Universidad Zaragoza
1542

Trabajo Fin de Máster

Realidad aumentada en cirugía: una aproximación semántica mediante aprendizaje profundo

Autor

Fernando Cacho Mairal

Directores

Dr. David González Ibáñez
Dr. Alberto Badías Herbera

Escuela de Ingeniería y Arquitectura

Junio de 2021

Universidad Zaragoza
1542

Trabajo Fin de Máster

Realidad aumentada en cirugía: una aproximación semántica mediante aprendizaje profundo

Autor

Fernando Cacho Mairal

Directores

Dr. David González Ibáñez

Dr. Alberto Badías Herbera

ESCUELA DE INGENIERÍA Y ARQUITECTURA

Curso 2020-2021

AGRADECIMIENTOS

En primer lugar, me gustaría dedicar este trabajo a David González y Alberto Badías, mis directores del Trabajo de Final de Máster. Muchas gracias por depositar vuestra confianza en mí y darme esta oportunidad.

Al Instituto de Investigación en Ingeniería de Aragón, gracias al cual he podido disfrutar de una beca de investigación durante estos últimos meses, y que me ha permitido iniciarme en el mundo de la investigación.

A mis compañeros del máster, con los que he compartido grandes momentos. También a mi familia y amigos más cercanos. Gracias a todos por estar ahí cuando más lo necesitaba.

Zaragoza, a 20 de junio de 2021

Realidad aumentada en cirugía: una aproximación semántica mediante aprendizaje profundo

RESUMEN

En este proyecto se ha desarrollado una herramienta que combina técnicas clásicas de visión por computador para tomar medidas precisas junto con técnicas de aprendizaje profundo. Esto permite crear un sistema capaz de entender la escena que está viendo, a la vez que la posiciona en el espacio de manera precisa, permitiendo incluso tomar medidas en verdadera magnitud. La fusión de estos dos tipos de tecnología permite abrir una nueva línea de trabajo, estableciendo las bases para la extracción semántica del interior de un paciente en un entorno quirúrgico.

El principal reto abordado en el proyecto es la identificación de las regiones internas del paciente (el hígado en este caso) a partir de imágenes planas tomadas con cámaras monoculares estándar, como un endoscopio. El objetivo final es la estimación de la *pose* (posición con respecto a la cámara, compuesta de traslación y rotación) del hígado, que se utilizará para localizar partes internas no visibles, como vasos sanguíneos o tumores, sobre el órgano en realidad aumentada durante una intervención quirúrgica.

Para el entrenamiento de la red neuronal se ha utilizado un modelo sintético del hígado, obtenido a partir de un simulador quirúrgico desarrollado en el Grupo AMB del Instituto de Investigación en Ingeniería de Aragón (I3A).

La principal dificultad del proyecto radica en el entrenamiento de la red para la obtención de la *pose* de forma precisa. Para ello, se ha reentrenado un modelo de red neuronal con imágenes del hígado, tanto sobre fondos homogéneos como sobre fondos simulando una intervención laparoscópica, con el objetivo de realizar predicciones en condiciones lo más realistas posibles.

Finalmente, la información obtenida mediante la red neuronal se ha incorporado a ORB-SLAM, para la obtención de resultados en tiempo real.

La principal novedad introducida en este proyecto es el uso conjunto de redes neuronales con ORB-SLAM. Esto permite realizar estimaciones de *pose* y escalado automáticamente sin la necesidad de utilizar información adicional, de modo que la herramienta se puede utilizar directamente con cámaras de laparoscopia, sin tener que recurrir a sensores adicionales como acelerómetros o LIDAR.

Índice

1. Introducción y objetivos **1**

1.1. Objetivo y alcance del proyecto 1

1.2. Estado del arte 2

1.3. Contenido del proyecto 4

2. Preproceso **5**

2.1. Calibración de la cámara 5

2.2. Jerarquía de los datos 7

2.3. Extracción de datos desde el simulador 8

3. Entrenamiento de la red neuronal **13**

3.1. Condiciones de entrenamiento 13

3.2. Evolución del entrenamiento 15

3.3. Resultados del entrenamiento 17

4. Integración con ORB-SLAM **21**

4.1. Nexo entre ORB-SLAM y PVNet 21

4.2. Calibración de la webcam 22

4.3. Segmentación de la nube de puntos 23

4.4. Seguimiento del objeto 24

4.4.1. Ajuste de la escala del modelo 24

4.4.2. Corrección de la pose 28

4.4.3. Registro del modelo 33

5. Resultados **39**

5.1. Modelo del gato 39

5.2. Modelo del hígado 40

6. Conclusiones **45**

Bibliografía **47**

Lista de Figuras **51**

Lista de Tablas **55**

Anexos **56**

A. Funcionamiento de PVNet **59**

A.1