre el grupo de investigación de robótica médica de la Universidad de Málaga y El Instituto de Biorobótica de la Scuola Superiore Sant'Anna de la Universidad de Pisa [13]. Para conseguir este objetivo, se plantean los siguientes objetivos específicos:

- Etiquetar la base de datos con la localización de las herramientas. Se etiquetarán las dos herramientas (izquierda y derecha), indicando su posición en cada imagen y distinguiendo una de otra, ya que realizan distintas acciones.
- Implementar una red neuronal para el reconocimiento de las herramientas. Esta red neuronal será capaz de identificar la posición de cada herramienta, así como qué herramienta es, en todos los vídeos del dataset.
- Evaluación automática de una de las tres tarea del dataset. En particular, se realizará la evaluación de la tarea "Post and Sleeve". Para ello será necesario detectar los objetos en la imagen. Además, se medirá el tiempo empleado en realizar la maniobra y el número de errores cometidos durante la tarea. Los datos obtenidos se utilizarán para evaluar la tarea.

.

1.3. Metodología

Para llevar a cabo los objetivos planteados anteriormente, se han realizado las siguientes fases de trabajo:

1. Revisión bibliográfica: Se llevará a cabo una revisión exhaustiva de la literatura existente sobre reconocimiento de herramientas quirúrgicas. Se analizarán los avances más recientes, los métodos utilizados y los desafíos encontrados en este campo.
2. Análisis de la base de datos: Se analizará la base de datos de la que se dispone para el entrenamiento y evaluación del modelo de reconocimiento. Esta base de datos contiene vídeos de herramientas quirúrgicas realizando 3 maniobras de entrenamiento básicas.
3. Etiquetado de la base de datos. Se ha etiquetado la base de datos para poder distinguir la herramienta izquierda de la derecha, ya que cada una desarrolla acciones diferentes en la maniobra. Por este motivo, es importante saber tanto la localización como que herramienta es.
4. Implementación del modelo de reconocimiento: Se implementará y entrenará un modelo de aprendizaje automático utilizando el conjunto de datos. Se explorarán diferentes arquitecturas de redes neuronales y se realizarán técnicas de procesamiento de imágenes para evaluar las tareas que realizan las herramientas mediante condiciones lógicas.
5. Evaluación del desempeño del sistema: Se realizarán pruebas y evaluaciones para medir la precisión y eficiencia del sistema propuesto. Se compararán los resultados obtenidos con los métodos existentes en la literatura para validar la eficacia del sistema propuesto.
6. Automatización de la evaluación de una de las tareas. Se automatizará la evaluación de una de las tareas. Cada tarea tiene una puntuación, esta puntuación se calcula teniendo en cuenta el tiempo total que el usuario ha necesitado para terminar la tarea, más los puntos de penalización por los errores cometidos durante el tarea.

1.4. Estructura de la memoria

Este trabajo de fin de máster se estructurará de la siguiente manera:

1. **Introducción**: En este capítulo se presentará el contexto del reconocimiento de herramientas quirúrgicas, los antecedentes y la relevancia del tema. También se describirán los objetivos del trabajo y la metodología utilizada.
2. **Estado del arte**: En este capítulo se revisará el estado actual de la investigación en el campo del reconocimiento de herramientas quirúrgicas. Se analizarán los avances más recientes, los enfoques y técnicas utilizadas, así como los desafíos y limitaciones existentes en este campo.

3. **Descripción de la base de datos**: En este capítulo se realizará una descripción de la base de datos utilizada, así como se explicará el modo de puntuación de las tres maniobras básicas que se incluyen en el dataset.

4. **Reconocimiento de herramientas quirúrgicas**: En este capítulo se describirá en detalle la implementación del sistema de reconocimiento de herramientas quirúrgicas. Se explicarán las técnicas y algoritmos utilizados, así como los aspectos técnicos y de programación relevantes. Además, se abordarán las decisiones de diseño y las consideraciones específicas para la implementación del sistema.

5. **Evaluación automática de la tarea *Post and sleeve***: En este capítulo se desarrollará como se ha realizado la evaluación automática de la tarea *Post and sleeve*. Además se comentaran los resultados obtenidos comparándolos con los proporcionados en el propio dataset.

6. **Conclusiones y líneas futuras**: En este capítulo se presentarán las conclusiones principales derivadas de este trabajo de fin de máster. Se discutirán los logros alcanzados en relación con los objetivos planteados y se resumirán los hallazgos más relevantes. Asimismo, se abordarán las limitaciones del sistema y se propondrán posibles mejoras y direcciones futuras para la investigación en este campo.

2. Estado del arte

2.1. Introducción

La cirugía mínimamente invasiva (CMI) ha experimentado un crecimiento significativo en las últimas décadas, revolucionando la práctica médica al ofrecer procedimientos quirúrgicos menos invasivos y más precisos. Este enfoque quirúrgico implica el uso de instrumentos quirúrgicos que se insertan a través de pequeñas incisiones, lo que a menudo limita la visión directa del cirujano sobre el sitio quirúrgico. Para abordar este desafío y mejorar la eficacia de la CMI, ha surgido un campo de investigación dedicado al reconocimiento de herramientas quirúrgicas y la deducción de la acción en tiempo real durante la cirugía. Este estado del arte se propone explorar los avances clave en esta área, destacando [3] y [14].

La robótica quirúrgica ha revolucionado la cirugía al ofrecer precisión, estabilidad y capacidades avanzadas que van más allá de las limitaciones de la cirugía tradicional. Sin embargo, para aprovechar al máximo estas tecnologías, es fundamental contar con datos precisos que permitan el análisis, la optimización y la mejora continua de los procedimientos quirúrgicos. Los conjuntos de datos han surgido como una herramienta esencial para avanzar en la robótica quirúrgica. En el artículo [13], se destaca la importancia de los avances en SDS para mejorar diversos aspectos de la cirugía, como el entrenamiento virtual, la evaluación de habilidades quirúrgicas y el aprendizaje de tareas complejas utilizando sistemas quirúrgicos robóticos. Varios estudios mencionados en el artículo [13], como [15], [16] y [17], demuestran cómo el uso de la ciencia de datos puede tener un impacto significativo en la mejora de la práctica quirúrgica y la atención al paciente.

La Ciencia de Datos Quirúrgicos (SDS, por sus siglas en inglés) está emergiendo como un nuevo dominio de conocimiento en el campo de la salud. En el ámbito de la cirugía, puede proporcionar numerosos avances en la capacitación virtual, la evaluación de las habilidades de los cirujano/as y el aprendizaje de tareas complejas a partir de sistemas robóticos quirúrgicos [15], así como en el ámbito del reconocimiento de gestos [16], [17]. La comprensión de la escena quirúrgica se ha convertido en una tarea esencial para el desarrollo de sistemas inteligentes capaces de colaborar con los cirujanos durante una intervención real [33]. El desarrollo de grandes conjuntos de datos relacionados con la ejecución de tareas quirúrgicas utilizando sistemas robóticos respaldaría estos avances, proporcionando información detallada sobre los movimientos del cirujano, tanto en términos de datos cinemáticos como dinámicos, así como grabaciones de vídeo.

2.2. Bases de datos públicas de robótica quirúrgica

La robótica quirúrgica ha emergido como un campo de la medicina que ha revolucionado la forma en que se realizan las intervenciones quirúrgicas. Uno de los elementos clave que ha permitido este avance es la disponibilidad de conjuntos de datos detallados y diversos que impulsan la investigación, el desarrollo y la mejora de los sistemas quirúrgicos robóticos. El uso de la ciencia de datos en el campo de la salud está emergiendo como un dominio de conocimiento crucial para impulsar avances significativos en la atención médica. En particular, el campo de la robótica quirúrgica se ha beneficiado enormemente de la aplicación de la ciencia de datos, lo que ha llevado a la aparición de un nuevo dominio conocido como Surgical Data Science (SDS).

Las principales utilidades de los Conjuntos de Datos en Robótica Quirúrgica son:

1. **Entrenamiento y Evaluación de Habilidades Quirúrgicas**: Uno de los usos más evidentes de los conjuntos de datos en robótica quirúrgica es en el entrenamiento y la evaluación de las habilidades quirúrgicas de los cirujanos. Estos datos permiten el desarrollo de simulaciones quirúrgicas realistas que ofrecen a los cirujanos en formación la oportunidad de practicar procedimientos en un entorno virtual antes de enfrentarse a cirugías reales. Además, la recopilación de datos en tiempo real durante procedimientos reales permite una evaluación objetiva de las habilidades quirúrgicas de los profesionales, lo que facilita la identificación de áreas de mejora y la personalización de programas de formación.

2. **Optimización de Procedimientos Quirúrgicos**: La robótica quirúrgica se beneficia enormemente de la optimización de procedimientos a través del análisis de datos. Los conjuntos de datos permiten a los investigadores y cirujanos identificar patrones y mejores prácticas en procedimientos específicos. Esto conduce a la refinación

de técnicas quirúrgicas, la reducción de tiempos de operación y la mejora general de los resultados para los pacientes.

3. **Investigación y Desarrollo de Algoritmos**: La ciencia de datos y la inteligencia artificial juegan un papel crucial en la automatización y mejora de la robótica quirúrgica. Los conjuntos de datos proporcionan la base para el desarrollo y la validación de algoritmos de visión por computadora, planificación de movimientos y control robótico. Estos algoritmos mejoran la precisión y la seguridad de los sistemas quirúrgicos, lo que resulta en un mejor rendimiento durante las cirugías.

4. **Evaluación de Nuevas Tecnologías y Dispositivos**: Los conjuntos de datos son herramientas esenciales para evaluar nuevas tecnologías y dispositivos en el campo de la robótica quirúrgica. Los investigadores pueden utilizar datos para comparar el rendimiento de diferentes sistemas robóticos, sensores y herramientas quirúrgicas. Esto contribuye a la toma de decisiones informadas sobre la adopción de nuevas tecnologías en entornos clínicos y mejora la atención al paciente.

Además, los conjuntos de datos públicos permiten comparar el rendimiento de diferentes algoritmos propuestos en la literatura. Rivas-Blanco et al. [3] proporciona una lista de 13 conjuntos de datos públicos disponibles en el ámbito quirúrgico. La mayoría de los conjuntos de datos incluyen datos de vídeo, pero solo dos de ellos incorporan datos cinemáticos, que proporcionan una gran cantidad de información útil para analizar métricas relacionadas con el movimiento de las herramientas. Los datos cinemáticos se registran a partir de un kit de investigación da Vinci (dVRK), una plataforma de investigación basada en el Sistema Quirúrgico da Vinci de primera generación (de Intuitive Surgical, Inc., Sunnyvale, CA). Esta plataforma cuenta con un paquete de software que proporciona datos cinemáticos y dinámicos de la Herramienta Maestra y los Manipuladores del Lado del Paciente.

Para respaldar estos avances, es fundamental contar con conjuntos de datos amplios y detallados que proporcionen información precisa sobre los movimientos quirúrgicos. Algunos de los conjuntos de datos más conocidos en este campo son:

1. **JIGSAWS (The JHU-ISI Gesture and Skill Assessment Working Set)**: El conjunto de datos JIGSAWS 2.1 a, se ha convertido en una referencia clave para el análisis de movimientos quirúrgicos y la evaluación de habilidades quirúrgicas. Se centra en tres tareas quirúrgicas realizadas por seis cirujanos utilizando sistemas robóticos como el dVRK. El dVRK, que se basa en el sistema quirúrgico da Vinci de primera generación. Ofrece datos cinemáticos y dinámicos, así como grabaciones de vídeo, lo que lo convierte en una plataforma ideal para recopilar información detallada sobre los movimientos quirúrgicos [18].

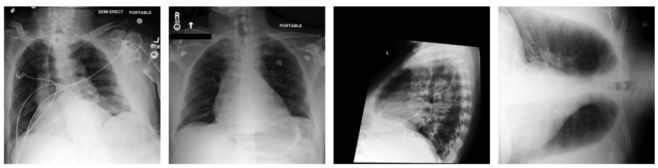
2. **ROSMA (Robotic Surgical Manuevers)**: A pesar de la utilidad del conjunto de datos JIGSAWS, existe una necesidad de conjuntos de datos adicionales que amplíen aún más el conocimiento en el campo de la robótica quirúrgica. El conjunto de datos ROSMA 2.1 c, es un esfuerzo más reciente que busca proporcionar una colección más amplia y diversa de datos relacionados con la robótica quirúrgica. Su objetivo principal es mejorar la comprensión de los movimientos quirúrgicos y facilitar la investigación en el campo. Es en este contexto que se presenta el conjunto de datos Robotic Surgical Manuevers (ROSMA) [16]. El objetivo principal de [16] es proporcionar un conjunto de datos más extenso y diverso relacionado con la robótica quirúrgica.

3. **MIMIC (Medical Information Mart for Intensive Care)**: Aunque se centra en la atención crítica más que en la cirugía, MIMIC 2.1 b, es un conjunto de datos valioso que contiene información detallada sobre pacientes en unidades de cuidados intensivos como se describe en [32]. Puede utilizarse para investigar procedimientos quirúrgicos de alto riesgo y evaluar el impacto de las intervenciones quirúrgicas en la recuperación de los pacientes, así como el tiempo de recuperación de los mismos como se puede ver en [31].

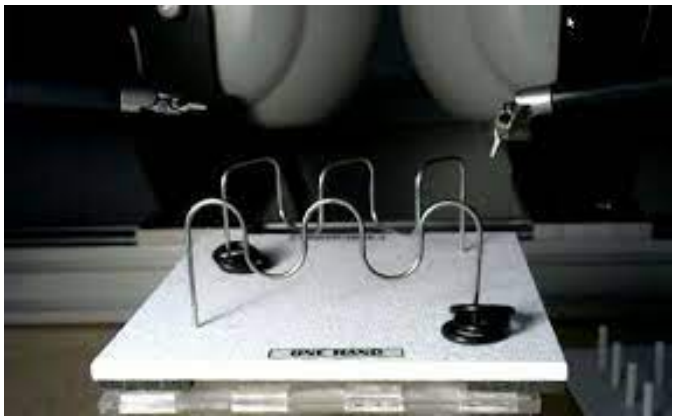
El conjunto de datos JIGSAWS 2.1, descrito en [18], es el conjunto de datos más conocido en robótica quirúrgica. Este conjunto de datos incluye datos cinemáticos de 76 dimensiones junto con datos de vídeo para 101 pruebas de tres tareas quirúrgicas elementales (suturar, hacer nudos y pasar la aguja), realizadas por 6 cirujanos utilizando el dVRK. Por otro lado, el conjunto de datos UCL dVRK [34] contiene 14 vídeos utilizando el dVRK en cinco tipos diferentes de tejido animal. Para cada imagen de vídeo, se produce una imagen asociada de las herramientas virtuales utilizando un simulador dVRK.

En [13] se presenta el conjunto de datos de Maniobras Quirúrgicas Robóticas (ROSMA), un conjunto de datos grande recopilado utilizando el dVRK, en colaboración entre la Universidad de Málaga (España) y el Instituto de Biorrobótica de la Scuola Superiore Sant'Anna (Italia), en el marco de un proyecto de TERRINet (Red Europea de

(a) JIGSAWS Dataset.



(b) MIMIC Dataset.



(c) ROSMA Dataset.

Figura 2.1: Distintas bases de datos.

Infraestructura de Investigación en Robótica). Este conjunto de datos contiene 36 variables cinemáticas, divididas en datos de 154 dimensiones, registrados a 50 Hz para 206 pruebas de tres tareas quirúrgicas de entrenamiento comunes. Estos datos se complementan con grabaciones de vídeo recopiladas a 15 imágenes por segundo con una resolución de 1024 x 768 píxeles. Además, se proporciona una evaluación de la tarea basada en el tiempo y errores específicos de la tarea, un archivo de datos de sincronización entre datos y vídeos, la matriz de transformación entre la cámara y los Manipuladores del Lado del Paciente, y un cuestionario con datos personales de los sujetos (género, edad, mano dominante) y experiencia previa en el uso de sistemas teleoperados y habilidades visuomotoras (deporte e instrumentos musicales). La simplicidad de las tareas grabadas facilita la realización de experimentos de laboratorio basados en estos datos. En resumen, las principales contribuciones de [13],[14] son:

1. Proporcionar un conjunto de datos grande de tareas quirúrgicas robóticas de acceso público, recopilado con el Kit de Investigación da Vinci, que incluye datos cinemáticos y de vídeo.

2. Completar el conjunto de datos con las matrices de proyección de la cámara que relacionan las coordenadas tridimensionales de las puntas de las herramientas con las coordenadas de imagen bidimensionales.

3. Para facilitar el uso de los datos, se proporciona un software MATLAB que ofrece la opción de reproducir los datos de los experimentos en temas de ROS (Sistema Operativo Robótico).

En conclusión, los conjuntos de datos desempeñan un papel esencial en la evolución de la robótica quirúrgica. Facilitan el entrenamiento, la evaluación, la investigación y el desarrollo de tecnologías avanzadas, lo que resulta en una atención médica más precisa, segura y eficaz. A medida que la ciencia de datos continúa avanzando, se espera que los conjuntos de datos en robótica quirúrgica sigan expandiéndose y contribuyendo a la mejora continua de la atención médica y los procedimientos quirúrgicos. La colaboración entre profesionales de la salud, ingenieros y científicos de datos seguirá siendo fundamental para impulsar este emocionante campo hacia el futuro.

2.3. Reconocimiento de herramientas

Como se meniona en [3] la cirugía mínimamente invasiva (CMI), o cirugía laparoscópica, se ha convertido en una práctica común en muchas intervenciones quirúrgicas con grandes beneficios para los pacientes. Sin embargo, introduce nuevos desafíos para los cirujanos, como la falta de visión directa, la sensación táctil y limitaciones en el movimiento de los instrumentos.

La cirugía asistida por robots (CAR) supera varios problemas de la CMI y mejora la eficiencia de los cirujanos con un movimiento más preciso e intuitivo de los instrumentos. El interés en los robots quirúrgicos es innegable si observamos los grandes esfuerzos económicos que las principales economías del mundo están haciendo para impulsar este mercado, que se espera que crezca a una tasa de crecimiento anual compuesta del 10.7 % durante el período de pronóstico de 2019 a 2029, alcanzando un mercado de 15.43 mil millones de dólares para 2029. En el ámbito académico, la comunidad científica también muestra un gran interés con miles de publicaciones en las últimas décadas y numerosos proyectos que están siendo financiados para avanzar en este campo.

Intuitive Surgical también está impulsando la investigación en robótica quirúrgica al apoyar a la comunidad científica con plataformas de investigación del Sistema Quirúrgico da Vinci, conocido como Kit de Investigación da Vinci (dVRK), y facilitando la cooperación entre diferentes grupos de investigación.

La eficiencia de los robots quirúrgicos como herramienta para mejorar las habilidades de los cirujanos se ha demostrado ampliamente. Sin embargo, en este momento estos sistemas no son capaces de proporcionar asistencia real al cirujano. Se limitan a replicar los movimientos realizados por el cirujano en una consola maestra en una plataforma esclava. Por lo tanto, los investigadores han dirigido sus esfuerzos al desarrollo de formas automáticas de asistencia para reducir la carga de trabajo de los cirujanos durante las intervenciones.
La capacidad de realizar tareas autónomas y tomar decisiones de forma autónoma en tiempo real requiere una comprensión profunda del entorno en el que el sistema está trabajando. Por lo tanto, reconocer qué elementos están en la escena e inferir lo que está ocurriendo en un momento particular durante una intervención es vital para avanzar en sistemas inteligentes para la CMI.

Los enfoques de modelado explícito para desarrollar técnicas de servomando visual en un escenario quirúrgico son ineficientes debido a la gran variabilidad entre personas, órganos y tejidos. En contraste, las técnicas de aprendizaje automático que aprenden modelos implícitos directamente a partir de datos brutos parecen ser muy adecuadas en estos escenarios dinámicos y complejos.

Además, la escena quirúrgica representa un gran desafío en las técnicas de percepción debido a la naturaleza dinámica y compleja del cuerpo humano, lo que genera un amplio campo de nuevas oportunidades de investigación. En [39] se presenta un interesante estudio de técnicas de Deep Learning en cirugía laparoscópica. El objetivo de su revisión es familiarizar a los clínicos con esta nueva técnica, por lo que se centraron en el valor clínico de los trabajos de informes.

El reconocimiento de herramientas quirúrgicas es un componente esencial para comprender lo que está sucediendo en el campo quirúrgico durante un procedimiento. La identificación precisa de las herramientas utilizadas por el cirujano es crucial para llevar a cabo un seguimiento adecuado de la cirugía y proporcionar retroalimentación en tiempo real. Además, es fundamental para la automatización de tareas quirúrgicas y la integración de sistemas de asistencia quirúrgica basados en visión por computadora como se puede ver en [3] y [14].

En los vídeos laparoscópicos, cada imagen suele contener más de un instrumento a la vez, por lo que la clasificación multi-etiqueta es más interesante que la clasificación binaria. En este tipo de algoritmos, cada instancia puede pertenecer a más de una clase. En [4] se presenta un método de clasificación multi-clase que combina dos modelos de redes neuronales convolucionales (CNN), VGGNet y GoogLeNet, para producir el resultado final. Cada red se entrena por separado y las predicciones de cada una se promedian para calcular la clasificación final. La principal limitación de este trabajo es que no considera la información temporal de los vídeos. Sin embargo, el contexto temporal es importante para distinguir las herramientas quirúrgicas y superar el problema de su alta similitud entre sí.

En [40] proponen incorporar información espacio-temporal en el problema de clasificación de herramientas utilizando una red LSTM profunda. En la primera etapa, se entrena una CNN para detectar la presencia de herramientas en cuadros individuales. Luego, las características aprendidas por la CNN se utilizan para aprender un modelo temporal utilizando una red LSTM, lo que proporciona una mayor precisión de la clasificación.

De manera similar, en [41] proponen monitorear el uso de herramientas durante la cirugía utilizando redes neuronales convolucionales y recurrentes. El marco propuesto consta de varias CNN que extraen características visuales de los vídeos y RNN para analizar la secuencia temporal a lo largo de toda la cirugía, basándose en las salidas de las CNN.
Con este enfoque, aumentaron el rendimiento del modelo a alrededor del 98 %. La dimensión temporal también se

considera en [42]. En este trabajo, los autores utilizan una Red de Convolución Gráfica (GCN) para aprender mejores características al considerar la relación entre imágenes de vídeo continuos. En [43] propusieron una arquitectura U-Net modificada para la segmentación semántica, con una alta puntuación de rendimiento. Sin embargo, este estudio solo considera tres herramientas diferentes en comparación con las siete clases de los trabajos anteriores.

En [6] presentaron en 2017 el primer enfoque que incorpora redes neuronales profundas (DNN) para la detección y localización de herramientas en cirugía asistida por robots. Aplicaron una Red de Propuesta de Región (RPN) junto con una red convolucional multimodal para la localización y una Fast R-CNN para la detección de objetos. En este trabajo, también introdujeron el conjunto de datos ATLAS Dione, el primer conjunto de datos público de vídeos de cirugía asistida por robot con anotaciones de herramientas.
En [44] y [45], los autores se centraron en la detección de la articulación de instrumentos robóticos. Modelaron cada herramienta como un conjunto de articulaciones y conexiones entre articulaciones (Figura 2.2). Luego, utilizaron una Red Neuronal Convolucional Totalmente Convolucional (FCNN) para detectar los pares de articulaciones, cuya salida se utiliza para estimar la posición de las herramientas.

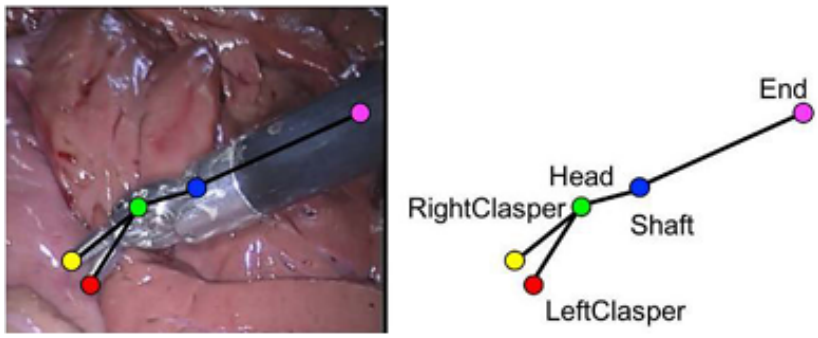


Figura 2.2: El esqueleto del instrumento *EndoWrist Needle Driver* se divide en 5 articulaciones y 4 conexiones.

El seguimiento en tiempo real de las herramientas quirúrgicas se aborda en [46]. En este trabajo, los autores utilizan una CNN con detector de segmentos de línea (LSD) para detectar las líneas de las herramientas y el contexto espacio-temporal (STC) para rastrear las herramientas imagen por imagen en tiempo real. En [47], se propone una CNN en cascada para reconocer y localizar herramientas quirúrgicas robóticas.

La red consta de una red en forma de reloj de arena, que produce mapas de calor del área de la punta de las herramientas, y una red VGG-16 modificada que realiza regresión de cuadros delimitadores en estos mapas de calor. Avanzando en este trabajo, en [48] proponen una CNN sin anclaje, modelando las herramientas quirúrgicas como un solo punto. Estos trabajos exhiben mejores resultados tanto en velocidad como en precisión. Yu et al. [39] se centraron en la detección de instrumentos quirúrgicos pequeños. Combinaron un mapa de atención creado a partir de características de alto nivel con características de bajo nivel para enriquecer la información semántica baja.
En [3] se destaca cómo las técnicas de Deep Learning (DL) Figura 2.3 han tenido un impacto significativo en la investigación de la CMI. La visión por ordenador desempeña un papel crucial en el reconocimiento de herramientas quirúrgicas, y las imágenes quirúrgicas se consideran una fuente de datos fundamental, ya que están disponibles en todo momento durante una intervención.

Una de las estrategias clave para el reconocimiento de herramientas es la clasificación de las mismas. Dada una imagen de la cirugía, un modelo de DL puede clasificar las herramientas presentes en la imagen. Este enfoque es útil para proporcionar una lista de herramientas utilizadas en un momento dado.
Recientemente, se han desarrollado modelos de DL altamente precisos capaces de clasificar una amplia variedad de herramientas quirúrgicas con alta precisión. Estos modelos utilizan redes neuronales convolucionales (CNN) y redes neuronales profundas para aprender características específicas de cada herramienta, lo que permite una clasificación precisa [3].

Otro enfoque en el reconocimiento de herramientas es la detección de las mismas. En lugar de clasificar, el modelo puede detectar y localizar las herramientas en la imagen. Esto implica dibujar cajas delimitadoras alrededor de cada herramienta identificada.

Figura 2.3: Métodos para el reconocimiento de herramientas en una imagen de cirujía: a) Clasificación, b) Detección y c) Segmentación.

La detección de herramientas quirúrgicas se ha vuelto especialmente importante en la automatización de tareas quirúrgicas y la asistencia robótica. Los modelos de DL han demostrado ser capaces de detectar instrumentos quirúrgicos incluso en condiciones difíciles, como imágenes borrosas o con sangre [3].

En tareas más avanzadas, se puede realizar la segmentación semántica de la imagen para identificar cada píxel correspondiente a una herramienta específica. La segmentación de herramientas es útil para proporcionar información detallada sobre la ubicación precisa de las herramientas en la imagen quirúrgica.
La segmentación se ha vuelto fundamental en la investigación de sistemas de asistencia quirúrgica basados en visión, ya que permite una comprensión más profunda de la interacción entre las herramientas y el tejido durante la cirugía. Esta información puede ser crucial para evitar lesiones en los tejidos circundantes y mejorar la precisión de los procedimientos [3].

Uno de los desafíos clave en el reconocimiento de herramientas es la disponibilidad de conjuntos de datos adecuados para el entrenamiento y la evaluación de algoritmos. En [3] se menciona la creación de conjuntos de datos específicos para esta tarea, como el conjunto de datos .EndoVis 2017 Tool Presence Detection Challenge", que contiene imágenes quirúrgicas anotadas con la presencia de herramientas quirúrgicas.
La evaluación de algoritmos de reconocimiento de herramientas se basa en métricas como la precisión, la sensibilidad y la puntuación F1. Estas métricas son fundamentales para medir el rendimiento de los modelos y garantizar su aplicabilidad en entornos clínicos [3].

2.4. Evaluación automática de maniobras

Una tarea importante en la formación médica es la evaluación de las habilidades quirúrgicas para calificar el desempeño de los aprendices y supervisar su desarrollo durante el proceso de entrenamiento. Esta evaluación suele realizarse manualmente por expertos, lo que no solo es muy laborioso, sino que también es subjetivo y carece de consistencia y fiabilidad. Para abordar estos problemas, muchos autores han abordado la tarea de la evaluación automática de habilidades a través del análisis descriptivo del movimiento de instrumentos, lo que requiere una alta ingeniería de características manuales o el uso de modelos predictivos como los Modelos Ocultos de Markov, logrando una alta precisión que oscila entre el 94.4 % y el 100 % [52], [53].

Sin embargo, estos métodos requieren una gran cantidad de tiempo y esfuerzo computacional para ajustar y modelar los parámetros. En contraste, los modelos de Deep Learning pueden procesar datos en bruto y realizar un aprendizaje automático de características para descubrir representaciones abstractas durante el proceso de formación. La Figura 2.4 muestra las características técnicas de los estudios que realizan la evaluación de habilidades quirúrgicas utilizando modelos de Deep Learning incluidos en esta revisión.

La mayoría de los trabajos encontrados en la literatura que realizan la segmentación de la evaluación de habilidades quirúrgicas mediante modelos de Deep Learning dividen los niveles de experiencia en tres categorías: principiante (N), intermedio (I) y experto (E). Por lo tanto, dada una tarea de rendimiento, los algoritmos de Deep Learning se entrenan para proporcionar la probabilidad de que los datos de entrada pertenezcan a una de estas clases. Este es el caso del trabajo desarrollado en [54]. Diseñaron una red neuronal convolucional unidimensional dedicada a la clasificación de habilidades quirúrgicas y lograron resultados muy competitivos con una precisión del 100 % en las tareas de sutura y

Ref.	Year	Method	DL model	Input data	Dataset	Results
[82]	2018	Level of expertise	CNN	Kinematic data	JIGSAWS	100% (acc)
[83]	2018	Level of expertise	CNN	Kinematic data	JIGSAWS	95.4% (acc)
[84]	2018	Level of expertise	SATR-DL	Kinematic data	JIGSAWS	96% (acc)
[85]	2019	Level of expertise	3D ConvNet	Images	JIGSAWS	95% (acc)
[86]	2019	Level of expertise	CNN+LSTM	Kinematic data	JIGSAWS	98.4% (acc)
[87]	2020	Level of expertise	CNN	Kinematic data	JIGSAWS	99.1% (acc)
[88]	2020	Detecting similar levels of expertise	SNN	Kinematic data	NPA	83.4% (acc)
[89]	2019	Pairwise ranking	LSTM	Kinematic data	JIGSAWS	75.1% (acc)
[15]	2018	Performance score (GOALS)	R-CNN (VGG16)	Images	m2cai16-tool-location	

* When more than one result is presented in the study, the one with the best performance is reported in this table.
** acc = accuracy.

Figura 2.4: Comparación de las publicaciones de evaluación de habilidades quirúrgicas utilizando modelos de Deep Learning.

paso de aguja del conjunto de datos JIGSAWS. Su código fuente está disponible públicamente.

Un enfoque similar se presenta en [55], cuyo modelo es capaz de interpretar de manera confiable las habilidades dentro de una ventana de 1-3 segundos sin necesidad de observar toda la prueba de entrenamiento. En [56] propusieron un modelo de múltiples salidas, SATR-DL, para el análisis de habilidades en tiempo real de aprendices y el reconocimiento de tareas, logrando precisiones del 96 % y el 100 % para estas dos tareas.

Otros estudios realizan la tarea de evaluación automática de habilidades utilizando solo datos de video. En [57] utilizaron una ConvNet 3D logrando una precisión del 95 % en el conjunto de datos JIGSAWS. El código fuente de este trabajo está disponible en GitLab. En [58] extendieron la evaluación automática de habilidades a procedimientos de cirugía abierta, utilizando unidades de medición inercial para obtener el movimiento de las manos de los participantes. Lograron una precisión del 98.2 % en experimentos in vitro. También realizaron experimentos en el conocido conjunto de datos de cirugía robótica JIGSAWS para demostrar la generalización de su enfoque, con resultados competitivos.

En [59] propusieron una evaluación automática de habilidades microquirúrgicas para cirugía microasistida por robot basada en el aprendizaje de transferencia entre dominios. El modelo preentrenado se obtiene a través del conjunto de datos JIGSAWS y se transfiere para la evaluación de habilidades microquirúrgicas. La idea es transferir el conocimiento adquirido de JIGSAWS para acelerar el aprendizaje en el nuevo dominio.

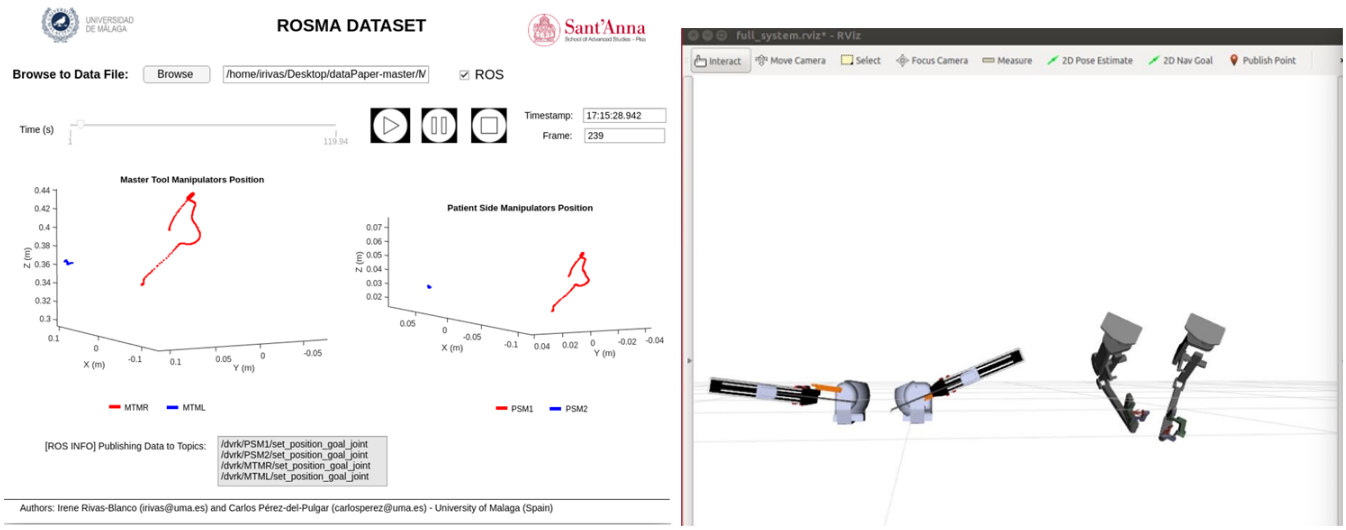
Los trabajos anteriores han demostrado la capacidad de distinguir entre cirujanos expertos, intermedios y principiantes. Sin embargo, aún queda por demostrar si las técnicas de Deep Learning pueden distinguir a los aprendices con niveles de experiencia similares entre sí. En este sentido, en [60] proponen una Red Neuronal de Disparo (SNN) para detectar a cirujanos de nivel similar utilizando solo datos cinemáticos. El propósito de este enfoque es ofrecer asistencia adaptativa durante la cirugía y el entrenamiento. De manera similar, en [61] abordan el problema de la evaluación de habilidades quirúrgicas como una tarea de clasificación por pares en la que se comparan dos acciones de entrada para identificar el mejor rendimiento quirúrgico.

Otros trabajos que abordan la evaluación de habilidades quirúrgicas se basan en el análisis del movimiento de las herramientas para extraer métricas clave para analizar el desempeño del cirujano. Esto permite, no solo clasificar el rendimiento en un nivel de experiencia, sino también proporcionar una puntuación de rendimiento para una demostración dada, lo que es muy útil para evaluar objetivamente a los aprendices. En [62]desarrollaron un enfoque que aprovecha las redes neuronales convolucionales basadas en regiones (R-CNN) para realizar la detección espacial de herramientas y luego utilizan esta información para analizar el movimiento de las herramientas. De esta manera, pueden extraer patrones de uso de herramientas, rango de movimiento y métricas de economía de movimiento para analizar las habilidades quirúrgicas. En este trabajo, utilizan una versión modificada de la rúbrica de evaluación GOALS para proporcionar una puntuación de rendimiento.

3. Descripción de la base de datos

3.1. Introducción

En este capítulo se realizará una descripción más exhaustiva de la base de datos ROSMA presentada en [13]. La base de datos ROSMA es una amplia colección de datos en robótica quirúrgica que utiliza el Da Vinci Research Kit. Los autores proporcionan en [13] un vídeo que muestra la configuración experimental utilizada para recopilar los datos, junto con un ejemplo de evaluación de cada una de las tres tareas y la interfaz gráfica de MATLAB donde se pueden visualizar los datos y reproducir las situaciones como se puede ver en la Figura 3.1.



(a) MATLAB GUI.

(b) RVIZ Simulation.

Figura 3.1: MATLAB GUI para reproducir las situaciones de las distintas tareas usando los datos de la base de datos.

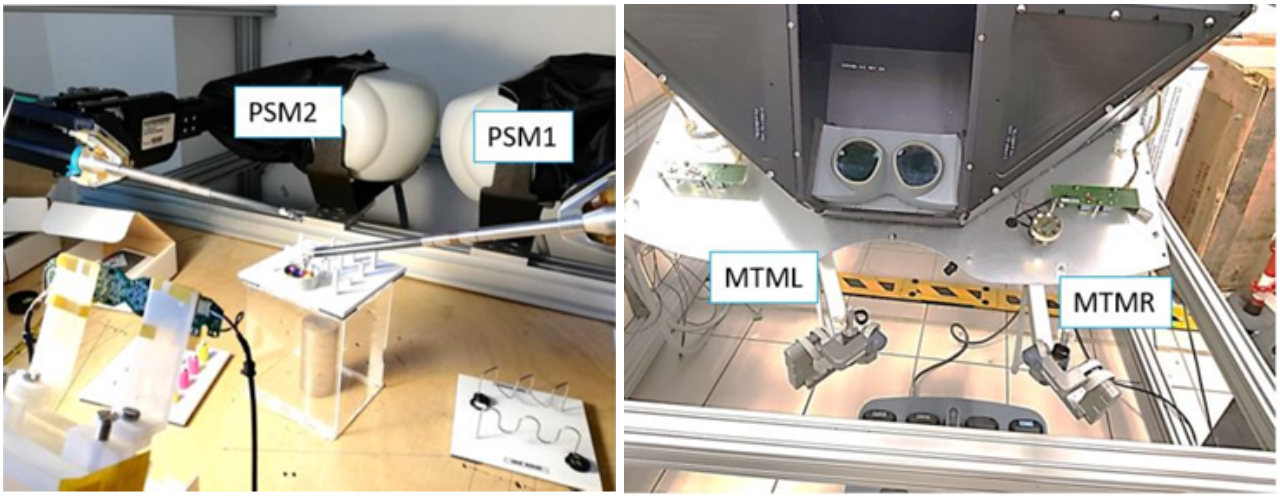
La principal fortaleza de este conjunto de datos en comparación con el de JIGSAWS se trata de la gran cantidad de datos registrados (155 variables cinemáticas, imágenes, evaluación de tareas y cuestionarios) y en el mayor número de usuarios (doce en lugar de seis). Esta gran cantidad de datos puede ser de utilidad en el campo de la inteligencia artificial para ser aplicada en la automatización de tareas en robótica quirúrgica, así como en la evaluación de habilidades quirúrgicas y el reconocimiento de gestos.

3.2. Descripción del sistema

El dVRK, respaldado por la Fundación Intuitive (Sunnyvale, CA), surgió como un esfuerzo comunitario para respaldar la investigación en el campo de la cirugía telerrobótica [35]. Está plataforma está compuesta por hardware del sistema da Vinci de primera generación, así como controladores de motores y un marco de software integrado con el Sistema Operativo Robótico (ROS) [36]. Hay más de treinta plataformas dVRK distribuidas en diez países de todo el mundo.

El Instituto de Biorrobótica de la Scuola Superiore Sant'Anna (Pisa, Italia) tiene un dVRK con dosmanipuladores del lado del paciente (llamados PSM del inglés Patient Side Manipulator), etiquetados como PSM1 y PSM2 (Figura 3.2a), y una consola maestra que consta de dos manipuladores de teleoperación (llamados MTM, del inglés Master Tool Manipulator), etiquetados como MTML y MTMR (Figura 3.2b). MTMR controla PSM1, mientras que MTML controla PSM2. Para los experimentos descritos en este documento, se utiliza visión estéreo con dos cámaras web comerciales, ya que el dVRK utilizado en los experimentos no estaba equipado con el manipulador de cámara endoscópica.

Cada PSM tiene 6 articulaciones siguiendo la cinemática descrita en [37], y un grado adicional de libertad para abrir y cerrar la pinza. La punta del instrumento se mueve alrededor de un centro de movimiento remoto, donde se establece el origen del sistema de referencia de base de cada manipulador. El movimiento de cada manipulador se describe mediante el sistema de referencia de la punta de la herramienta correspondiente con respecto a la posición de inicio del PSM. Los MTM utilizados para teleoperar remotamente los PSM tienen 7 grados de libertad, además de la



(a) Manipuladores esclavos. (b) Consola maestra.

Figura 3.2: Plataforma del Kit de Investigación da Vinci disponible en el Instituto de Biorrobótica de la Scuola Superiore Sant'Anna (Pisa, Italia).

apertura y cierre del instrumento. Los sistemas de referencia de base de cada manipulador están relacionados a través del marco común HRSV, como se muestra en la Figura 3.3 b.

3.3. Descripción de las tareas

El conjunto de datos ROSMA contiene el rendimiento de tres tareas de entrenamiento (Figura 3.4), realizadas por los doce sujetos. Se realizaron seis pruebas por tarea, lo que resulta en un total de 18 pruebas por sujeto. Sin embargo, durante el procesamiento posterior de los datos, se identificaron errores de grabación en algunas de las pruebas, lo que resultó en un número variable de pruebas para cada sujeto y tarea [16].

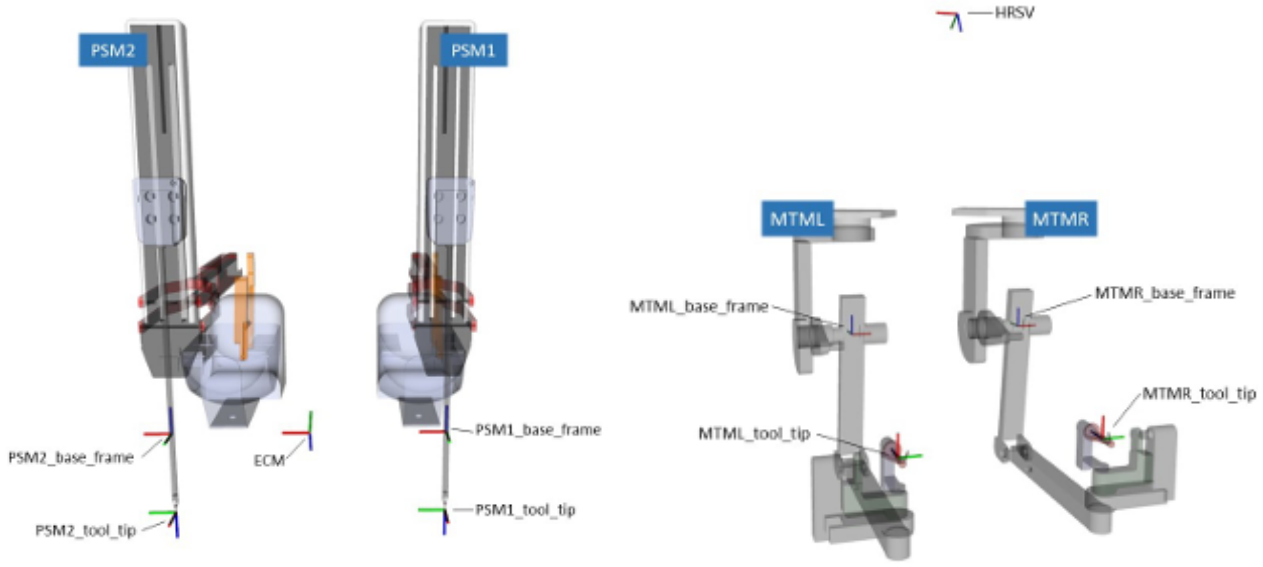
A continuación se explicarán los protocolos de cada uno de los tres experimentos que se pueden encontrar en el dataset presentado en [13].

El protocolo de la maniobra realizada en el experimento *Post and Sleeve*, visto en la Figura 3.4 a tiene las siguientes características:

- **Objetivo:** Mover los objetos de color de lado a lado de la mesa.
- **Posición inicial:** La mesa esta situada con las columnas de los tetones en posición vertical (de izquierda a derecha: 4-2-2-4). Los seis objetos estarán posicionados en uno de los lados de la mesa.
- **Procedimiento:** El usuario tenía que coger un objeto con una herramienta, pasarla a la otra herramienta y colocar el objeto en uno de los cilindros del lado opuesto de la mesa. Si un objeto se cae, es considerado una penalización y no puede recogerse de nuevo.
- **Repeticiones:** Seis ensayos: tres de derecha a izquierda y otros tres de izquierda a derecha.
- **Penalizaciones:** 15 puntos de penalización si el objeto se ha caído.
- **Puntuación:** Tiempo en segundos + puntos de penalización.

El protocolo de la maniobra realizada en el experimento *Pea on a peg*, que se puede ver en la Figura 3.4 b se define de la siguiente manera:

- **Objetivo:** Poner las esferas en los 14 cilindros de la mesa de experimentos.
- **Posición inicial:** Todas las esferas estarán en el recipiente.



(a) Cinemática del lado del paciente.

(b) Cinemática del lado del cirujano.

Figura 3.3: Cinemática desde el lado del paciente y el cirujano. La cinemática de cada PSM es definido con respecto al sistema de referencia común ECM, mientras que los MTM se definen respecto al sistema HRSV.

(a) Post and sleeve

(b) Pea on a peg

(c) Wire Chaser

Figura 3.4: Capturas de las tres tareas de la base de datos ROSMA en la posición inicial.

- **Procedimiento:** El usuario tenía que coger las esferas una por una del recipiente y colocarlas en la parte superior de los cilindros. Las esferas que se colocarán en la zona derecha se cogían con la herramienta derecha y viceversa. Si una circunferencia se caía se consideraba un error y no se podía coger de nuevo.
- **Repeticiones:** Seis ensayos: tres colocando las circunferencias en la parte derecha de la mesa y otros tres colocando las circunferencias en la parte izquierda.
- **Penalizaciones:** 15 puntos de penalización si una circunferencia se ha caído.
- **Puntuación:** Tiempo en segundos + puntos de penalización.

El protocolo de la maniobra realizada en el experimento *Wire chaser*, que se ve en la Figura 3.4 c sigue la siguiente estructura

- **Objetivo:** Mover el anillo desde una parte a otra de la mesa.
- **Posición inicial:** La mesa esta posicionada con el texto *one hand* al frente. Los tres anillos estan en la parte derecha de la mesa.
- **Procedimiento:** El usuario tenía que coger uno de los anillos y pasarlo a través del hilo metálico de un lado a otro de la mesa. Los usuarios deben usar solo una herramienta para mover el anillo, pero pueden ayudarse con

la otra mano en caso de necesitarlo. Si un anillo se cae se considera un error, pero puede cogerse de nuevo para terminar la tarea.

- **Repeticiones:** Seis ensayos: tres moviendo los anillos de derecha a izquierda y tres moviendo los anillos de derecha a izquierda.
- **Penalizaciones:** 10 puntos de penalización si un anillo se ha caído.
- **Puntuación:** Tiempo en segundos + puntos de penalización.

En este TFM se va a realizar únicamente la evaluación automática de la tarea *Post and sleeve* ya que en las demás tareas resulta de gran complejidad detectar los objetos con los que se realizan.
En el caso de la tarea *Pea on a peg* las esferas de colores tienen un tamaño reducido por lo que es díficil detectarlas mediante segmentación de colores ya que los ruidos afectarían la detección de manera considerable.
En el caso de *Wire chaser* al ser los objetos de color oscuro pueden ser confundidos con el fondo de los vídeos y el cable metálico puede dar problemas al poder confundirse con las herramientas.

Las herramientas al ser un elemento común si se podrían detectar en las tres tareas de las que se dispone en el dataset descrito en [13]. De este modo no sería posible evaluar automáticamente las otras dos tareas, ya que sería necesario detectar la posición de los objetos.

3.4. Estructura del dataset

El conjunto de datos ROSMA se divide en tres tareas de entrenamiento realizadas por doce sujetos. Los experimentos se llevaron a cabo de acuerdo con las recomendaciones de su institución, con el consentimiento informado por escrito de los sujetos, de conformidad con la Declaración de Helsinki. Antes de comenzar el experimento, a cada sujeto se le informó sobre el objetivo de los ejercicios y las métricas de error. La duración total de los datos registrados es de 8 horas, 19 minutos y 40 segundos, y la cantidad total de datos cinemáticos es de aproximadamente 1.5 millones para cada parámetro. El conjunto de datos consta de 415 archivos, distribuidos de la siguiente manera: 206 archivos de datos en formato CSV (valores separados por comas), 206 archivos de datos de vídeo en formato MP4, un archivo en formato CSV con la evaluación de los ejercicios, llamado 'scores.csv', un archivo en formato TXT con los datos de sincronización entre los archivos CSV y los archivos de vídeo, llamado synchronizationData.txt', y un archivo, también en formato CSV, con las respuestas del cuestionario personal, llamado 'User questionnaire-dvrkDatasetExperiment.csv'.

El nombre de los archivos de datos y vídeo sigue la jerarquía:<ID de usuario>nombre de la tarea<número de intento>. La descripción de cada uno de estos campos es la siguiente:

- <ID de usuario>: proporciona un identificador único para cada usuario y va desde 'X01' hasta 'X12'.
- <Nombre de la tarea>: puede ser una de las siguientes etiquetas, según la tarea que se esté realizando: 'Pea on a Peg', 'Post and Sleeve' o 'Wire Chaser'.
- <Número de intento>: es el número de repetición del usuario en la tarea actual y varía de '01' a '06'.

Por ejemplo, el nombre del archivo 'X03 Pea on a Peg 04' corresponde al cuarto intento del usuario 'X03' realizando la tarea Pea on a Peg.

Cada usuario realizó un total de seis intentos por tarea, pero durante el procesamiento posterior de los datos, los autores encontraron errores de grabación en algunos de ellos. Esa es la razón por la que algunos usuarios tienen menos intentos en ciertas tareas.

1. **Archivos de datos:** Los archivos de datos están en formato CSV y contienen 155 columnas: la primera columna, etiquetada como 'Fecha', tiene la marca de tiempo de cada conjunto de medidas, y las otras 154 columnas contienen los datos cinemáticos de los manipuladores del lado del paciente (PSMs) y los manipuladores del lado maestro (MSMs). La estructura de estas 154 columnas se describe en la Tabla III, que muestra los índices de columna para cada movimiento cinemático, el número de columnas, la etiqueta descriptiva de cada variable, las unidades de datos y los publicadores de ROS de los datos. Las etiquetas descriptivas de las columnas tienen el siguiente formato: <nombre del componente><movimiento cinemático><variable>. Los valores de la marca de tiempo tienen una precisión de milisegundos y se expresan en el formato: Año-Mes-Día.Hora:Minutos:Segundos.Milisegundos. Dado que los datos se registraron a 50 muestras por segundo, el intervalo de tiempo entre las filas es de 20 ms.

2. **Archivos de vídeo:** Las imágenes se grabaron con una de las dos cámaras web comerciales utilizadas durante los experimentos para lograr la visión estéreo, a una velocidad de 15 imágenes por segundo y una resolución de 1024 x 768 píxeles. La hora del reloj interno de la computadora que grabó las imágenes se muestra en la esquina superior derecha de las imágenes, con una precisión de segundos. La cámara web se colocó frente al sistema dVRK de modo que PSM1 esté en el lado derecho de las imágenes y PSM2 en el lado izquierdo. La matriz de proyección de la cámara que relaciona los puntos 3D del mundo desde los archivos de datos con sus correspondientes proyecciones en imágenes se describe en la Sección IV.

3. **Sincronización de datos:** Los datos cinemáticos y los datos de vídeo se registraron utilizando dos computadoras diferentes, ambas ejecutando Ubuntu 16.04. Los relojes internos de estas computadoras se sincronizaron en una referencia de tiempo común utilizando un servidor de Protocolo de Tiempo en Red (NTP). La sincronización de los relojes internos se repitió antes de comenzar el experimento de un nuevo usuario si había un período de inactividad entre usuarios superior a una hora. Dado que las imágenes y los datos se registraron por separado, aunque ambas computadoras estaban sincronizadas en tiempo real, los archivos de vídeo y datos no comienzan al mismo tiempo, es decir, para un intento en particular, la marca de tiempo del primer fotograma del vídeo no corresponde a los datos de la primera fila del archivo de datos correspondiente. Por lo tanto, se realizó una sincronización manual entre los archivos de vídeo y datos para proporcionar el fotograma de vídeo inicial y la fila de datos con la misma marca de tiempo. Para cada intento, el procedimiento de sincronización se realizó de la siguiente manera: Dado que el tiempo mostrado en los vídeos tiene una precisión de segundos, para cada intento, buscamos manualmente el fotograma con la primera pausa de segundos. Luego, buscamos la fila en el archivo de datos correspondiente con la marca de tiempo que corresponde al tiempo mostrado en el vídeo, es decir, de las 50 muestras por segundo, seleccionamos la fila correspondiente a la primera. De esta manera, el error máximo en este punto de sincronización es de 20 ms. Esta sincronización manual se almacena en el archivo 'synchronizationData.txt', con la siguiente estructura: <intentos><fotograma inicial><fila inicial>. Por lo tanto, para usar los datos, se deben obviar los fotogramas de vídeo antes del 'fotograma inicial' y los datos correspondientes con filas antes de la 'fila inicial'.

4. **Evaluación de ejercicios:** El archivo 'scores.csv' contiene la evaluación de cada ejercicio de acuerdo con la puntuación de la Tabla II. Por lo tanto, para cada intento, se muestra el tiempo de ejecución de la tarea (en segundos), el número de errores y la puntuación final.

5. **Cuestionario personal:** Después de completar el experimento, se pidió a los participantes que completaran un formulario para recopilar datos personales que podrían ser útiles para futuros estudios y análisis de los datos. El formulario contiene preguntas relacionadas con los siguientes temas: edad, mano dominante, preferencias de tareas, antecedentes médicos, experiencia previa en el uso del da Vinci u otro dispositivo teleoperado y habilidades de coordinación mano-ojo. Las preguntas que requieren un nivel de experiencia son de opción múltiple y van desde 1 (bajo) hasta 5 (alto)

4. Reconocimiento de herramientas quirúrgicas

4.1. Introducción

Las redes neuronales (Figura 4.1) han experimentado un rápido avance en los últimos años y se han convertido en una herramienta fundamental en diversos campos , como el procesamiento de imágenes, el procesamiento de lenguaje natural, la visión por ordenador, la medicina y muchos otros como se puede ver en [20], [21] y [22].

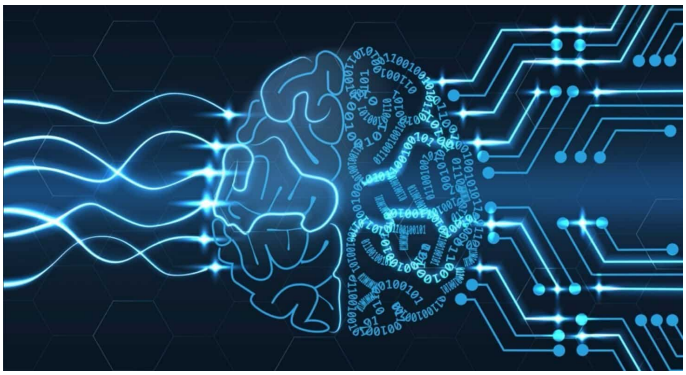


Figura 4.1: Comparación gráfica entre el cerebro y las redes neuronales.

Las redes neuronales están compuestas por unidades fundamentales llamadas neuronas artificiales [23]. Cada neurona artificial recibe múltiples entradas, aplica una transformación lineal (pesos y sesgos) seguida de una función de activación no lineal y produce una salida. Estas neuronas se agrupan en capas [21]. Una red neuronal típica consta de una capa de entrada, una o varias capas ocultas y una capa de salida. La información fluye de la capa de entrada a través de las capas ocultas hasta llegar a la capa de salida [20] (Figura 4.2).

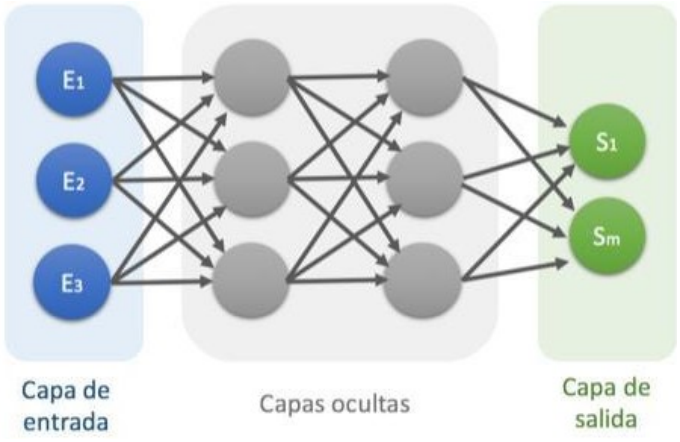


Figura 4.2: Arquitectura básica de una Red Neuronal.

Las conexiones entre las neuronas están ponderadas por pesos. Cada peso representa la importancia relativa de la conexión correspondiente [22]. Durante el entrenamiento de la red neuronal, estos pesos se ajustan para optimizar el rendimiento de la red. Después de la transformación lineal, se aplica una función de activación no lineal a la salida de cada neurona [20] (Figura 4.3). Esto introduce la capacidad de las redes neuronales para aprender relaciones no lineales en los datos.

Durante la propagación hacia adelante (forward propagation), los datos de entrada se propagan a través de la red neuronal capa por capa [23]. Cada neurona realiza su cálculo y pasa su salida a las neuronas de la capa siguiente.

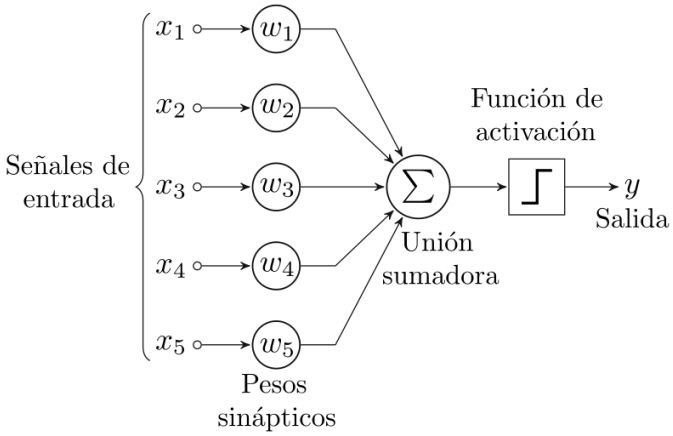


Figura 4.3: Esquema básico de pesos y función de activación de una Red Neuronal.

Después de que los datos se propagan por toda la red, se compara la salida predicha con el valor objetivo utilizando una función de pérdida. Esta función mide la discrepancia entre las predicciones y los valores reales [22].

La retropropagación del error (backward propagation) es el algoritmo clave utilizado para ajustar los pesos de la red neuronal durante el entrenamiento (Figura 4.4) [22]. El error se propaga hacia atrás desde la capa de salida hasta la capa de entrada, y los pesos se actualizan en función de la contribución de cada neurona al error total. Para mejorar el rendimiento de la red neuronal, se utilizan algoritmos de optimización que ajustan iterativamente los pesos de la red en función del error calculado[21].

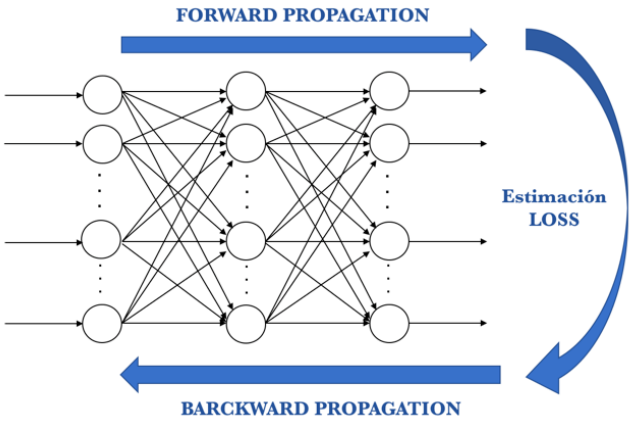


Figura 4.4: Esquema básico de propagación hacia adelante/atrás y uso del error.

Las Redes Neuronales Convolucionales (CNN) son las arquitecturas de Deep Learning más conocidas y las más utilizadas en aplicaciones de procesamiento de imágenes. Algunas implementaciones bien conocidas de CNN incluyen AlexNet [20], VGGNet [22], GoogleNet [23] o ResNet [50]. Una CNN se compone de una serie de capas de convolución y pooling seguidas de una capa completamente conectada. El papel de cada una de estas capas de la red es el siguiente:

1. Capas de Convolución: la operación de convolución aprende patrones locales para extraer las características de alto nivel de la imagen de entrada. Por lo general, la primera capa de convolución captura características de bajo nivel, como bordes o colores, mientras que las capas exteriores proporcionan una comprensión de alto nivel de las imágenes.

2. Capas de Pooling: el objetivo de estas capas es reducir la dimensionalidad de los mapas de características a través de una función como max-pooling o average-pooling.

3. Capas Completamente Conectadas: estas capas son responsables de la clasificación real de la imagen al aprender combinaciones no lineales de las características de alto nivel extraídas en las capas anteriores.

El diseñador de la red debe decidir el número óptimo de capas para lograr un equilibrio entre un buen rendimiento del modelo, generalización con nuevos datos y alta velocidad computacional para realizar inferencias en tiempo real. Las redes poco profundas modelan un número reducido de parámetros y, por lo tanto, pueden realizar predicciones muy rápidas con una mayor generalización pero menos precisión. En este caso, la red aún no ha modelado todos los parámetros relevantes de los datos de entrenamiento.

En contraste, las redes muy profundas modelan un alto número de parámetros y pueden proporcionar predicciones de alta precisión para los datos de entrenamiento, pero pueden carecer de generalización a nuevos datos debido al sobreajuste de la red, es decir, la red puede estar aprendiendo patrones específicos de los datos de entrenamiento que no son relevantes para nuevos datos. Otro factor clave para diseñar buenos modelos predictivos es contar con grandes cantidades de datos etiquetados para el entrenamiento. Sin embargo, obtener una cantidad suficiente de datos anotados en dominios específicos como la cirugía es difícil y costoso. Para aliviar este problema, la mayoría de las redes se preentrenan utilizando datos etiquetados de otros dominios, como ImageNet [10].
Una de las limitaciones de las CNN es que no pueden manejar tamaños de imagen de entrada variables. En contraste, las Redes Neuronales Convolucionales Completamente Convolucionales (FCNN) tienen la ventaja sobre las CNN de operar con entradas de cualquier tamaño, produciendo una salida con dimensiones espaciales reducidas. Esto las hace adecuadas para la etiquetación semántica de píxeles de extremo a extremo, ya que la configuración espacial de la imagen se conserva a lo largo de las capas. Sin embargo, carecen de capacidades en tiempo real y las máscaras suelen tener agujeros o no respetan los bordes. Otra limitación de las CNN es que carecen de la capacidad para procesar información temporal de datos que se presentan en secuencias, como los datos de vídeo.

Para considerar las dependencias temporales en los datos de entrada, utilizamos Redes Neuronales Recurrentes (RNN). A diferencia de las redes neuronales de alimentación directa, las unidades de procesamiento en una RNN forman un ciclo. Esto permite que la red tenga memoria de los estados anteriores y use esa memoria para influir en la salida actual. La principal implementación de las RNN son las redes LSTM (Long Short Term Memory). Una LSTM consta de bloques de estado de memoria a través de los cuales fluye la señal mientras está regulada por compuertas de entrada, olvido y salida [51]], lo que permite agregar o eliminar información del estado de la celda.

La puerta de entrada decide qué valores se actualizarán, mientras que la puerta de olvido se utiliza para descartar información. Finalmente, la puerta de salida retiene la información que no se utiliza en el paso de tiempo actual, pero que puede ser útil en el futuro. Para aprovechar ambas redes, muchos autores proponen modelos de Deep Learning que combinan CNN con RNN conectadas en una configuración en serie. Estos modelos utilizan una CNN para extraer características espaciales de las imágenes de entrada, y su salida se alimenta a una RNN para tener en cuenta el contexto temporal de los datos.

De todas las opciones disponibles se va a usar la red YOLO, ya que este tipo de redes suelen utilizarse en aplicaciones de detección de objetos en imágenes y vídeos en tiempo real. Son especialmente conocidas por su capacidad para realizar la detección de objetos de manera rápida y eficiente, ya que procesan una imagen completa de una sola vez en lugar de dividirla en múltiples regiones de interés, lo que reduce significativamente el tiempo de procesamiento.
Algunos de los usos más comunes de las redes YOLO son la detección de objetos en cámaras de seguridad, en la industria de automóviles autónomos, en detección de objetos en imágenes médicas, detección de objetos en drones, automatización de robots, etc. A continuación, se explicarán más en profundidad este tipo de redes y sus características mas representativas.

4.2. Redes YOLO (You Only Look Once)

Las redes YOLO (You Only Look Once) son un conjunto de algoritmos de detección de objetos ampliamente reconocidos en el campo de la visión por ordenador. Estas redes están diseñadas para detectar y localizar objetos en imágenes y vídeos en tiempo real, proporcionando resultados precisos y veloces. A lo largo de los años, las redes YOLO han experimentado varias iteraciones y mejoras que han llevado a un estado del arte en la detección de objetos [24].

La primera versión de YOLO, conocida como YOLO v1, fue presentada en 2015. Este enfoque introdujo un nuevo paradigma al tratar la detección de objetos como un problema de regresión en lugar de clasificación. YOLO v1 utilizaba una única red neuronal convolucional (CNN) que dividía la imagen de entrada en una cuadrícula y predecía cajas delimitadoras y las probabilidades de clase asociadas para cada región de la cuadrícula. Aunque era rápido, tenía limitaciones en la detección de objetos pequeños y en la precisión de la localización [24].

En 2016, se presentó YOLO v2 como una mejora significativa de su predecesor. Esta versión incorporó varias mejoras clave, como el uso de detección a múltiples escalas mediante el uso de capas pasantes (skip connections) para capturar características de diferentes niveles de resolución. Además, YOLO v2 implementó cajas priorizadas (anchors) para mejorar la precisión de la localización y empleó una estrategia de entrenamiento en dos etapas para mejorar la precisión global. Estas mejoras permitieron una detección más precisa y un rendimiento mejorado [24].

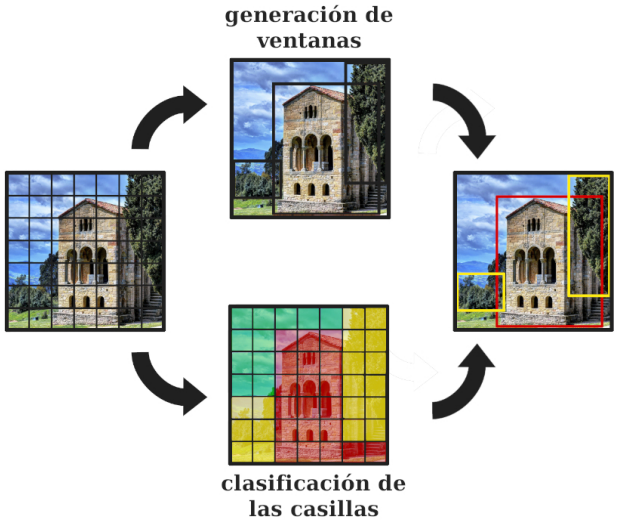


Figura 4.5: Pasos de red YOLO priorizando las celdas.

Posteriormente, en 2018, se lanzó YOLO v3, que presentó una arquitectura aún más profunda y sofisticada. Esta versión utilizó una red llamada Darknet-53, que consta de 53 capas convolucionales, para extraer características más ricas y mejorar la precisión de detección [24]. Además, YOLO v3 incorporó múltiples tamaños de cajas priorizadas (anchors) y una técnica conocida como "predicción de múltiples escalas", que permitió detectar objetos a diferentes resoluciones y escalas. Estas mejoras llevaron a un avance significativo en la precisión y la capacidad de detección de YOLO (Figura 4.6).

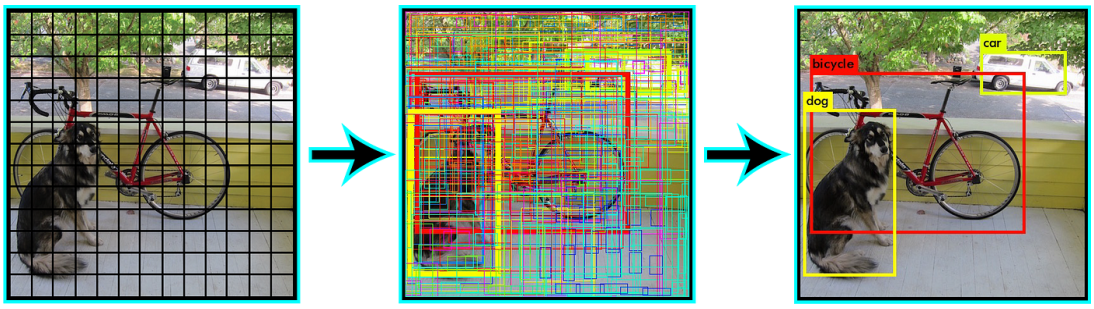


Figura 4.6: Pasos usando YOLOv3.

En 2020, se presentaron dos versiones adicionales: YOLO v4 y YOLO v5. YOLO v4 incorporó varias técnicas avanzadas, como la normalización de lotes con pesos móviles (Mish), bloques CSPDarknet53, atención espacial (SPP) y detección de objetos con enfoque (SAM). Estas mejoras condujeron a una mayor precisión y un rendimiento notablemente mejorado. Por otro lado, YOLO v5 se enfocó en mejorar la velocidad y la eficiencia al adoptar una arquitectura más simple y liviana basada en el modelo EfficientNet [25]. Esta versión también se centró en el entrenamiento en dispositivos con recursos limitados, como teléfonos móviles y sistemas embebidos.

En [38] se explica como la red YOLO cambia el tamaño de la imagen de entrada y ejecuta una única red convolucional en la imagen estableciendo un umbral para detectar las regiones según la confianza del modelo creado.
Se comenta también como generar cuadros delimitadores potenciales en una imagen y luego ejecutar un clasificador

en estos cuadros propuestos. Después de la clasificación, el pos-procesamiento se utiliza para refinar los cuadros delimitadores, eliminando las detecciones duplicadas y volviendo a clasificar los cuadros en función de otros objetos de la escena. Estos procesos complejos son lentos y difíciles de optimizar porque cada componente individual debe entrenarse por separado.

Por tanto en [38] se replantea la detección de objetos como un problema de regresión única, directamente desde los píxeles de la imagen hasta las coordenadas del cuadro delimitador y las probabilidades de clases que se definen. Usando este sistema, solo se mira una vez la imagen para predecir qué objetos están presentes y dónde están. También se comenta que pese a su rapidez YOLO aún sigue por detrás de los sistemas de última generación en cuanto al tema de precisión. Aunque es cierto que este tipo de red puede identificar rápidamente objetos en imágenes, le es complicado detectar con precisión objetos especialmente pequeños.

En [38] se explica como la red YOLO usa características de toda la imagen para predecir cada cuadro delimitador, también predice todos los cuadros delimitadores y las clases para una imagen simultáneamente. Esto lo hace dividiendo la imagen en una rejilla de SxS, si el centro de un objeto esta dentro de una celda de la rejilla, esa celda se encarga de detectar el objeto.
Cada celda predice B cuadros delimitadores y la precisión para estos cuadros. En caso de que no exista ningún objeto en esa celda esta puntuación debe ser 0. Por otro lado, la precisión se quiere para igualar la intersección sobre la unión (IOU) entre las cuadros predecidos y los cuadros de verdad (ground truth). Por tanto, cada celda predice C probabilidades de clases condicionales. Y se predice una probabilidad de set de clases por celda, independientemente del número de casillas B. Para obtener la precisión se multiplican las probabilidades de clases condicional y las predicciones de cuadros individuales de confianza. Todo esto se puede apreciar visualmente en la Figura 4.7.

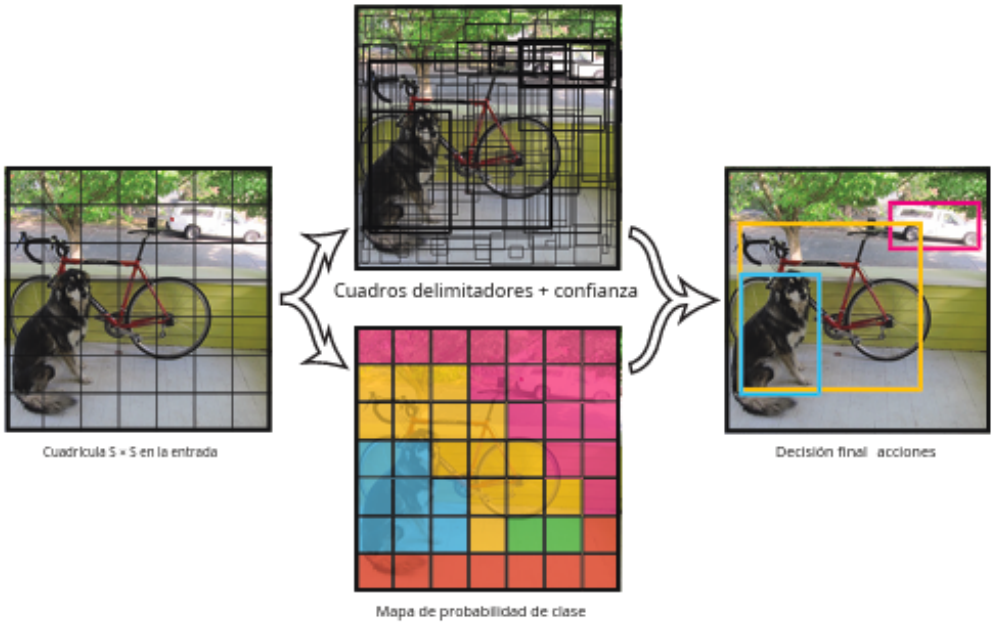


Figura 4.7: Obtención de la precisión de la red YOLO.

Las redes YOLO tienen grandes restricciones espaciales, ya que cada celda de la cuadrícula solo predice dos cuadros y solo puede tener una clase. Esta restricción espacial limita la cantidad de objetos cercanos que estas redes pueden predecir.
Como este tipo de redes aprenden a predecir cuadros delimitadores a partir de datos, le resulta difícil generalizar a objetos en relaciones de aspecto o configuraciones nuevas o inusuales. Estas redes también utilizan características relativamente simples para predecir cuadros delimitadores, ya que su arquitectura tiene múltiples capas de reducción de resolución de la imagen de entrada.
Finalmente, mientras estas redes entrenan con una función de pérdida que se aproxima al rendimiento de detección, la función de pérdida trata los errores de la misma manera en cuadros delimitadores pequeños que en cuadros delimitadores grandes. Un pequeño error en una casilla grande es generalmente benigno, pero un pequeño error en una casilla

pequeña tiene un efecto mucho mayor en el conjunto. La principal fuente de error de estas redes son las localizaciones incorrectas.

4.3. Etiquetado de la base de datos

Antes de entrenar la red neuronal es necesario etiquetar los datos disponibles, y para que los resultados de detección sean buenos es necesario tener un gran número de etiquetas, normalmente unas 3000 imágenes suelen ser suficientes para obtener unos resultados adecuados. En este TFM, han sido necesarias 28890 imágenes, divididas entre entrenamiento (19965 imágenes) y test (8925 imágenes), debido a que se necesita un acierto muy alto para poder realizar con eficiencia la evaluación automática de la tarea. En el apartado 4.6.1 se explica como se ha realizado el proceso de etiquetado.

Por tanto, se han seleccionando una serie de vídeos, los cuáles estarán divididos según la función deseada. De forma habitual se dividen de la siguiente manera: se usa una cantidad de imágenes que den un porcentaje aproximado del 70 % del total para entrenar la red y en un 30 % para realizar el test. En el caso particular de este TFM, se han usado 19965 imágenes para entrenamiento y 8925 para test. En la Tabla 4.1 viene el desglose de los vídeos etiquetados usados y que función tienen, así como el porcentaje de imágenes dedicados para realizar entrenamiento y test. Como se puede ver, se ha empleado el 69 % de las imágenes para entrenamiento y el 31 % de las imágenes para el test.

Utilidad de los vídeos etiquetados	Vídeo etiquetado	imágenes usados	Porcentaje de imágenes usados
Entrenamiento	X01 Pea on a peg 01	1845	**0.69 % (19965 imágenes)**
	X01 Pea on a peg 02	1530	
	X02 Pea on a peg 01	2250	
	X02 Pea on a peg 02	2145	
	X03 Pea on a peg 01	1905	
	X03 Pea on a peg 02	1680	
	X04 Pea on a peg 01	2880	
	X04 Pea on a peg 02	1845	
	X01 Post and sleeve 01	1905	
	X11 Post and sleeve 04	1980	
Test	X01 Pea on a peg 06	2040	**0.31 % (8925 imágenes)**
	X02 Pea on a peg 06	2325	
	X03 Pea on a peg 06	2295	
	X04 Pea on a peg 04	2265	

Tabla 4.1: División final de los imágenes etiquetados para entrenamiento y test.

4.4. Entrenamiento de la red neuronal

Entrenar una red neuronal YOLO, es un proceso que implica varios pasos fundamentales para que la red pueda aprender a realizar tareas específicas. El primer paso es recopilar un conjunto de datos que contenga imágenes o vídeos que representen las condiciones en las que se desea que la red realice la detección de objetos. Cada imagen o fotograma debe estar etiquetado con la ubicación y la clase de los objetos que se desean detectar. El conjunto de datos etiquetados se ha descrito en el apartado 4.3.

Posteriormente, las imágenes y las etiquetas deben ser preprocesadas para que sean adecuadas para el entrenamiento de la red. Esto puede incluir operaciones como el re-dimensionamiento de imágenes, la normalización de píxeles, y la conversión de etiquetas de objetos a un formato específico que la red pueda entender. Una vez realizado el preprocesamiento de datos, es necesario configurar los distintos parámetros de la red como se ha descrito en 4.5.

A continuación, se comienza el entrenamiento alimentando la red con el conjunto de datos etiquetados. El entrenamiento implica iteraciones donde se presentan las imágenes de entrenamiento a la red, se calcula la pérdida (diferencia entre las predicciones y las etiquetas reales) y se ajustan ciertos parámetros de la red a través de la retropropagación. Una vez completado el entrenamiento de la red, se debe evaluar su rendimiento mediante un conjunto de datos de test independiente al de entrenamiento para medir su capacidad de detección de objetos en condiciones reales.

Una vez que el rendimiento de la red entrenada sea satisfactorio, se puede integrar en la aplicación para realizar tareas de detección de objetos. Es importante destacar que el entrenamiento de redes YOLO es un proceso intensivo en términos de recursos computacionales y puede requerir hardware especializado, como GPUs, para acelerar el proceso. Además, la elección de los parámetros y la calidad del conjunto de datos de entrenamiento son factores críticos para obtener un buen rendimiento en la detección de objetos.

4.5. Diseño de la red neuronal

Para el diseño de la red neuronal es necesario saber como se han configurado los distintos parámetros de interés y como se ha adaptado lo descrito en [19] a las necesidades de este TFM, por eso a continuación se explican que parámetros se han configurado y porque:

- Tamaño de la red: Al elegir el tamaño de entrada de red, se debe tener en cuenta el tamaño mínimo necesario para ejecutar la propia red, el tamaño de las imágenes de entrenamiento y el costo computacional en el que se incurre al procesar los datos en el tamaño seleccionado. Normalmente se elije un tamaño de entrada de red que sea parecido al tamaño de las imágenes de entrenamiento y mayor que el tamaño de entrada requerido para la red. En este caso se ha elegido un tamaño de [227 227 3] para reducir el coste computacional, ya que si se aumenta el tamaño de la red, el coste computacional aumenta considerablemente.
- Datos de entrenamiento para estimación: El tamaño de la red se usa para preprocesar los datos de entrenamiento y calcular los cuadros de anclaje, porque las imágenes de entrenamiento son mayores a 227 x 227.
- Número de anclajes: Se ha definido el número de anclajes como 6 para tener un buen equilibrio entre el número de anclajes y la meanIoU (el significado de estos parámetros se comentará en el apartado de implementación).
- Se han establecido un número de épocas igual a 80, ya que se considera un número suficiente para el entrenamiento de la red.
- Se ha establecido un tamaño de mini lote o batch en 8. A mayor tamaño de mini lotes es posible un entrenamiento más estable con tasas de aprendizaje también altas. El problema es que este parámetro depende la memoria disponible, por este motivo se ha establecido ese tamaño de mini lote.
- Se ha establecido una tasa de aprendizaje del 0.001, ya que se considera una tasa aceptable para el entrenamiento de la red.
- El periodo de preparación denota el número de iteraciones que se usa para aumentar exponencialmente la tasa de aprendizaje. También ayuda a estabilizar los gradientes a tasas de aprendizaje más altas. Se ha decidido tomar un valor de 1000, ya que se considera suficiente para el entrenamiento que se quiere realizar.
- El umbral de penalización se ha definido como 0.5, esto quiere decir que se penalizarán las detecciones que se superpongan menos de 0.5 de los cuadros de entrenamiento.

4.6. Implementación

En este apartado se ha hecho uso de MATLAB para realizar la implementación de la red neuronal, para realizar el entrenamiento de la red se ha hecho uso de varías toolbox de MATLAB, incluyendo *Computer Vision Toolbox*, *Deep Learning Toolbox*, *Image Processing Toolbox* y *Parallel Computing Toolbox*. En [19] se hace una descripción cómo utilizar las toolbox mencionadas para entrenar un detector de objetos utilizando YOLO v3.
A continuación, se realiza una breve explicación de los principales puntos usados para la implementación:

- Se ha hecho uso de una red preentrenada utilizando la función *downloadPretrainedYOLOv3Detector*. Esto evita tener que esperar a que se complete el entrenamiento desde cero, agilizándolo. Aunque también es posible entrenar la red desde cero si se deseará.
- Se utilizarán datos etiquetados previamente que contendrán imágenes del dataset usado con información sobre las herramientas que se desean etiquetar mediante la red YOLO.
- Se hace una aumento de los datos para mejorar la precisión del modelo mediante la transformación aleatoria de los datos originales durante el entrenamiento. Se aplican varias transformaciones, como cambios en el color, volteo horizontal y cambio de escala, a las imágenes de entrenamiento para aumentar la variedad de los datos.

- Se ha definido el detector YOLO v3, en este caso se ha basado en la red *SqueezeNet* y utiliza dos cabezales de detección al final. El segundo cabezal de detección tiene el doble de tamaño que el primero, por lo que es capaz de detectar objetos más pequeños. Se utilizan *anchor boxes* estimadas a partir de los datos de entrenamiento para tener prioridades iniciales mejores y ayudar a la red a predecir los cuadros delimitadores con precisión.
- Los datos de entrenamiento se han preprocesado para que sean adecuados para el entrenamiento. Se han cambiado el tamaño de las imágenes y escalado los píxeles de las imágenes.
- Se han definido los parámetros de entrenamiento.
- Se inicia el entrenamiento del modelo utilizando un bucle. Se utilizan datos de entrenamiento preprocesados y se aplican gradientes, regularización y los parámetros de entrenamiento para ajustar el modelo.
- Se evalúa el rendimiento del detector de objetos utilizando métricas como la precisión promedio (average precision) y se muestra la curva de precisión-recall.
- Finalmente, se muestra cómo utilizar el detector entrenado para realizar detecciones de objetos en nuevas imágenes.

4.6.1. Proceso de etiquetado

Antes de diseñar y entrenar la red neuronal capaz de reconocer las herramientas en la imagen, es necesario etiquetar la base de datos con la localización de las herramientas. Para realizar este proceso, MATLAB ofrece una herramienta muy potente llamada, *vídeo Labeler* [30]. Dicha herramienta se trata de una aplicación que facilita el etiquetado de regiones de interés rectangulares (ROI), etiquetas de ROI de polilínea, etiquetas de ROI de píxeles y etiquetas de escena en un vídeo o secuencia de imágenes.

Los datos etiquetados mediante esta herramienta se pueden usar para validar o entrenar algoritmos, como clasificadores de imágenes, detectores de objetos y redes de segmentación semántica e instancia.

Entre otras funcionalidades, la herramienta *vídeo Labeler* nos ofrece las siguiente posibilidades:

- Etiquetar manualmente un cuadro de imagen de un vídeo.
- Etiquetar automáticamente a través de cuadros de imagen utilizando un algoritmo de automatización.
- Exportar el archivo *gTruth* con los datos de las etiquetas.

Un archivo gTruth generalmente contiene información relacionada con la verdad fundamental o ground truth en el contexto de procesamiento de imágenes, visión por ordenador y tareas de aprendizaje automático. Este archivo se utiliza para anotar y almacenar información sobre objetos o regiones de interés en imágenes o secuencias de imágenes. Algunos de los elementos que suelen tener los archivos gTruth son:

- Etiquetas de clase: Identificación de las clases o categorías de objetos presentes en las imágenes. En el caso de este TFM *Right_tool* y *Left_tool*.
- Localización de regiones de interés: Coordenadas espaciales que definen la ubicación y el tamaño de regiones específicas en una imagen. Estas regiones suelen estar asociadas con las clases. En el caso de este TFM, las coordenadas del delimitador alrededor de una herramienta.
- Etiquetas de Clase para Regiones: Asociación de etiquetas de clase a regiones específicas en una imagen. En este TFM, se indica que herramienta se encuentra en una región de interés.
- Tiempo: Información sobre el tiempo en secuencias de imágenes, lo que es útil en tareas de seguimiento de objetos en vídeo. En este caso se indicar que etiquetas y en que localización se tiene asociado a cada frame.

Para realizar el proceso de etiquetado de las imágenes, en primer lugar se deben cargar los datos no etiquetados, posteriormente definir las etiquetas que se necesitan dibujar y establecer un intervalo de tiempo. Después, se pueden crear etiquetas ROI, subetiquetas, atributos y etiquetas de escena. Por último, se pueden etiquetar los datos *gTruth* manualmente o utilizar un algoritmo de automatización.

Una vez que se haya completado el etiquetado del vídeo, se pueden exportar los datos en formato *gTruth* para su uso en algoritmos de visión por ordenador basados en Deep Learning. Para el uso de esta aplicación es necesario la

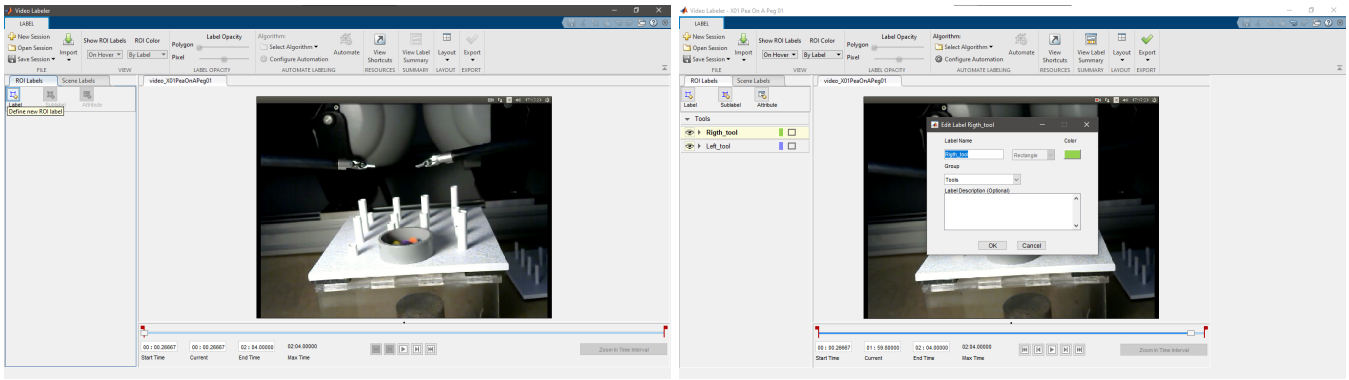
instalación de *Computer Vision Toolbox*.

Conocidas las opciones disponibles dentro de la herramienta *vídeo Labeler* se procede al etiquetado de las herramientas en los vídeos. Con la aplicación *vídeo Labeler*.

El objetivo de la red neuronal es reconocer tanto la herramienta izquierda como la herramienta derecha en la imagen, de forma separada. Por tanto, la base de datos se ha etiquetado con las siguientes etiquetas:

- *Right_Tool*: correspondiente a la herramienta derecha.
- *Left_Tool*: correspondiente a la herramienta izquierda.

En la Figura 4.8 se muestra un ejemplo de una imagen etiquetada, así como de los pasos previos. Como se puede ver en dicha Figura se observa que la parte etiquetada de ambas herramientas han sido las muñecas, ya que es la parte que es de utilidad, porque es la que interactúa con los demás objetos.



(a) Primer paso para la creación de etiquetas.

(b) Creación de etiquetas en grupo.

Figura 4.8: Creación de las etiquetas.

Una vez definidas las etiquetas, se indica el ROI (Region of Interest) correspondiente a cada etiqueta en cada imagen de los vídeos que posteriormente se usarán para el entrenamiento de la red, en la Figura 4.9 se pueden ver las ROI´s definidas en un imagen.

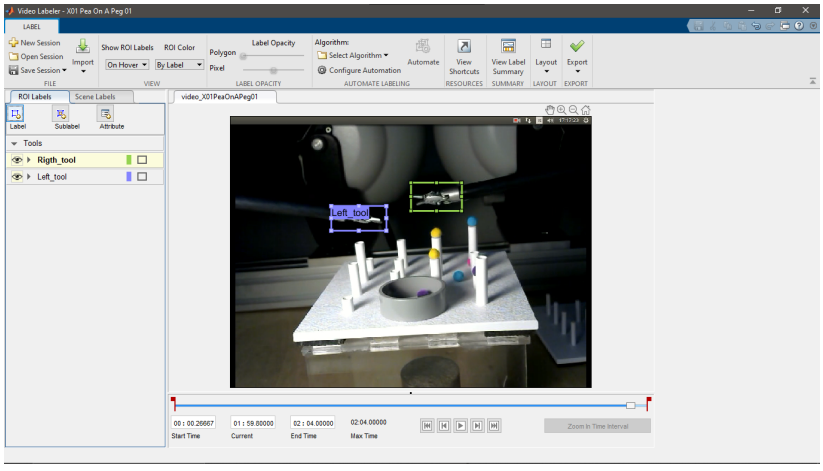


Figura 4.9: Ejemplo de una imagen etiquetada en la que se muestran las regiones de interés etiquetadas.

Para agilizar el etiquetado y no ir imagen por imagen etiquetando las herramientas manualmente, en la herramienta (*vídeo Labeler*) se puede encontrar un algoritmo de automatización, llamado *Point Tracker*, este algoritmo es capaz de seguir una o más ROIs rectangulares en intervalos cortos utilizando el algoritmo Kanade-Lucas-Tomasi (KLT).

Aunque el método automatizado no realiza el etiquetado de un vídeo entero correctamente, se puede hacer por tramos e ir corrigiendo los ROI´s no realizados correctamente por el método automático.

4.6.2. Entrenamiento de la red

La implementación del proceso de entrenamiento se ha realizado siguiendo los siguientes pasos, descritos en el pseudocódigo 4.1. Para iniciar el entrenamiento, el primer paso a dar es cargar los datos de los vídeos etiquetados. Los archivos de las imágenes etiquetadas están formados por la propia imagen, y los cuadros delimitadores de las herramientas para cada una de las imágenes etiquetadas. Posteriormente, se guardan los datos de ambos datos en un archivo, esto se hace para no tener que repetir el proceso continuamente al realizar distintas pruebas, ya que resulta un proceso con un coste computacional alto.
El resultado de este código es un archivo con la red neuronal entrenada. Este archivo contiene las siguientes características:

- *Network*: Donde se incluyen las características de la red neuronal como son las capas (en este caso se tienen 75 capas), conexiones de la red (en este caso se tienen 84 conexiones distintas), los aprendibles de la red, los estados, el nombres de las entradas en la red y los nombres de las salidas de la red.
- *Anchor Boxes*: Donde se encuentran las cajas de anclaje obtenidas mediante el código.
- *ClassNames*: Donde se encuentran los nombres de las clases definidas, en este caso, Right_tool y Left_tool.
- *InputSize*: Es el tamaño de las red, definido en el código.
- *Learnables*: Son los aprendibles de la red, son los mismos que se encuentran en el apartado *Network*.
- *State*: Son los estados de la red, son los mismos que los que se encuentran en el apartado *Network*.

Pseudocódigo 4.1 Obtiene el archivo para el entrenamiento y test de la red

```
1: begin                                   ▷ Obtiene el archivo de la red neuronal entrenada
2:     Cargar los archivos de los vídeos etiquetados para el entrenamiento
3:     Crear una variable con todos los archivos de entrenamiento
4:     Cargar los archivos de los vídeos etiquetados para el test
5:     Crear crea una variable con todos los archivos de test
6:     Crear archivo con los datos de entrenamiento y test
7: end
```

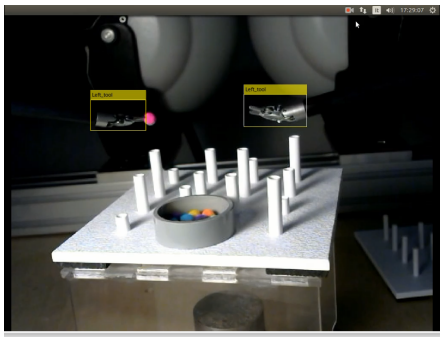
A continuación, se realiza una validación de los datos creados, para ello se realiza un bucle por cada conjunto de datos, comprobando que todas las imágenes tienen su correspondiente recuadro identificando cada una de las herramientas. Esto se hace tanto con el set de entrenamiento como con el set de test, el procedimiento se puede ver en el pseudocódigo 4.2.

Pseudocódigo 4.2 Validación de los datos

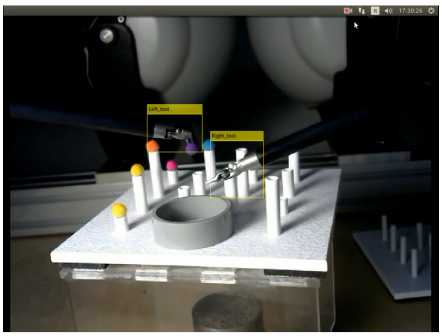
```
1: begin                                   ▷ Se chequea si los recuadros están anotados correctamente
2:     Se crea una variable que contendrá las imágenes
3:     Se crea una variable que contendrá los recuadros
4:     for i to length(imágenes) do
5:         Se muestran las imágenes con los recuadros delimitadores
6:     end for
7: end
```

Obteniendo imágenes como las que se muestran en las Figuras 4.10 y 4.11, donde se ven tanto las etiquetas como los recuadros que delimitan las zonas de las herramientas, por lo que se puede decir que los sets de datos están bien estructurados.

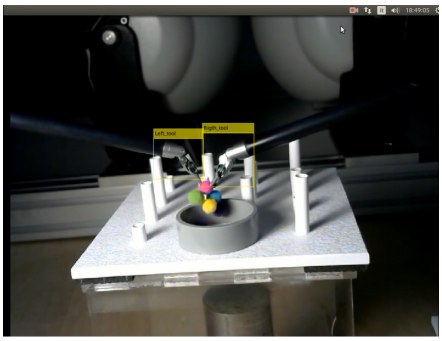
El siguiente paso sería configurar la red neuronal, los parámetros necesarios para configurar la red neuronal han sido definidos anteriormente en el apartado 4.5. Una vez definidos se obtienen las cajas de anclaje y el meanIOU. Se



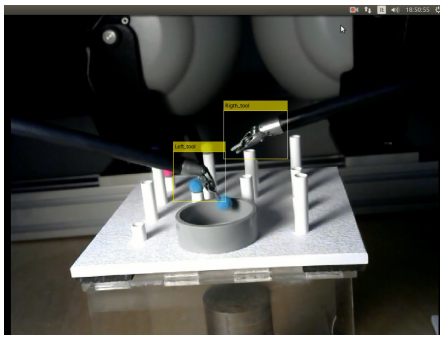
(a) Validación del vídeo 1 de test.



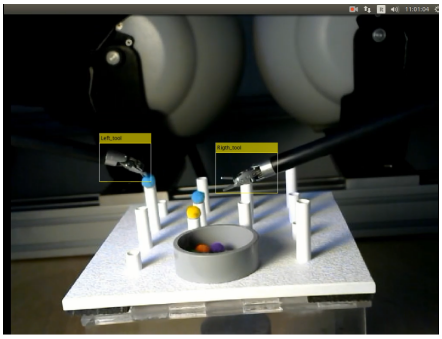
(b) Validación del vídeo 1 de test.



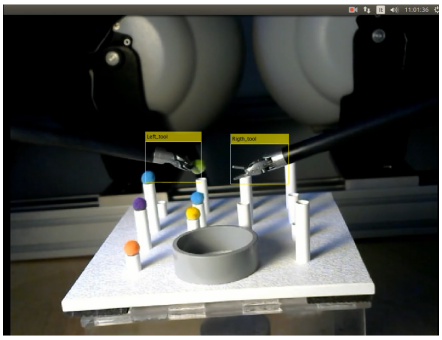
(c) Validación del vídeo 2 de test.



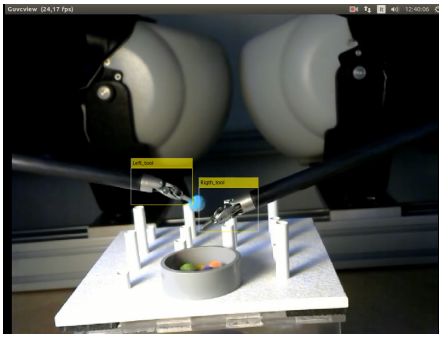
(d) Validación del vídeo 2 de test.



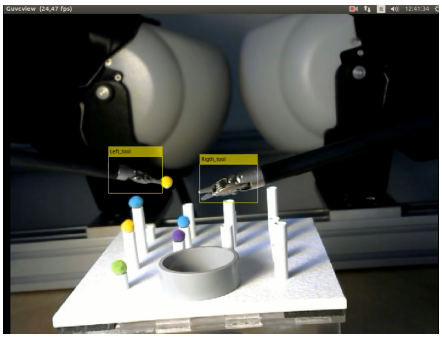
(e) Validación del vídeo 3 de test.



(f) Validación del vídeo 3 de test.

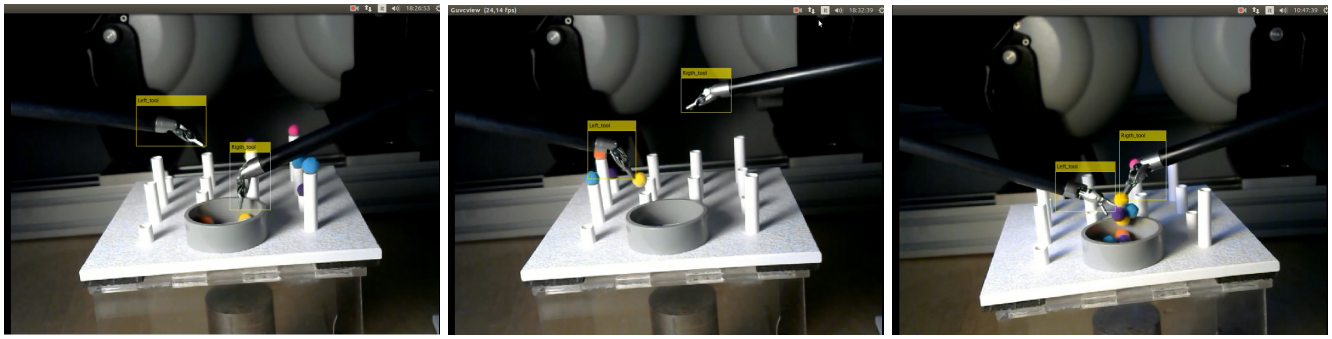


(g) Validación del vídeo 4 de test.

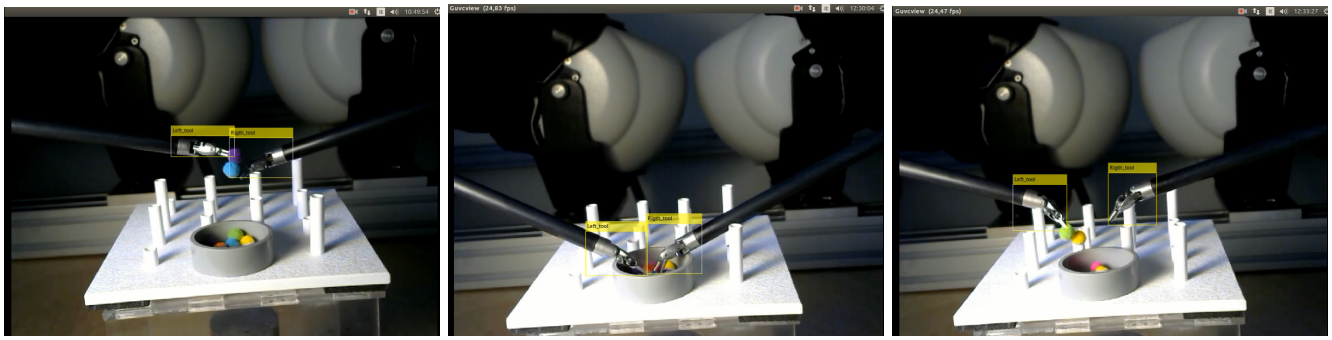


(h) Validación del vídeo 4 de test.

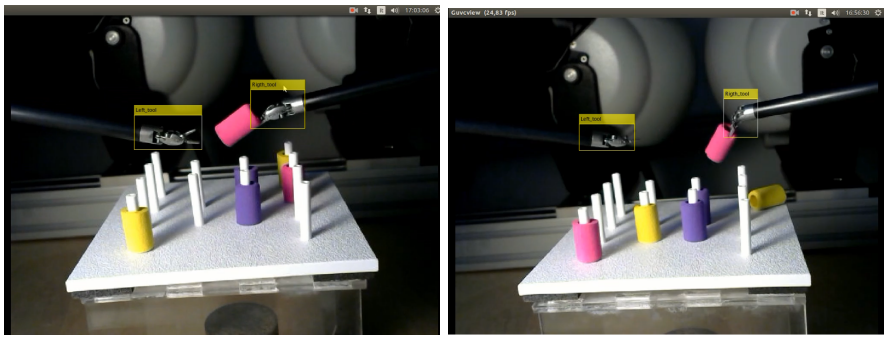
Figura 4.10: Validación del set de datos de test.



(a) Validación del vídeo 1 de entrenamiento. (b) Validación del vídeo 2 de entrenamiento. (c) Validación del vídeo 3 de entrenamiento.



(d) Validación del vídeo 4 de entrenamiento. (e) Validación del vídeo 5 de entrenamiento. (f) Validación del vídeo 6 de entrenamiento.



(g) Validación del vídeo 9 de entrenamiento. (h) Validación del vídeo 10 de entrenamiento.

Figura 4.11: Validación del set de datos de entrenamiento.

obtienes las cajas de anclaje que se van a usar en ambos cabezales de detección. Las cajas de anclaje son unas celdas de tamaño de [Mx1], donde M denota el número de cabezales de detección. Cada cabezal de detección consta de una matriz [Nx2], donde N es el número de anclajes que se van a utilizar. Se selecciona una caja de anclaje para cada cabezal de detección en función del tamaño del mapa de entidades. Normalmente se usan las cajas de anclaje más grandes a escala más baja y las más pequeñas a escala más alta. Para ello, se ordenan primero las cajas de anclaje de mayor a menor y se asignan las tres primeras al primer cabezal de detección y las tres últimas al segundo cabezal de detección.

El meanIoU calcula la relación entre el área donde se sobreponen 2 límites y el área donde se conectan. Estos dos cuadros delimitadores son el cuadro delimitador real y cuadro delimitador que se predice mediante la red. Se ha obtenido un meanIoU (Medida de calidad para evaluar las cajas de anclaje estimadas) de 0.8697, valor que siendo mayor a 0.5 garantiza que las cajas de anclaje se superpongan correctamente con las cajas de los datos de entrenamiento.

Posteriormente, es necesario pre-procesar los datos de entrenamiento, siguiendo lo descrito en [19], es decir, se escalan las imágenes de entrenamiento al tamaño de la red definido anteriormente para adaptarlas al tamaño de la red.

Normalmente se realizan también otras operaciones de pre-procesamiento como voltear la imagen o saturarla, para adaptarse a distintas situaciones de movimiento o iluminación. En este caso, al aparecer las herramientas izquierda y derecha siempre por el mismo lugar y tener unas condiciones de luminosidad más o menos estables entre imágenes se ha optado no aplicar *data augmentation* para pre-procesar los datos de entrenamiento, ya que puede ser contraproducente a la hora del entrenamiento.
En un inicio se intento aplicar *data augmentation*, pero generaba problemas, ya que confundí constantemente las herramientas izquierda y derecha al voltear las imágenes y aplicando distintas saturaciones tampoco ayudaba a la mejora del entrenamiento.

Una vez preprocesados los datos de entrenamiento, se especifican los parámetros de la red y una vez definidos los parámetros de entrada de entrenamiento y hecho el pre-procesamiento de los datos de entrenamiento, es hora de entrenar la red siguiendo el ejemplo [19].

Una vez preprocesados los datos, se procede al entrenamiento de la red neuronal. En primer lugar, se crea una figura que servirá para mostrar tanto la tasa de aprendizaje de la red como las pérdidas que la red está experimentando a medida que se entrena. A continuación, se inicia un bucle que se repetirá desde el inicio de las épocas hasta el número de épocas definidas previamente. Dentro de este bucle se realizan varias acciones importantes. En cada época, los datos se mezclan o aleatorizan, lo que se hace para mejorar el proceso de aprendizaje de la red, ya que presenta los datos de manera diferente en cada época.

En cada iteración del bucle interno, se procesa un lote de datos, lo que significa que se toma un conjunto de datos de entrenamiento y se lo pasa a la red para que la procese. Después de procesar el lote, se calculan los gradientes del modelo y las pérdidas, lo que es fundamental para que la red pueda ajustar sus parámetros y aprender de los datos. A continuación, se determina la tasa de aprendizaje actual, un valor importante para controlar cuánto deben actualizarse los parámetros en cada paso de entrenamiento.

Posteriormente, se actualizan los parámetros aprendibles del detector y se actualiza el estado del modelo. La información sobre la tasa de aprendizaje y las pérdidas se muestran y los gráficos correspondientes se actualizan. Finalmente, una vez que se ha completado el entrenamiento, la red neuronal entrenada se guarda en una variable que a su vez se guarda en un archivo externo para su uso posterior sin necesidad de volver a realizar este proceso, ya que tiene un coste computacional alto. Esta estructura esta descrita más gráficamente en el pseudocódigo 4.3.

4.6.3. Reconocimiento de herramientas

Una vez obtenida la red neuronal entrenada, se procede a crear una función que cree un archivo de salida que incluya la etiqueta, localización, puntuación y centro de las herramientas en cada imagen de un vídeo de entrada dada a la función. Para ello se ha procede a crear una estructura que contendrá los siguientes elementos:

- Label: Se guardarán las etiquetas de las herramientas, izquierda (Left_tool) y derecha (Right_tool), encontradas por la red en cada imagen del vídeo de entrada.

Pseudocódigo 4.3 Entrenamiento de la red

```
1: begin                                          ▷ Crea el archivo de la red neuronal entrenada
2:     Se crea una Figura para mostrar la tasa de aprendizaje y las pérdidas
3:     for i to número de épocas do
4:         Se aleatorizan los datos en cada época
5:         Se procesa un lote cada vez
6:         Se calculan los gradientes del modelo y las pérdidas
7:         Se determina la tasa de aprendizaje
8:         Se actualizan los parámetros de estado del modelo
9:         Se muestra la información sobre la pérdida y aprendizaje de cada iteración
10:        Se actualizan los gráficos correspondientes
11:    end for
12:    Una vez completado el entrenamiento, la red neuronal entrenada se almacena en una variable
13: end
```

- Location: Se guardarán las localizaciones (cuadros delimitadores) de las herramientas encontradas por la red en cada imagen del vídeo de entrada.
- Score: Se guardarán las puntuaciones de las herramientas encontradas por la red en cada imagen del vídeo de entrada. Estas puntuaciones son las puntuaciones de confianza, es decir la probabilidad de que el objeto, en este caso herramienta, detectada sea la correcta.
- Center: Se guardarán los centros de las localizaciones (cuadros delimitadores) de las herramientas encontradas por la red en cada imagen del vídeo de entrada.

Para obtener esa estructura de salida, se ha creado un código que contiene los pasos descritos en el pseudocódigo 4.4

Pseudocódigo 4.4 Función para obtener la estructura para las herramientas

```
1: begin                                          ▷ Función para obtener la estructura para etiquetar las herramientas
2:     Se crea una Figura para mostrar la tasa de aprendizaje y las pérdidas
3:     Se obtiene el número de fotogramas del vídeo de entrada
4:     Se crea la estructura de datos donde se almacenarán los datos deseados
5:     for i to número de imágenes do
6:         Se Lee el imagen actual del vídeo
7:         Se utiliza la Red entrenada para detectar las herramientas en el imagen actual, obteniendo los recuadros
       delimitadores, la puntuación y las etiquetas correspondientes.
8:         Se eliminan las detecciones duplicadas
9:         Se inicializan las matices para almacenar los datos de las herramientas detectadas en el imagen actual
10:        Se verifica que solo existe una herramienta de cada tipo por imagen
11:        Se calcula y almacena el centro de cada herramienta en la estructura
12:        Se almacena el resto de la información en la estructura
13:    end for
14: end
```

4.7. Resultados

En este apartado se describen los resultados obtenidos relativos a la detección de herramientas en vídeos de la base de datos ROSMA.

4.7.1. Entrenamiento de la red

En un inicio se comenzaron realizando pruebas con una cantidad distinta de imágenes a la mostrada en la Tabla 4.1, pero al no ser los resultados lo suficientemente buenos, se hizo un proceso iterativo modificando la cantidad de imágenes usados hasta alcanzan los resultados deseados. Las diferencias entre las tres versiones de la red neuronal

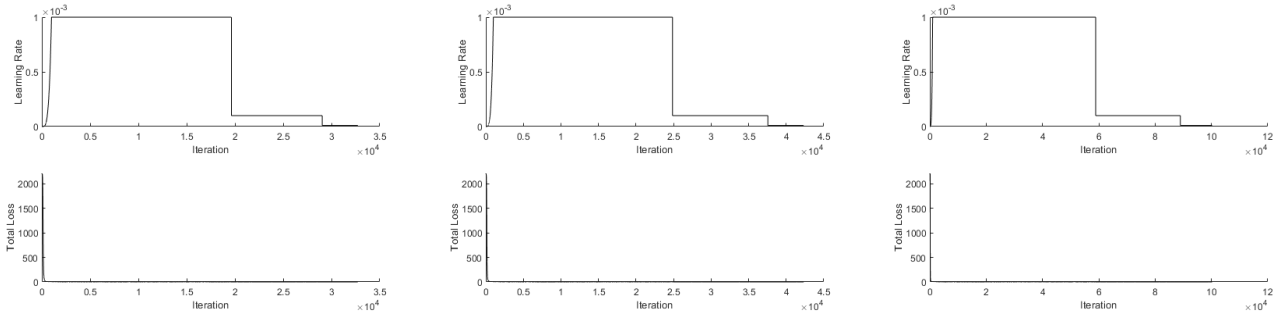
preentrenada se pueden ver en la Tabla 4.2.

Versión de red neuronal	Uso de los vídeos	Vídeos usados	Imágenes por vídeo	Imágenes totales
v01	**Entrenamiento**	X01 Pea on a peg 01	1845	**6510**
		X01 Pea on a peg 02	1530	
		X01 Pea on a peg 03	1740	
		X01 Pea on a peg 04	1395	
	Test	X01 Pea on a peg 05	1770	**3810**
		X01 Pea on a peg 06	2040	
v02	**Entrenamiento**	X01 Pea on a peg 01	1845	**14715**
		X01 Pea on a peg 02	1530	
		X01 Pea on a peg 03	1740	
		X01 Pea on a peg 04	1395	
		X01 Pea on a peg 05	1770	
		X01 Pea on a peg 06	2040	
		X02 Pea on a peg 01	2250	
		X02 Pea on a peg 02	2145	
	Test	X02 Pea on a peg 03	1725	**7395**
		X02 Pea on a peg 04	1740	
		X02 Pea on a peg 05	1605	
		X02 Pea on a peg 06	2325	
v03	**Entrenamiento**	X01 Pea on a peg 01	1845	**19965**
		X01 Pea on a peg 02	1530	
		X02 Pea on a peg 01	2250	
		X02 Pea on a peg 02	2145	
		X03 Pea on a peg 01	1905	
		X03 Pea on a peg 02	1680	
		X04 Pea on a peg 01	2880	
		X01 Post and sleeve 01	1905	
		X11 Post and sleeve 04	1980	
	Test	X01 Pea on a peg 06	2040	**8925**
		X02 Pea on a peg 06	2325	
		X03 Pea on a peg 06	2295	
		X04 Pea on a peg 04	2265	

Tabla 4.2: Diferencias entre las tres versiones de red neuronal preentrenada.

Las curvas mostradas en la Figura 4.12, en la parte superior representan la tasa de aprendizaje que se ha ido consiguiendo en cada uno de los tres entrenamientos de red. Mientras que las imágenes de la parte inferior representan las pérdidas en función del número de imágenes usadas en el proceso de entrenamiento.

En la Figura 4.12 se pueden ver como avanza el aprendizaje tras el primer entrenamiento con una cantidad de vídeos etiquetados inferior contra el segundo y último entrenamiento realizado, la cantidad de imágenes usadas en las tres versiones se pueden encontrar en la Tabla 4.2. En estas imágenes se puede visualizar como a mayor número de imágenes la tasa de aprendizaje se reduce en iteraciones mas tardías, es decir, a medida que se avanza en el entrenamiento de la red neuronal y se usa más ejemplos de datos (imágenes), se reduce la tasa de aprendizaje en cada iteración para ayudar a la red a converger de manera más precisa y estable hacia una solución óptima. Esto es importante para evitar problemas como el sobreajuste y garantizar que la red pueda generalizar bien a datos no vistos.



(a) Aprendizaje de la red v01. (b) Aprendizaje de la red v02. (c) Aprendizaje de la red v03.

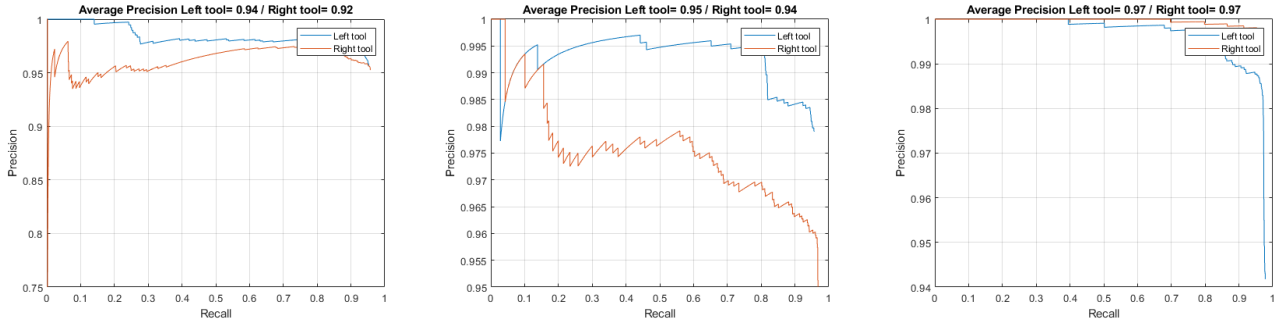
Figura 4.12: Aprendizaje de la red con distintas cantidades de vídeos.

En las Figura 4.13 se representa la precisión media que se obtiene para la detección de las herramientas que se ha

obtenido con las distintas redes neuronales entrenadas. En rojo se puede visualizar la precisión media de la herramienta derecha (Right_tool), mientras que en azul la de la herramienta izquierda (Left_tool).
En el eje X se representa el *recall* (también conocido como "sensibilidad.º "tasa de verdaderos positivos") en este contexto se refiere a una métrica que indica la proporción de objetos reales que se han identificado correctamente como positivos por el modelo de detección o clasificación. Mientras en el eje Y se gráfica la precisión, es decir, la proporción de objetos identificados como positivos por el modelo de detección que son verdaderos positivos.

Para obtener esta precisión se hace uso de las puntuaciones de confianza que se obtienen de la red YOLO. Se puede observar como con el aumento de imágenes etiquetadas en el entrenamiento y test va mejorando la precisión hasta llegar al último entrenamiento realizado con 10 vídeos de entrenamiento con una cantidad total de 19965 imágenes etiquetadas.
Como se puede ver en la Figura 4.13 con la versión v01 se tiene una precisión media para las herramientas derecha e izquierda de un 0.94 y 0.92 respectivamente. Con la v02 se tiene una precisión media de un 0.95 y 0.94 y por último con la v03 se obtiene 0.97 en ambas herramientas.

Con las tres versiones la red funciona bien para la detección de las herramientas, en las primeras versiones, había algunas imágenes en las que no se detectaban alguna de las dos herramientas, sin embargo, con el aumento de imágenes de entrenamiento de la última versión y la consecuente mejora de precisión las herramientas se detectan en prácticamente todas las imágenes de forma eficiente como se vera más adelante en esta memoria. Por lo que se puede decir que la red configurada y entrenada funciona correctamente.



(a) Aprendizaje de la red v01. (b) Aprendizaje de la red v02. (c) Aprendizaje de la red v03.

Figura 4.13: Precisión de detección de la red con distintas cantidades de vídeos.

En definitiva, aunque en el primer entrenamiento se obtiene una precisión de 0.92, que se puede considerar una precisión adecuada, a la hora de etiquetar los vídeos de la tarea *Post and sleeve* había una gran cantidad de herramientas que no se identificaban, mientras que conforme se aumentaban el número de imágenes etiquetados la detección de herramientas mejoraba considerablemente. Finalmente, con la versión v03, donde se tiene una precisión del 0.97 en ambas herramientas, se detectan las herramientas prácticamente en todos los imágenes sin problema, con un porcentaje de acierto muy alto.

4.7.2. Detección de herramientas

Una vez entrenada la red, se procede al etiquetado de las herramientas en los distintos vídeos, obteniendo unos resultados de etiquetado de herramienta bastante aceptables. En la Tabla 4.3 se puede ver la precisión obtenidas en el etiquetado de ambas herramientas para cada vídeo del experimento Post And Sleeve. Estas puntuaciones son las puntuaciones de confianza que se obtienen de la red YOLO, es decir, la probabilidad de que el objeto identificado en esa celda sea el mismo que se ha etiquetado.

Vídeo etiquetado	Precisión media herramienta izquierda	Precisión media herramienta derecha
X01 Post And Sleeve 01	1.00	1.00
X01 Post And Sleeve 02	1.00	1.00
X01 Post And Sleeve 03	1.00	1.00

X01 Post And Sleeve 04	1.00	0.99
X01 Post And Sleeve 05	0.99	1.00
X01 Post And Sleeve 06	1.00	0.98
X02 Post And Sleeve 01	1.00	1.00
X02 Post And Sleeve 02	1.00	1.00
X02 Post And Sleeve 03	1.00	1.00
X02 Post And Sleeve 04	1.00	1.00
X02 Post And Sleeve 05	1.00	1.00
X02 Post And Sleeve 06	0.99	0.99
X03 Post And Sleeve 01	0.99	0.98
X03 Post And Sleeve 02	1.00	0.96
X03 Post And Sleeve 03	1.00	0.99
X03 Post And Sleeve 04	1.00	0.99
X03 Post And Sleeve 05	1.00	1.00
X04 Post And Sleeve 01	1.00	1.00
X04 Post And Sleeve 02	1.00	1.00
X04 Post And Sleeve 03	0.99	1.00
X04 Post And Sleeve 04	1.00	1.00
X04 Post And Sleeve 05	1.00	1.00
X05 Post And Sleeve 01	0.99	0.99
X05 Post And Sleeve 02	0.99	0.98
X05 Post And Sleeve 03	0.99	0.99
X05 Post And Sleeve 04	0.99	0.98
X05 Post And Sleeve 05	0.98	0.99
X06 Post And Sleeve 01	1.00	1.00
X06 Post And Sleeve 02	0.99	1.00
X06 Post And Sleeve 03	1.00	1.00
X06 Post And Sleeve 04	1.00	0.99
X06 Post And Sleeve 05	1.00	0.99
X06 Post And Sleeve 06	0.99	1.00
X07 Post And Sleeve 01	0.98	0.99
X07 Post And Sleeve 02	0.99	1.00
X07 Post And Sleeve 03	1.00	1.00
X07 Post And Sleeve 04	1.00	0.99
X07 Post And Sleeve 05	1.00	1.00
X07 Post And Sleeve 06	1.00	0.99
X08 Post And Sleeve 01	0.99	1.00
X08 Post And Sleeve 02	0.98	1.00
X08 Post And Sleeve 03	0.98	1.00
X08 Post And Sleeve 04	0.98	1.00
X08 Post And Sleeve 05	0.98	1.00
X08 Post And Sleeve 06	0.99	1.00
X09 Post And Sleeve 01	0.99	1.00
X09 Post And Sleeve 02	0.99	0.99
X09 Post And Sleeve 03	0.99	0.99
X09 Post And Sleeve 04	1.00	1.00
X10 Post And Sleeve 01	1.00	1.00
X10 Post And Sleeve 02	0.99	0.99
X10 Post And Sleeve 03	0.99	1.00
X10 Post And Sleeve 04	0.98	0.99
X10 Post And Sleeve 05	0.99	1.00
X10 Post And Sleeve 06	1.00	1.00
X11 Post And Sleeve 01	1.00	1.00
X11 Post And Sleeve 02	1.00	1.00

X11 Post And Sleeve 03	1.00	0.99
X11 Post And Sleeve 04	1.00	1.00
X11 Post And Sleeve 05	1.00	1.00
X11 Post And Sleeve 06	1.00	1.00
X12 Post And Sleeve 01	0.99	0.98
X12 Post And Sleeve 02	0.98	0.99
X12 Post And Sleeve 03	1.00	0.99
X12 Post And Sleeve 04	0.95	0.99
Media	0,99	0,99

Tabla 4.3: Precisión media del etiquetado de las herramientas.

Como se puede ver se obtiene una precision media de 0.99 para ambas herramientas. Esto quiere decir que solo en el 1 % de las imágenes se han etiquetado incorrectamente las herramientas (Right_tool y Left_tool).

5. Evaluación automática de la tarea de 'Post and Sleeve´

5.1. Introducción

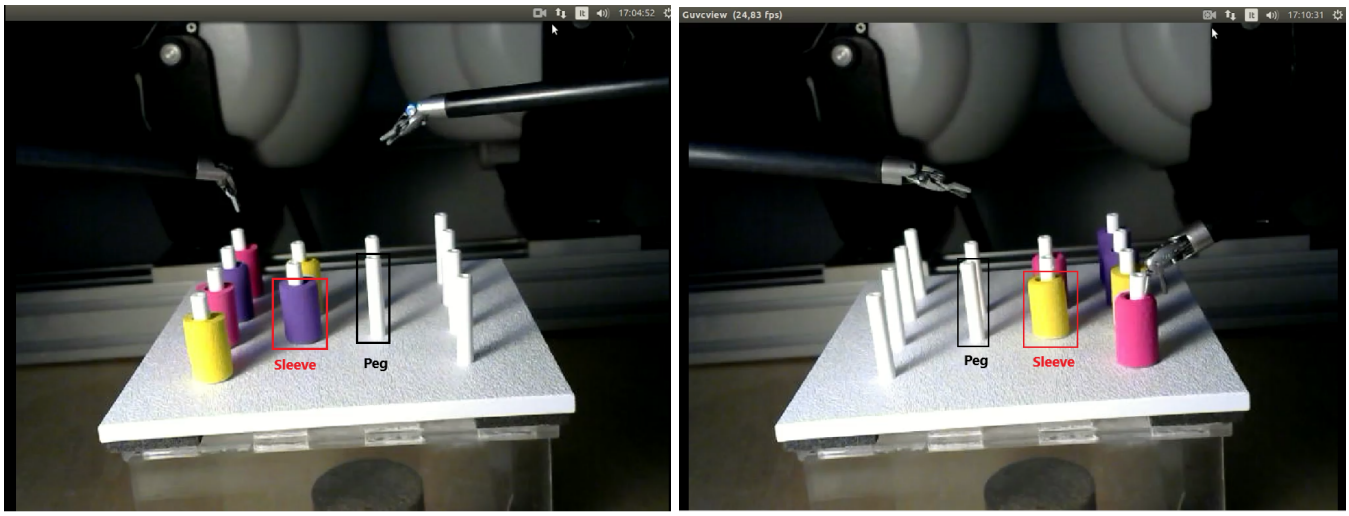
La aparición de la cirugía mínimamente invasiva y de cirugía asistida por sistemas robóticas presenta múltiples ventajas tanto para los pacientes como para el personal médico. Sin embargo, para los cirujanos y cirujanas conlleva un largo proceso de entrenamiento, para aprender el manejo de estos instrumentos. Por tanto, una de las tareas esenciales en la formación médica es la evaluación de las habilidades quirúrgicas para calificar el desempeño de los cirujanos y monitorear su desarrollo durante el proceso de formación. Esta evaluación generalmente se realiza de manera manual por cirujanos expertos, lo que colleva los siguientes problemas:

- Los cirujanos expertos deben emplear una gran cantidad de tiempo en supervisar y evaluar a los estudiantes.
- La evaluación de las maniobras es muy subjetiva, ya que se basa principalmente en la percepción del evaluador. Por tanto, la evaluación depende en gran medida de los criterios de cada cirujano/a.

Para resolver estos problemas, muchos autores están desarrollando métodos para evaluar los procesos de entrenamiento de los cirujanos noveles de forma automática. En la base de datos ROSMA, cada experimento tiene una puntuación, asociado al desempeño de la tarea. Esta puntuación se ha realizado manualmente, contando el número de errores y el tiempo empleado en realizar la maniobra para cada uno de los 207 experimentos que tiene la base de datos. Uno de los objetivos de este trabajo es realizar un algoritmo de asignación automática de la puntuación para una de las tres maniobras incluidas en la base de datos. Se ha decidido realizar esta evaluación automática sobre la tarea "Post and Sleeve", ya que los objetos utilizados durante esta tarea son fácilmente identificables con técnicas de visión convencionales.

5.2. Protocolo de la maniobra *Post and Sleeve*

En la Figura 5.1 se pueden ver las dos posibilidades de situación inicial del experimento. La mesa de experimentación consta de doce cilindros o *pegs* y 6 objetos de colores o *sleeves*, en la Figura 5.1 se ha identificado un cilindro de la mesa señalado con un recuadro en negro y un objeto de color señalado mediante un recuadro en rojo.



(a) Situación inicial del experimento con los objetos a la izquierda. (b) Situación inicial del experimento con los objetos a la derecha.

Figura 5.1: Situaciones inicial del experimento.

El protocolo de la maniobra realizada en el experimento *Post and Sleeve* obtenido de [13], tiene como objetivo mover los objetos de color de un lado a otro de la mesa de pruebas. En la posición inicial se encuentra la mesa situada con las columnas de los cilindros en posición vertical (de izquierda a derecha: 4-2-2-4). Los seis objetos estarán posicionados en uno de los lados de la mesa.
El usuario tenía que coger un objeto con una herramienta, pasarla a la otra herramienta y colocar el objeto en un tetón del lado opuesto de la mesa. Si un objeto se cae, es considerado una penalización y no puede recogerse de nuevo. Esta mecánica se repite en seis ocasiones: tres de derecha a izquierda y otros tres de izquierda a derecha.

Por cada penalización se tienen 15 puntos de penalización y la puntuación fina se obtiene sumando el tiempo en segundos más los puntos de penalización.

5.3. Sistema de inferencia

5.3.1. Introducción

Tal y como se ha descrito anteriormente, la puntuación de cada experimento se realiza sumando al tiempo empleado en completar la tarea una penalización en función del número de errores cometidos. Por tanto, para realizar la evaluación automática de la tarea es necesario identificar lo siguiente:

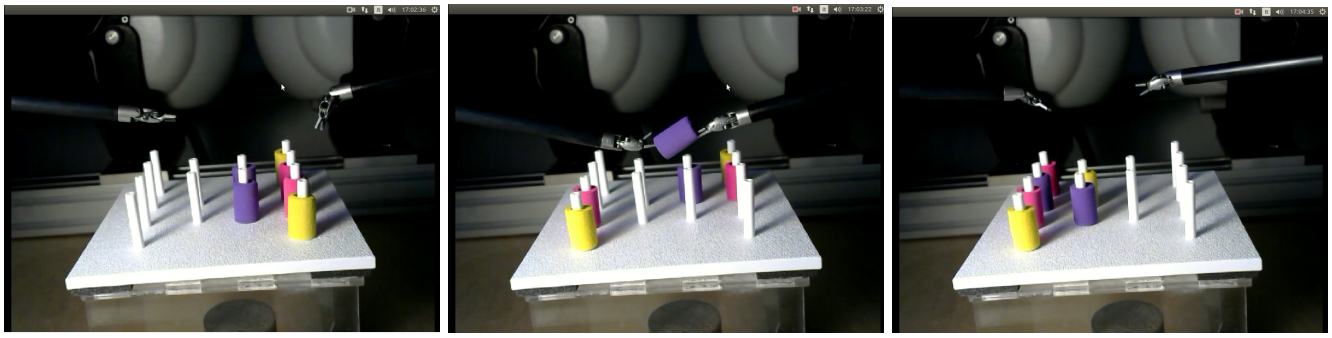
- Detección del inicio y fin de la tarea: para poder contabilizar de forma automática el tiempo empleado en realizar la maniobra, es necesario detectar el inicio y fin de la misma.
- Detección de errores: para poder restar las penalizaciones que conllevan los errores cometidos durante la tarea, es necesario identificar cuántos errores se han cometido al finalizar la maniobra.

5.3.2. Detección de inicio y fin de la tarea

Para poder calcular el tiempo empleado en realizar la maniobra, es necesario detectar el instante de inicio y fin de la tarea. Para ello, se hace uso del diagrama de estados representado en la Figura 5.3, en la que se muestran los estados por los que se pasará durante la evaluación del experimento.

- Estado inicial: En este estado se permanecerá mientras que ninguna de las herramientas se mueva, una vez comiencen a desplazarse se pasará al estado maniobra. Mientras el experimento se encuentre en este estado el tiempo de experimento permanecerá a 0.
- Estado maniobra: Una vez alguna de las herramientas se desplace se entrará en este estado, donde se realizará el traspaso de los objetos de colores de un lado de la mesa de trabajo a otro. Mientras el experimento se encuentre en este estado el tiempo de experimento irá aumentando. El tiempo aumentará hasta que no se detecte que todos los objetos se han desplazado de la zona inicial a la contraria. Una vez se detecte esta condición se pasará al estado fin.
- Estado fin: Se llega a este estado fin cuando se ha detectado que el experimento ha finalizado.

En la Figura 5.2 se puede ver un imagen del vídeo X01 Post and Sleeve 01 en cada estado, es decir, estado inicial, maniobra y fin.



(a) Estado inicial. (b) Estado maniobra. (c) Estado fin.

Figura 5.2: imagen en cada estado del vídeo X01 Post and Sleeve 01.

En la Figura 5.3 se puede ver un diagrama de estados donde se resume la transición entre estados.

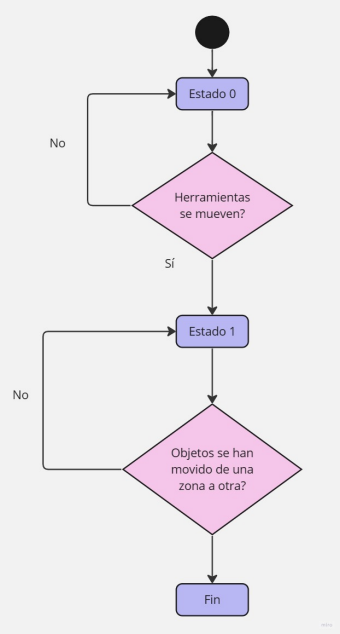
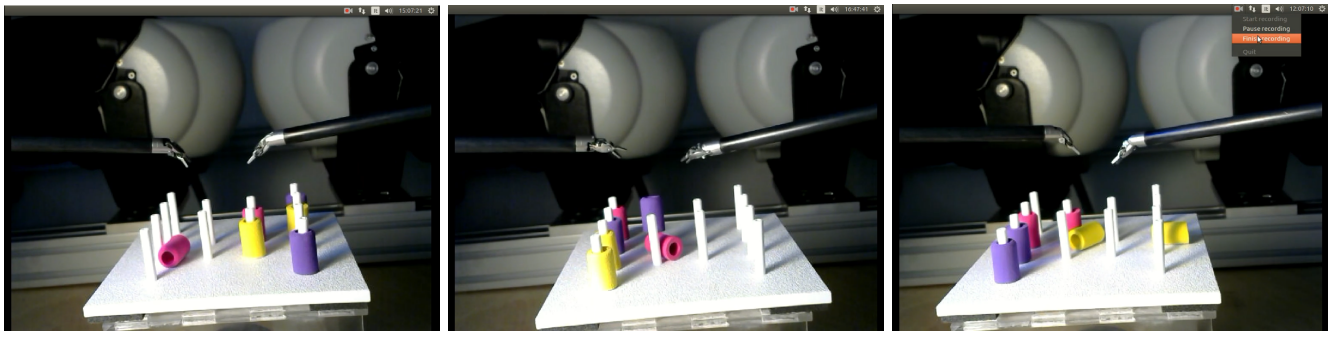


Figura 5.3: Diagrama de estados.

5.3.3. Detección de errores

Para la detección de errores se ha optado por usar la orientación de los objetos en el imagen final, en caso de que haya algún objeto de manera horizontal se contará como un error.
Debido a que se pueden caer objetos en ambas zonas de la mesa, se ha decidido obviar los objetos con orientación horizontal que se encuentren en la parte donde estaban los objetos inicialmente para determinar el tiempo de experimento. De esta manera se pueden detectar los errores independientemente de la zona, y en función de la zona en la que se encuentren tenerlos o no en cuenta para dictaminar el tiempo de experimento total.

En la Figura 5.4, se pueden visualizar objetos horizontales en la zona inicial donde se encontraban los objetos de colores, en la zona final y en ambas zonas, y que contarán como errores con nuestro algoritmo.



(a) Error en zona inicial. (b) Error en zona final. (c) Error en ambas zonas.

Figura 5.4: Objetos horizontales que contarán como error en distintas zonas.

El problema de este tipo de detección de errores es que los objetos que se queden ocultos tras otros objetos o que se caigan de la mesa de trabajo no son detectados como errores. Pero aún así se obtienen resultados aceptables con este tipo de detección.

5.4. Implementación

En esta sección se procederá con la descripción del código usado para evaluar datos de todos los vídeos de los que se van a identificar cuanto tarda el experimento y cuantos errores se han tenido. Además, se detallará el algoritmo de

segmentación de colores utilizado para detectar objetos en la imagen, y se analizarán los resultados obtenidos.

5.4.1. Código implementado

La descripción del código se puede ver en el pseudocódigo 5.1. En primer lugar, se inicia el proceso obteniendo el número total de imágenes presentes en el vídeo y se establece una estructura de datos diseñada para almacenar los errores detectados durante el experimento, junto con el tiempo transcurrido.
A continuación, se establece la tasa de imágenes, que en este caso se fija en 15 imágenes por segundo para el vídeo. Luego, se inicia un bucle que se repetirá desde la primera imagen hasta el número total de imágenes del vídeo.

Dentro de este bucle, se realizan varias acciones fundamentales. Si la imagen actual corresponde a la primera, se determina la zona inicial en la que se encuentran los objetos. Si, por otro lado, la imagen actual es posterior a la segunda, se verifica la estabilidad de las etiquetas de las herramientas, asegurándose de que no haya cambios bruscos en las etiquetas o que las herramientas no estén etiquetadas de manera inconsistente durante un período determinado de imágenes.

Cuando la imagen actual supera el número 10, se inicia un proceso para actualizar los valores de los centros de herramientas y objetos cada 10 imágenes. En caso de que alguna herramienta haya experimentado un movimiento, se cambia al estado de "maniobra". En el estado de "maniobra", se verifica la cantidad de objetos en la zona opuesta a donde inicialmente se encontraban (ya sea izquierda o derecha). Además, se examina si hay objetos horizontales en la imagen más reciente, y si se detectan, se registra un error.

Si los objetos no han cambiado de la zona inicial a la final o si las herramientas siguen en movimiento, se mantiene el estado de "maniobraz se actualiza el tiempo transcurrido durante el experimento.
En cada iteración del bucle, se almacenan tanto los errores detectados como el tiempo actual del experimento. En resumen, este proceso permite analizar un vídeo en busca de cambios en las etiquetas de herramientas y objetos, detectar movimientos y errores durante el experimento y registrar estos datos para su posterior análisis.

5.4.2. Identificación de los objetos en la imagen

Para identificar los objetos de colores en las imágenes se va a hacer uso de técnicas de segmentación de colores. La segmentación por colores es una técnica utilizada en el procesamiento de imágenes para identificar y aislar regiones de interés basadas en sus propiedades cromáticas. El objetivo es separar los píxeles o grupos de píxeles de una imagen en diferentes categorías de colores. Antes de realizar la segmentación, se suelen aplicar técnicas de preprocesamiento a la imagen para mejorar la calidad y facilitar la detección de los colores. Esto puede incluir operaciones como suavizado, corrección de iluminación, filtrado de ruido, entre otros.

Posteriormente, la imagen se transforma a un espacio de color adecuado para la segmentación por colores. Algunos espacios de color populares para esta técnica son RGB (rojo, verde, azul) [27], HSV (matiz, saturación, valor) [26] y LAB (luminosidad, componente a, componente b) (Figura 5.5). Cada espacio de color tiene características únicas que facilitan la extracción y segmentación de los colores.

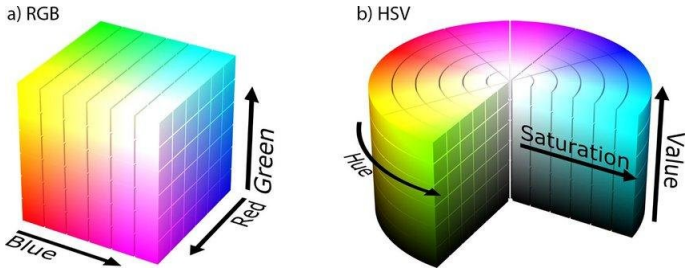
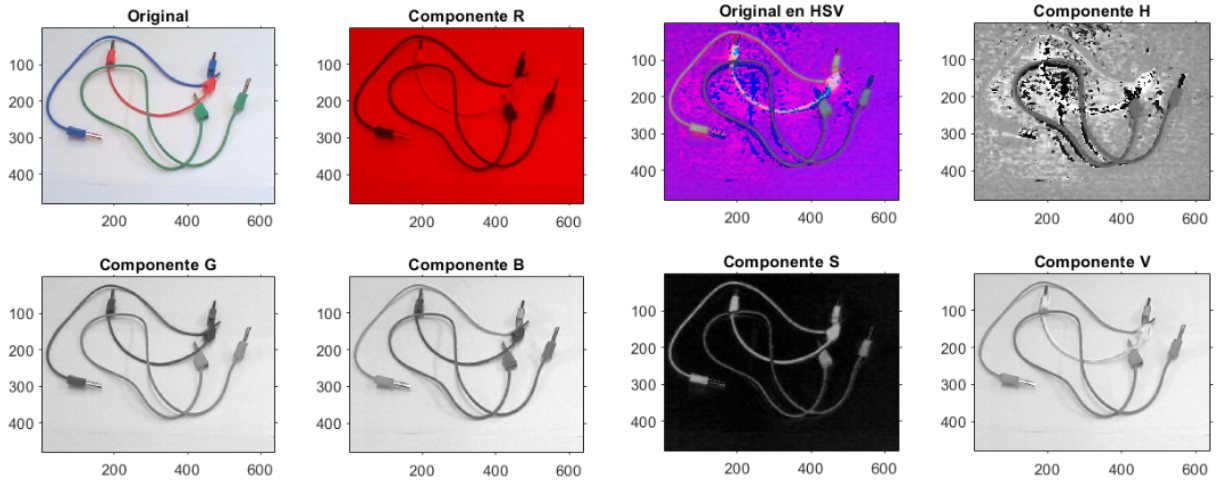


Figura 5.5: Esquema de color RGB (a) y HSV (b).

Después, se aplica un umbral a la imagen en el espacio de color seleccionado para separar los píxeles en diferentes categorías de colores (Figura 5.6). El umbral puede ser fijo o adaptativo según las características específicas de la imagen y los colores que se deseen segmentar [26] [27].

Pseudocódigo 5.1 Función para automatizar la evaluación y detectar los errores de todos los vídeos

```
Se obtiene el número de imágenes del vídeo
Se crea la estructura de datos donde se almacenarán los datos deseados
Se define el imagenrate
Se definen distintas variables y constantes
for 1 to numimágenes do
    Obtener los centros y orientaciones de los objetos de color en la imagen actual
    Obtener los centros de las herramientas en la imagen actual
    if imagenIdx = 1 then
        Calcular en que zona del imagen estan los objetos al inicio
    end if
    if imagenIdx>2 then
        Verificar estabilidad de etiquetas de herramientas
    end if
    if imagenIdx>10 then
        Actualizar historial de centros de herramientas y objetos
        if Alguna herramienta se ha movido then
            Se pasa al estado 1
        end if
        if Estado igual a 1 then
            if Objetos estaban a la izquierda al inicio then
                Comprobar cuantos objetos están en la derecha
            end if
            if Objetos estaban a la derecha al inicio then
                Comprobar cuantos objetos están en la izquierda
            end if
        end if
        if Algún objeto en el último imagen esta horizontal then
            Se suma un error
        end if
        if Los objetos (no horizontales) no se han movido de un lado a otro o las herramientas se mueven then
            Se actualiza el tiempo del experimento
        end if
    end if
    Almacenar errores y tiempo del experimento
end for
```



(a) Componentes RGB

(b) Componentes HSV

Figura 5.6: Imagen separada en componentes RGB y HSV.

Una vez realizada la umbralización, se pueden aplicar técnicas adicionales para eliminar el ruido y refinar la segmentación. Esto puede incluir operaciones de filtrado, operaciones morfológicas como erosión y dilatación, y técnicas de conectividad para agrupar regiones de píxeles contiguos [26] [27].

Después de segmentar los colores de interés, se puede proceder al etiquetado de las regiones segmentadas y analizar las características adicionales. Esto puede incluir el cálculo de la forma, el tamaño, la ubicación o cualquier otra propiedad de las regiones segmentadas relevante para cada aplicación.

Para la segmentación por colores se ha decidido crear una estructura de forma similar a la función del etiquetado mediante el uso de la red neuronal mostrado en el apartado anterior. En este caso se ha optado por una estructura formada por los siguientes apartados:

- Label: Se guardarán las etiquetas (colores) de los objetos detectados mediante la segmentación de colores en el vídeo de entrada.
- Location: Se guardarán las localizaciones (cuadros delimitadores) de los objetos detectados mediante la segmentación de colores en el vídeo de entrada.
- Orientation: Se guardarán las orientaciones (cuadros delimitadores) de los objetos detectados mediante la segmentación de colores en el vídeo de entrada.
- Center: Se guardarán los centros de las localizaciones (cuadros delimitadores) de los objetos detectados mediante la segmentación de colores en el vídeo de entrada.

Al solo disponer objetos de color rosa, morado y amarillo se han definido tres umbrales HSV de colores que detecten correctamente los objetos de los vídeos, estos rangos se definirán en la descripción del código.
Para obtener la estructura de salida deseada, se ha decidido crear una función que tiene la estructura descrita en 5.2.

Esta función comienza por establecer una estructura de datos diseñada para almacenar información crucial sobre los objetos detectados. Dentro de esta estructura se incluyen detalles esenciales como el color, la ubicación, el centro, la orientación y el área de todos los objetos que se identifiquen en las imágenes.

A continuación, se procede a definir rangos de colores en el espacio de color HSV para tres posibles colores de objetos: amarillo, morado y rosa. Estos rangos de color son esenciales para la identificación de objetos en las imágenes y se configuran específicamente en términos de los valores de matiz, saturación y valor en el espacio de color HSV.

El proceso continúa mientras haya imágenes por etiquetar, en cada iteración, se selecciona una imagen que se convierte al formato LAB. Esto se hace para extraer la componente de luminosidad (L) de la imagen. Se crea una máscara adaptativa en el canal L con una sensibilidad predeterminada de 0.9.

Luego, la imagen se convierte al formato HSV, y se genera una máscara individual para cada uno de los colores definidos anteriormente. Estas máscaras se emplean para identificar y etiquetar los objetos que coinciden con los colores específicos en la imagen.
Para refinar los resultados y mejorar la precisión de la detección, se aplican operaciones morfológicas en las máscaras. Esto tiene como objetivo eliminar ruidos pequeños que puedan afectar la calidad de la detección.

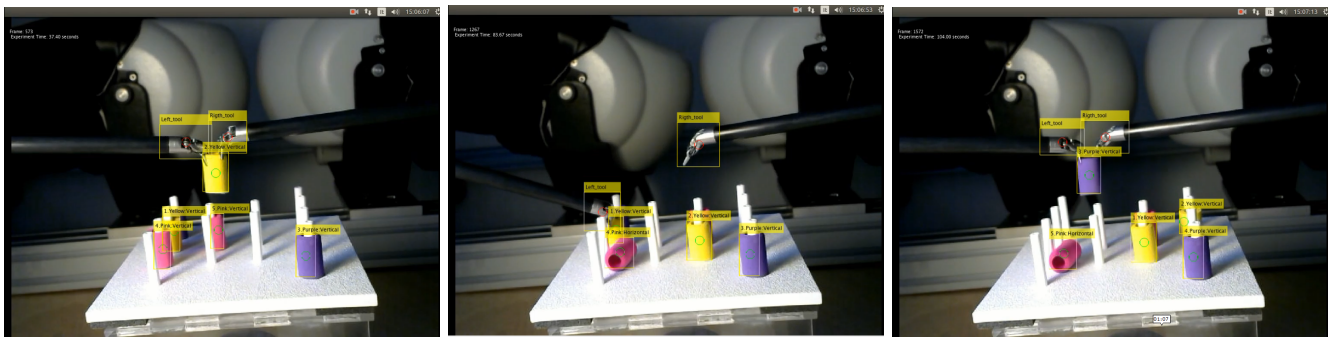
Las propiedades de los objetos etiquetados en cada máscara se recolectan y combinan en un conjunto de datos unificado. Para evitar confusiones y garantizar una detección precisa, se eliminan los objetos que se encuentren en las áreas superior e inferior de la imagen y que no sean relevantes para el análisis.
Luego, se inicia un bucle que recorre los objetos detectados en cada imagen. Durante este proceso, se determina el color, la ubicación, el centro, la orientación y el área de cada objeto.

Finalmente, todos los datos y detalles recopilados se almacenan en la estructura de datos que se estableció al principio del proceso. En resumen, este proceso permite detectar y caracterizar objetos en imágenes, identificando su color, ubicación, orientación y otras propiedades relevantes.

Pseudocódigo 5.2 Función para obtener la estructura para los objetos

```
1: begin                                  ▷ Función para obtener la estructura para etiquetar los objetos
2:     Se crea la estructura de datos donde se almacenarán los datos deseados
3:     Se definen los rangos de colores para identificar los objetos
4:     while Halla imágenes por etiquetar do
5:         Se lee el imagen a etiquetar
6:         Se convierte el imagen a formato LAB
7:         Se extrae la componente L(Luminosidad)
8:         Se crea una máscara adaptativa en el canal L
9:         Se convierte el imagen a formato HSV
10:        Se crea una máscara por cada color
11:        Se aplican operaciones morfológicas para eliminar ruidos pequeños en las máscaras
12:        Se obtienen las propiedades de los objetos etiquetados por cada color
13:        Se combinan las etiquetas de todos los colores
14:        Se filtran los objetos que se encuentran en la zona inferior y superior del imagen para evitar errores
15:        Se eliminan los objetos con un área menor a 500 píxeles
16:        Se inicializan las variables para almacenar etiquetas, recuadros, orientaciones, centros y áreas
17:        Se determina el color de cada objeto usando una función auxiliar
18:        Se almacenan los datos de etiquetas, recuadros, orientaciones, centros y áreas en las variables temporales
19:        Se almacenan los datos obtenidos en la estructura de salida
20:        Se incrementa el índice de imagen para la siguiente iteración
21:    end while
22: end
```

Mediante la función definida, se realiza una identificación de los distintos objetos de colores que se encuentran en el experimento de una forma bastante eficiente. En la Figura 5.7 se pueden visualizar objetos etiquetados en distintos momentos del vídeo X01 Post and sleeve 04.



(a) imagen etiquetado 573. (b) imagen etiquetado 1267. (c) imagen etiquetado 1572.

Figura 5.7: Objetos etiquetados en el vídeo X06 Post and sleeve 04.

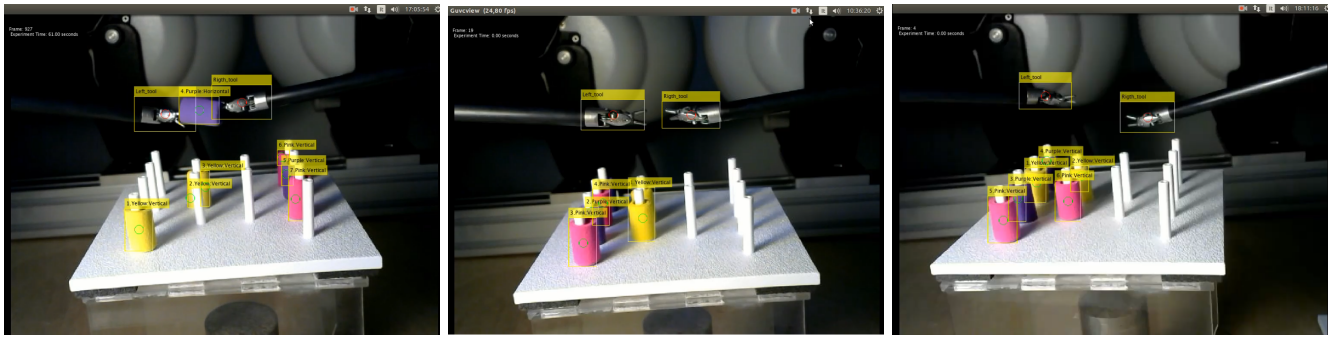
5.5. Resultados

5.5.1. Detección de objetos

En cuanto a la detección de objetos, se han obtenido los porcentajes de acierto que se pueden ver en la Tabla 5.1. Estos porcentajes a veces son algo bajos, ya que en las situaciones que los objetos del mismo color están muy próximos entre sí se detectan como uno solo. En el momento que los objetos se separan la detección mejora considerablemente. También hay casos en los que un único objeto se detecta como dos, debido a que el objeto queda dividido visualmente por uno de los cilindros de la mesa de experimento, debido a estas casuísticas hay porcentajes algo más bajos y otros que superan el 100 %.

En este caso se ha calculado la puntuación de cada vídeo dividiendo el número de objetos detectados en cada imagen por el número de objetos que debe haber en todo momento, es decir 6. Posteriormente, se ha obtenido el

porcentaje de cada vídeo, haciendo la media de los porcentajes de cada imagen.
En la Figura 5.8 se pueden visualizar las casuísticas descritas anteriormente por los que el porcentaje de acierto varía entre un gran rango de valores.



(a) Objeto dividido por un cilindro. (b) Objetos detectados como uno solo. (c) Objetos completamente detectados.

Figura 5.8: Distintas situaciones que varían el porcentaje de acierto al detectar objetos.

Vídeo etiquetado	Porcentaje de acierto
X01 Post And Sleeve 01	112 %
X01 Post And Sleeve 02	104 %
X01 Post And Sleeve 03	91 %
X01 Post And Sleeve 04	99 %
X01 Post And Sleeve 05	93 %
X01 Post And Sleeve 06	102 %
X02 Post And Sleeve 01	117 %
X02 Post And Sleeve 02	115 %
X02 Post And Sleeve 03	92 %
X02 Post And Sleeve 04	107 %
X02 Post And Sleeve 05	98 %
X02 Post And Sleeve 06	112 %
X03 Post And Sleeve 01	101 %
X03 Post And Sleeve 02	98 %
X03 Post And Sleeve 03	100 %
X03 Post And Sleeve 04	106 %
X03 Post And Sleeve 05	110 %
X04 Post And Sleeve 01	77 %
X04 Post And Sleeve 02	92 %
X04 Post And Sleeve 03	79 %
X04 Post And Sleeve 04	96 %
X04 Post And Sleeve 05	82 %
X05 Post And Sleeve 01	92 %
X05 Post And Sleeve 02	88 %
X05 Post And Sleeve 03	71 %
X05 Post And Sleeve 04	78 %
X05 Post And Sleeve 05	74 %
X06 Post And Sleeve 01	68 %
X06 Post And Sleeve 02	95 %
X06 Post And Sleeve 03	81 %
X06 Post And Sleeve 04	92 %
X06 Post And Sleeve 05	76 %
X06 Post And Sleeve 06	106 %

X07 Post And Sleeve 01	65 %
X07 Post And Sleeve 02	76 %
X07 Post And Sleeve 03	77 %
X07 Post And Sleeve 04	81 %
X07 Post And Sleeve 05	91 %
X07 Post And Sleeve 06	81 %
X08 Post And Sleeve 01	60 %
X08 Post And Sleeve 02	56 %
X08 Post And Sleeve 03	62 %
X08 Post And Sleeve 04	47 %
X08 Post And Sleeve 05	59 %
X08 Post And Sleeve 06	50 %
X09 Post And Sleeve 01	68 %
X09 Post And Sleeve 02	89 %
X09 Post And Sleeve 03	70 %
X09 Post And Sleeve 04	71 %
X10 Post And Sleeve 01	78 %
X10 Post And Sleeve 02	85 %
X10 Post And Sleeve 03	86 %
X10 Post And Sleeve 04	83 %
X10 Post And Sleeve 05	81 %
X10 Post And Sleeve 06	84 %
X11 Post And Sleeve 01	70 %
X11 Post And Sleeve 02	92 %
X11 Post And Sleeve 03	86 %
X11 Post And Sleeve 04	79 %
X11 Post And Sleeve 05	90 %
X11 Post And Sleeve 06	88 %
X12 Post And Sleeve 01	98 %
X12 Post And Sleeve 02	90 %
X12 Post And Sleeve 03	96 %
X12 Post And Sleeve 04	105 %
Media	86.12 %

Tabla 5.1: Porcentaje de acierto del etiquetado de los objetos.

5.5.2. Obtención automática de la puntuación de cada maniobra

Para comparar los resultados obtenidos se ha podido usar el documento *Scores* proporcionado en la base de datos utilizada, descrita en [13]. Este documento contiene información de los tres experimentos de la base de datos, y contiene los siguientes datos:

- Nombre del vídeo.
- Tiempo de experimento, contado desde que el usuario inicia la tarea hasta que la termina.
- Errores cometidos durante el experimento.
- Puntuación del experimento, es decir la suma del tiempo de experimento más el número de errores multiplicados por el tiemp de penalización.

Tras realizar la evaluación automática de los vídeos del experimento *Post and sleeve* de la base de datos se realiza una comparación de los resultados del archivo *Scores* que contiene los datos descritos anteriormente con los obtenidos con el código implementado durante este TFM. En la tabla 5.2 se pueden comparar dichos resultados.

Vídeo	Tiempo de experimento real (s)	Tiempo de experimento automático (s)	Errores reales	Errores automáticos
X01 Post And Sleeve 01	118.45	126.60	0	0
X01 Post And Sleeve 02	105.46	109.73	0	0
X01 Post And Sleeve 03	97.45	99.93	0	0
X01 Post And Sleeve 04	84.63	87.07	0	0
X01 Post And Sleeve 05	86.86	90.87	1	1
X01 Post And Sleeve 06	97.89	100.20	0	0
X02 Post And Sleeve 01	160.15	161.00	0	0
X02 Post And Sleeve 02	128.40	130.87	0	0
X02 Post And Sleeve 03	117.59	116.27	0	1
X02 Post And Sleeve 04	116.41	116.93	0	0
X02 Post And Sleeve 05	103.04	107.87	0	0
X02 Post And Sleeve 06	120.18	123.93	0	0
X03 Post And Sleeve 01	217.72	203.80	0	0
X03 Post And Sleeve 02	272.78	227.13	0	0
X03 Post And Sleeve 03	231.91	233.87	2	0
X03 Post And Sleeve 04	156.26	159.67	1	1
X03 Post And Sleeve 05	121.99	124.60	0	0
X04 Post And Sleeve 01	145.16	142.33	2	2
X04 Post And Sleeve 02	183.50	145.33	0	0
X04 Post And Sleeve 03	175.07	174.20	0	0
X04 Post And Sleeve 04	129.94	122.33	1	1
X04 Post And Sleeve 05	111.25	110.33	0	0
X05 Post And Sleeve 01	161.39	162.07	0	0
X05 Post And Sleeve 02	156.03	146.40	0	0
X05 Post And Sleeve 03	177.53	180.20	0	0
X05 Post And Sleeve 04	122.77	116.00	0	0
X05 Post And Sleeve 05	134.56	134.60	0	0
X06 Post And Sleeve 01	144.00	136.60	0	0
X06 Post And Sleeve 02	90.90	91.60	0	0
X06 Post And Sleeve 03	83.91	84.53	0	0
X06 Post And Sleeve 04	110.9	114.27	1	1
X06 Post And Sleeve 05	79.59	76.87	0	0
X06 Post And Sleeve 06	108.32	109.27	0	0
X07 Post And Sleeve 01	181.40	179.60	1	1
X07 Post And Sleeve 02	130.94	132.22	0	0
X07 Post And Sleeve 03	115.85	107.27	0	0
X07 Post And Sleeve 04	92.62	93.87	0	0
X07 Post And Sleeve 05	79.90	75.47	1	1
X07 Post And Sleeve 06	89.85	89.80	0	0
X08 Post And Sleeve 01	180.54	173.33	0	0
X08 Post And Sleeve 02	129.52	133.40	0	0
X08 Post And Sleeve 03	102.25	97.87	0	0
X08 Post And Sleeve 04	117.94	122.07	0	0
X08 Post And Sleeve 05	113.55	109.27	0	0
X08 Post And Sleeve 06	123.86	128.00	0	0
X09 Post And Sleeve 01	129.17	121.00	0	0
X09 Post And Sleeve 02	125.83	123.67	0	0
X09 Post And Sleeve 03	90.73	83.20	0	0
X09 Post And Sleeve 04	85.60	85.07	0	0
X10 Post And Sleeve 01	160.02	153.33	1	0
X10 Post And Sleeve 02	131.53	131.33	2	0

X10 Post And Sleeve 03	96.03	95.00	0	0
X10 Post And Sleeve 04	99.83	103.80	0	0
X10 Post And Sleeve 05	79.50	83.00	0	0
X10 Post And Sleeve 06	76.42	81.33	0	0
X11 Post And Sleeve 01	210.51	210.60	1	0
X11 Post And Sleeve 02	154.70	159.13	2	0
X11 Post And Sleeve 03	121.25	119.27	0	0
X11 Post And Sleeve 04	124.52	125.33	1	1
X11 Post And Sleeve 05	83.47	82.27	0	0
X11 Post And Sleeve 06	99.03	121.20	1	0
X12 Post And Sleeve 01	193.11	365.60	0	0
X12 Post And Sleeve 02	149.32	143.47	0	0
X12 Post And Sleeve 03	145.73	149.47	0	0
X12 Post And Sleeve 04	158.87	152.47	0	0

Tabla 5.2: Comparación entre los tiempos y errores originales y los obtenidos automáticamente.

En el experimento, se calcula la puntuación final sumando 15 segundos por error al tiempo total del experimento. Se ha hecho lo mismo con los resultados obtenidos automáticamente. Se pueden ver las puntuaciones originales con las obtenidas con el algoritmo en la Tabla 5.3.

Vídeo	Puntuación Real (s)	Puntuación automática (s)
X01 Post And Sleeve 01	118.45	126.60
X01 Post And Sleeve 02	105.46	109.73
X01 Post And Sleeve 03	97.45	99.93
X01 Post And Sleeve 04	84.63	87.07
X01 Post And Sleeve 05	101.86	105.87
X01 Post And Sleeve 06	97.89	100.20
X02 Post And Sleeve 01	160.15	161.00
X02 Post And Sleeve 02	128.40	130.87
X02 Post And Sleeve 03	117.59	131.27
X02 Post And Sleeve 04	116.41	116.93
X02 Post And Sleeve 05	103.04	107.87
X02 Post And Sleeve 06	120.18	123.93
X03 Post And Sleeve 01	217.72	203.80
X03 Post And Sleeve 02	272.78	227.13
X03 Post And Sleeve 03	261.91	233.87
X03 Post And Sleeve 04	171.26	174.67
X03 Post And Sleeve 05	121.99	124.60
X04 Post And Sleeve 01	175.16	172.33
X04 Post And Sleeve 02	183.50	145.33
X04 Post And Sleeve 03	175.07	174.20
X04 Post And Sleeve 04	144.93	137.33
X04 Post And Sleeve 05	111.25	110.33
X05 Post And Sleeve 01	161.39	162.07
X05 Post And Sleeve 02	156.03	146.40
X05 Post And Sleeve 03	177.53	180.20
X05 Post And Sleeve 04	122.77	116.00
X05 Post And Sleeve 05	134.56	134.60
X06 Post And Sleeve 01	144.00	136.60
X06 Post And Sleeve 02	90.90	91.60
X06 Post And Sleeve 03	83.91	84.53
X06 Post And Sleeve 04	125.90	129.27
X06 Post And Sleeve 05	79.59	76.87

X06 Post And Sleeve 06	108.32	109.27
X07 Post And Sleeve 01	196.39	194.60
X07 Post And Sleeve 02	130.94	132.22
X07 Post And Sleeve 03	115.85	107.27
X07 Post And Sleeve 04	92.62	93.87
X07 Post And Sleeve 05	94.90	90.47
X07 Post And Sleeve 06	89.85	89.80
X08 Post And Sleeve 01	180.54	173.33
X08 Post And Sleeve 02	129.52	133.40
X08 Post And Sleeve 03	102.25	97.87
X08 Post And Sleeve 04	117.94	122.07
X08 Post And Sleeve 05	113.55	109.27
X08 Post And Sleeve 06	123.86	128.00
X09 Post And Sleeve 01	129.17	121.00
X09 Post And Sleeve 02	125.83	123.67
X09 Post And Sleeve 03	90.73	83.20
X09 Post And Sleeve 04	85.60	85.07
X10 Post And Sleeve 01	175.02	153.33
X10 Post And Sleeve 02	161.53	131.33
X10 Post And Sleeve 03	96.03	95.00
X10 Post And Sleeve 04	99.83	103.80
X10 Post And Sleeve 05	79.50	83.00
X10 Post And Sleeve 06	76.42	81.33
X11 Post And Sleeve 01	225.51	210.60
X11 Post And Sleeve 02	184.70	159.13
X11 Post And Sleeve 03	121.25	119.27
X11 Post And Sleeve 04	139.52	140.33
X11 Post And Sleeve 05	83.47	82.27
X11 Post And Sleeve 06	114.03	121.20
X12 Post And Sleeve 01	193.11	365.60
X12 Post And Sleeve 02	149.32	143.47
X12 Post And Sleeve 03	145.73	149.47
X12 Post And Sleeve 04	158.87	152.47

Tabla 5.3: Comparación entre las puntuaciones originales y las obtenidos automáticamente.

Como se puede ver en las Tablas 5.2 y 5.3 los resultados en casi todos los casos salen prácticamente igual en el caso automatizado al que proporciona en el archivo *Scores* junto al dataset, por lo que se puede decir que se ha cumplido el objetivo de este TFM. Este objetivo era automatizar la obtención de la puntuación de los vídeos de la tarea *Post and Sleeve*.

6. Conclusiones y líneas futuras

6.1. Conclusiones

En el presente TFM se esperaba realizar el reconocimiento de herramientas en vídeos de robótica quirúrgica y la evaluación automática de una tarea concreta, en este caso esa tarea es la tarea *Post and sleeve* del dataset proporcionado en [13]. Para ello, primero era necesario obtener los siguientes datos:

- **Etiquetado de herramientas mediante redes neuronales:** En este apartado se ha decidido hacer uso de la red neuronal YOLO, ya que para este tipo de tareas ha demostrado ser una solución eficiente para rastrear objetos específicos. Como se pueden ver en los resultados de dichas detección se ha obtenido una precisión media en los vídeos de este experimento para ambas herramientas de 0.99, un valor bastante alto, y que tras etiquetar los vídeos, se ha podido realizar la comprobación de que el etiquetado de las herramientas se ha realizado con una precisión bastante alta.
- **Etiquetado de objetos de colores:** En este apartado se ha decidido optar por la segmentación por colores, aplicando distintas operaciones a los distintos imágenes del experimento como aplicar máscaras de color o aplicando máscaras adaptativas. La forma de evaluar este etiquetado ha sido mediante un porcentaje de acierto, donde se ha obtenido un valor del 68 %, aunque parece algo bajo, este etiquetado ha sido lo suficientemente bueno como para poder conseguir el objetivo final de este TFM.

Una vez obtenidas las etiquetas de las herramientas y los objetos de colores, se ha hecho uso de un sistema de inferencia mediante reglas lógicas para evaluar los experimentos. Para comprobar si los resultados han sido buenos, se han comparado los obtenidos mediante el código implementado con los de un archivo proporcionado por el dataset, donde se proporcionaba la información obtenida. Al comprobar dicha información, que viene desglosada por vídeo en la Tabla 5.3 se ha comprobado que los datos obtenidos mediante la evaluación automatizada se acercan bastante a los datos proporcionados en el dataset, por lo que se puede decir que el desarrollo de este TFM ha acabado con un resultado satisfactorio.

El reconocimiento de herramientas quirúrgicas y la deducción de la acción o la evaluación automática de las maniobras en cirugía mínimamente invasiva representan un área de investigación emocionante y en constante evolución. Los avances en este campo tienen el potencial de mejorar significativamente la precisión y la seguridad de los procedimientos quirúrgicos mínimamente invasivos, así como de cambiar la forma en que se realizan estas intervenciones médicas. Los avances presentados en [3] y [14], junto con otros avances, están allanando el camino hacia un futuro más prometedor en este campo crítico para la atención médica.

En resumen, este proyecto que combina etiquetado de herramientas con YOLO, etiquetado de objetos por colores y evaluación automática de experimentos mediante inferencia es un área emocionante de investigación y desarrollo en la automatización industrial y la robótica. La mejora continua de estos componentes y su integración en sistemas completos promete avanzar en la eficiencia y la versatilidad de una amplia gama de aplicaciones prácticas.

6.2. Líneas Futuras

A medida que el campo del reconocimiento de herramientas y la deducción de la acción en cirugía mínimamente invasiva continúa avanzando, se presentan varias tendencias y desafíos importantes:

1. **Reentrenar la red usando otras bases de datos:** Se puede hacer uso de transfer learning con imágenes de otras bases de datos como JIGSAWS, que usen herramientas similares. Pudiendo extrapolar el reconocimiento de las herramientas derecha e izquierda a estas bases de datos similares.
2. **Sistema de inferencia más completo:** Se puede mejorar el sistema de inferencia, incluyendo estados intermedios dentro de las maniobras, para detectar los errores durante la ejecución de la tarea. Esto se podría implementar haciendo uso de otro tipo de red neuronal etiquetando la base de datos con las acciones básicas que componen cada maniobra.
3. **Mejora de la detección de objetos:** Para mejorar el sistema de detección de objetos, se podrían detectar los objetos mediante redes neuronales al igual que las herramientas, mejorando su detección. Esto permitiría una mejor evaluación de los experimentos.

4. **Validación y optimización en aplicaciones a tiempo real:** Una línea importante es llevar a cabo pruebas y validaciones exhaustivas en entornos robóticos a tiempo real para realizar la evaluación de las tareas en el mismo momento que se ejecutan.

Referencias

[1] R. E. Perez and S. D. Schwaitzberg, "Robotic surgery: finding value in 2019 and beyond," Ann. Laparosc. Endosc. Surg., vol. 4, no. 0, pp. 51–51, May 2019, doi: 10.21037/ALES.2019.05.02.

[2] RH. Singh et al., "Robotic Surgery Improves Technical Performance and Enhances Prefrontal Activation During High Temporal Demand," Ann. Biomed. Eng., vol. 46, no. 10, pp. 1621–1636, Oct. 2018, doi: 10.1007/S10439-018-2049-Z/FIGURES/5.

[3] I. Rivas-Blanco, C. J. Perez-Del-Pulgar, I. Garcia-Morales, V. F. Munoz, and I. Rivas-Blanco, "A Review on Deep Learning in Minimally Invasive Surgery," IEEE Access, vol. 9, pp. 48658–48678, 2021, doi: 10.1109/ACCESS.2021.3068852.

[4] S. Wang, A. Raju, and J. Huang, "Deep Learning based multi-label classification for surgical tool presence detection in laparoscopic vídeos," Proc. - Int. Symp. Biomed. Imaging, pp. 620–623, 2017, doi: 10.1109/ISBI.2017.7950597.

[5] RL. C. Garcia-Peraza-Herrera et al., "ToolNet: Holistically-nested real-time segmentation of robotic surgical tools," IEEE Int. Conf. Intell. Robot. Syst., vol. 2017-Septe, pp. 5717–5722, 2017, doi: 10.1109/IROS.2017.8206462.

[6] D. Sarikaya, J. J. Corso, and K. A. Guru, "Detection and Localization of Robotic Tools in Robot-Assisted Surgery vídeos Using Deep Neural Networks for Region Proposal and Detection," IEEE Trans. Med. Imaging, vol. 36, no. 7, pp. 1542–1549, Jul. 2017, doi: 10.1109/TMI.2017.2665671.

[7] Y. Fu et al., "More unlabelled data or label more data? a study on semi-supervised laparoscopic image segmentation," in Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 2019, vol. 11795 LNCS, pp. 173–180, doi: 10.1007/978-3-030-33391-1_20.

[8] A. Nazir et al., "SPST-CNN: Spatial pyramid based searching and tagging of liver's intraoperative live views via CNN for minimal invasive surgery," J. Biomed. Inform., vol. 106, p. 103430, Jun. 2020, doi: 10.1016/j.jbi.2020.103430.

[9] M. Hwang et al., "An Adaptive Regularization Approach to Colonoscopic Polyp Detection Using a Cascaded Structure of Encoder–Decoders," Int. J. Fuzzy Syst., vol. 21, no. 7, pp. 2091–2101, Oct. 2019, doi: 10.1007/s40815-019-00694-y.

[10] S. Petscharnig, K. Schoffmann, J. Benois-Pineau, S. Chaabouni, and J. Keckstein, "Early and Late Fusion of Temporal Information for Classification of Surgical Actions in Laparoscopic Gynecology," in Proceedings - IEEE Symposium on Computer-Based Medical Systems, 2018, vol. 2018-June, pp. 369–374, doi: 10.1109/CBMS.2018.00071.

[11] X. Gao, Y. Jin, Q. Dou, and P.-A. Heng, "Automatic Gesture Recognition in Robot-assisted Surgery with Reinforcement Learning and Tree Search," in 2020 IEEE International Conference on Robotics and Automation (ICRA), 2020, pp. 8440–8446, doi: 10.1109/icra40945.2020.9196674.

[12] A. P. Twinanda, S. Shehata, D. Mutter, J. Marescaux, M. De Mathelin, and N. Padoy, "EndoNet: A Deep Architecture for Recognition Tasks on Laparoscopic vídeos," IEEE Trans. Med. Imaging, vol. 36, no. 1, pp. 86–97, Jan. 2017, doi: 10.1109/TMI.2016.2593957.

[13] I. Rivas-Blanco, C. J. P. Del-Pulgar, A. Mariani, G. Tortora and A. J. Reina, .ª surgical dataset from the da Vinci Research Kit for task automation and recognition,"2023 3rd International Conference on Electrical, Computer, Communications and Mechatronics Engineering (ICECCME), Tenerife, Canary Islands, Spain, 2023, pp. 1-6, doi: 10.1109/ICECCME57830.2023.10253032.

[14] Nwoye, C. I., Gonzalez, C., Yu, T., Mascagni, P., Mutter, D., Marescaux, J., and Padoy, N. (2022). Recognition of Instrument-Tissue Interactions in Endoscopic vídeos via Action Triplets.

[15] Vedula SS and Hager GD (2020) Surgical data science: The new knowledge domain. Innovative Surgical Sciences 2(3): 109–121. DOI:10.1515/iss-2017-0004

[16] Perez-del Pulgar CJ, Smisek J, Rivas-Blanco I, Schiele A and Munoz VF (2019) Using Gaussian Mixture Models for Gesture Recognition During Haptically Guided Telemanipulation. Electronics 8(7): 772. DOI:10.3390/electronics8070772.

[17] Ahmidi N, Tao L, Sefati S, Gao Y, Lea C, Haro BB, Zappella L, Khudanpur S, Vidal R and Hager GD (2017) A Dataset and Benchmarks for Segmentation and Recognition of Gestures in Robotic Surgery. IEEE Transactions on Biomedical Engineering 64(9): 2025–2041. DOI:10.1109/TBME.2016. 2647680.

[18] Gao Y, Swaroop Vedula S, Reiley CE, Ahmidi N, Varadarajan B, Lin HC, Tao L, Zappella L, Bejar B, Yuh DD, Chiung C, Chen G, Vidal R, Khudanpur S and Hager GD (2014) JHUISI Gesture and Skill Assessment Working Set (JIGSAWS): A Surgical Activity Dataset for Human Motion Modeling. In: MICCAI Workshop: Modeling and Monitoring of Computer Assisted Interventions (M2CAI). Boston, MA.

[19] MathWorks. (s.f.). Object Detection using YOLO v3 Deep Learning. Recuperado de https://es.mathworks.com/help/vision/ug/object-detection-using-yolo-v3-deep-learning.html

[20] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," Commun. ACM, vol. 60, no. 6,pp. 84–90, May 2017.

[21] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in Proc. 3rd Int. Conf. Learn. Represent. Conf. Track (ICLR), Sep. 2015, pp. 1–14.

[22] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions,"in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), vols. 7–12,Jun. 2015, pp. 1–9.

[23] F. Chollet, Deep Learning With Python. Shelter Island, NY, USA: Manning, 2017.

[24] Mahendru, M., & Dubey, S. K. (2021). Real Time Object Detection with Audio Feedback using Yolo vs. Yolo_v3. En 11th International Conference on Cloud Computing, Data Science & Engineering (Confluence), 7 Pages. IEEE.

[25] Yu, X., Kuan, T. W., Zhang, Y., & Yan, T. (2022). YOLO v5 for SDSB Distant Tiny Object Detection. En 10th International Conference on Orange Technology (ICOT), 4 Pages. IEEE.

[26] Ganesan, P., Sathish, B. S., Leo Joseph, L. M. I., Sajiv, G., Murugesan, R., Akilandeswari, A., & Gomathi, S. (2023). HSV Model based Skin Color Segmentation using Uncomplicated Threshold and Logical AND Operation. En 9th International Conference on Advanced Computing and Communication Systems (ICACCS), 5 Pages. IEEE.

[27] Boopathi Kumar, E., & Thiagarasu, V. (2017). Color channel extraction in RGB images for segmentation. En 2nd International Conference on Communication and Electronics Systems (ICCES), 6 Pages. IEEE.

[28] Luo, G., & Yin, C. (2016). Deductive reasoning and computing based on propositional logic. En 2016 IEEE 15th International Conference on Cognitive Informatics & Cognitive Computing (ICCI*CC), 6 Pages. IEEE.

[29] Titov, A. V. (2022). Non-finite methods of Reconstructions of Non-Classical Propositional Logic Variants based on the Theory of Structures in Control Problems of Complex Systems. En 2022 15th International Conference Management of large-scale system development (MLSD), 4 Pages. IEEE.

[30] MathWorks. (s.f.). Get Started with the vídeo Labeler. Recuperado de https://es.mathworks.com/help/vision/ug/get-started-with-the-vídeo-labeler.html

[31] Hasan, M. N., Hamdan, S., Poudel, S., Vargas, J., & Poudel, K. (2023). Prediction of Length-of-stay at Intensive Care Unit (ICU) Using Machine Learning based on MIMIC-III Database. En 2023 IEEE Conference on Artificial Intelligence (CAI), 3 Pages. IEEE.

[32] Moody GB, Mark RG. A Database to Support Development and Evaluation of Intelligent Intensive Care Monitoring. Computers in Cardiology 23:657–660 (1996).

[33] F. Setti, E. Oleari, A. Leporini, D. Trojaniello, A. Sanna, U. Capitanio, F. Montorsi, A. Salonia, and R. Muradore, "A Multirobots Teleoperated Platform for Artificial Intelligence Training Data Collection in Minimally Invasive Surgery," in 2019 International Symposium on Medical Robotics, ISMR 2019. Institute of Electrical and Electronics Engineers Inc., 5 2019.

[34] E. Colleoni, P. Edwards, and D. Stoyanov, "Synthetic and Real Inputs for Tool Segmentation in Robotic Surgery," in International Conference on Medical Image Computing and Computer-Assisted Intervention - MICCAI 2020. Lima, Peru: Springer, Cham, 10 2020, pp. 700–710. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-030-59716-0 67

[35] P. Kazanzides, Z. Chen, A. Deguet, G. S. Fischer, R. H. Taylor, and S. P. Dimaio, "An Open-Source Research Kit for the da Vinci R Surgical System," in IEEE International Conference on Robotics & Automation (ICRA), Hong Kong, China, 2014, pp. 6434–6439.

[36] Z. Chen, A. Deguet, R. H. Taylor, and P. Kazanzides, "Software architecture of the da vinci research kit," in Proceedings- 2017 1st IEEE International Conference on Robotic Computing, IRC 2017. Institute of Electrical and Electronics Engineers Inc., 5 2017, pp. 180–187.

[37] G. A. Fontanelli, F. Ficuciello, L. Villani, and B. Siciliano, "Modelling and identification of the da Vinci Research Kit robotic arms," in IEEE International Conference on Intelligent Robots and Systems, vol. 2017 Septe. Institute of Electrical and Electronics Engineers Inc., 12 2017, pp. 1464–1469.

[38] Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You Only Look Once: Unified, Real-Time Object Detection. University of Washington, Allen Institute for AI, Facebook AI Research.

[39] R. Anteby, N. Horesh, S. Soffer, Y. Zager, Y. Barash, I. Amiel, D. Rosin, M.Gutman,andE.Klang, "Deep Learning visual analysis in laparoscopic surgery: A systematic review and diagnostic test accuracy meta-analysis", Surgical Endoscopy, vol. 35, no. 4, pp. 15211533, Apr. 2021.

[40] K. Mishra, R. Sathish, and D. Sheet, "Learning latent temporal connectionism of deep residual visual abstractions for identifying surgical tools in laparoscopy procedures", in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW). Washington, DC, USA: IEEE Computer Society, Jul. 2017, pp. 22332240.

[41] H. Al Hajj, M. Lamard, P.-H. Conze, B. Cochener, and G. Quellec, "Monitoring tool usage in surgery vídeos using boosted convolutional and recurrent neural networks", Med. Image Anal., vol. 47, pp. 203218, Jul. 2018.

[42] S.Wang, Z.Xu, C.Yan, and J.Huang, "Graph Convolutional Nets for Tool Presence Detection in Surgical vídeos"(Lecture Notes in Computer Science: Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), vol. 11492. Cham, Switzerland: Springer-Verlag, Jun. 2019, pp. 467478.

[43] T. Kurmann, P. M. Neila, X. Du, P. Fua, D. Stoyanov, S. Wolf, and R. Sznitman, "Simultaneous Recognition and Pose Estimation of Instruments in Minimally Invasive Surgery"(Lecture Notes in Computer Science: Lecture Notes in Arti cial Intelligence and Lecture Notes in Bioinformatics), vol. 10434. Cham, Switzerland: Springer-Verlag, Sep. 2017, pp. 505513.

[44] X. Du, T. Kurmann, P.-L. Chang, M. Allan, S. Ourselin, R. Sznitman, J. D. Kelly, and D. Stoyanov, Articulated multi-instrument 2-D pose estimation using fully convolutional networks, IEEE Trans. Med. Imag., vol. 37, no. 5, pp. 12761287, May 2018.

[45] E. Colleoni, S. Moccia, X. Du, E. De Momi, and D. Stoyanov, Deep Learning based robotic tool detection and articulation estimation with spatio-temporal layers, IEEE Robot. Autom. Lett., vol. 4, no. 3, pp. 27142721, Jul. 2019.

[46] Z.Chen, Z.Zhao, and X.Cheng, "Surgical instruments tracking based on Deep Learning with lines detection and spatio-temporal context", in Proc. Chin. Autom. Congr. (CAC). Piscataway, NJ, USA: IEEE, Oct. 2017, pp. 27112714.

[47] Z. Zhao, T. Cai, F. Chang, and X. Cheng, Real-time surgical instrument detection in robot-assisted surgery using a convolutional neural network cascade, Healthcare Technol". Lett., vol. 6, no. 6, pp. 275279, Dec. 2019.

[48] Y. Liu, Z. Zhao, F. Chang, and S. Hu, .An anchor-free convolutional neural network for real-time surgical tool detection in robot-assisted surgery", IEEE Access, vol. 8, pp. 7819378201, 2020.

[49] A. A. Shvets, A. Rakhlin, A. A. Kalinin, and V. I. Iglovikov, .Automatic instrument segmentation in robot-assisted surgery using deep learning", in Proc. 17th IEEE Int. Conf. Mach. Learn. Appl. (ICMLA). Piscataway, NJ, USA: IEEE, Dec. 2018, pp. 624628.

[50] K. He, X. Zhang, S. Ren, and J. Sun, Deep residual learning for image recognition, in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2016, pp. 770778.

[51] A. Shrestha and A. Mahmood, Review of deep learning algorithms and architectures, IEEE Access, vol. 7, pp. 5304053065, 2019.

[52] C. E. Reiley and G. D. Hager, Task versus subtask surgical skill evaluation of robotic minimally invasive surgery, in Proc. Int. Conf. Med. Image Comput. Comput.-Assist. Intervent. (MICCAI). Berlin, Germany: Springer, 2009, pp. 435442.

[53] L. Tao, E. Elhamifar, S. Khudanpur, G. D. Hager, and R. Vidal, Sparse Hidden Markov Models for Surgical Gesture Classi cation and Skill Evaluation (Lecture Notes in Computer Science: Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), vol. 7330. Berlin, Germany: Springer, 2012, pp. 167177.

[54] H. I. Fawaz, G. Forestier, J. Weber, L. Idoumghar, and P. A. Müller, Evaluating Surgical Skills from Kinematic Data Using Convolutional Neural Networks (Lecture Notes in Computer Science: Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), vol. 11073. Cham, Switzerland: Springer-Verlag, Sep. 2018, pp. 214221.

[55] Z. Wang and A. M. Fey, Deep learning with convolutional neural network for objective skill evaluation in robot-assisted surgery, Int. J. Comput. Assist. Radiol. Surgery, vol. 13, no. 12, pp. 19591970, Dec. 2018.

[56] Z. Wang and A. M. Fey, SATR-DL: Improving surgical skill assessment and task recognition in robot-assisted surgery with deep neural networks, in Proc. 40th Annu. Int. Conf. IEEE Eng. Med. Biol. Soc. (EMBC). Piscataway, NJ, USA: IEEE, Jul. 2018, pp. 17931796.

[57] I. Funke, S. T. Mees, J. Weitz, and S. Speidel, Video-based surgical skill assessment using 3Dconvolutional neural networks, nt.J.Comput. Assist. Radiol. Surgery, vol. 14, no. 7, pp. 12171225, Jul. 2019.

[58] X. A. Nguyen, D. Ljuhar, M. Pacilli, R. M. Nataraja, and S. Chauhan, Surgical skill levels: Classification and analysis using deep neural network model and motion signals, Comput. Methods Programs Biomed., vol. 177, pp. 18, Aug. 2019.

[59] D. Zhang, Z. Wu, J. Chen, A. Gao, X. Chen, P. Li, Z. Wang, G. Yang, B. Lo, and G.-Z. Yang, Automatic microsurgical skill assessment based on cross-domain transfer learning, IEEE Robot. Autom. Lett., vol. 5, no. 3, pp. 41484155, Jul. 2020.

[60] N. Getty, Z. Zhao, S. Gruessner, L. Chen, and F. Xia, Recurrent and spiking modeling of sparse surgical kinematics, in Proc. Int. Conf. Neuromorphic Syst. New York, NY, USA: ACM, Jul. 2020, pp. 15.

[61] B. B. Oul, M. F. Gilgien, and P. D. ahin, Ranking Robot-Assisted Surgery Skills Using Kinematic Sensors (Lecture Notes in Computer Science: Lecture Notes in Arti cial Intelligence and Lecture Notes in Bioinformatics), vol. 11912. Springer, Nov. 2019, pp. 330336.

[62] A. Jin, S. Yeung, J. Jopling, J. Krause, D. Azagury, A. Milstein, and L. Fei-Fei, Tool detection and operative skill assessment in surgical videos using region-based convolutional neural networks, in Proc. IEEE Winter Conf. Appl. Comput. Vis. (WACV). Piscataway, NJ, USA: IEEE, Mar. 2018, pp. 691699.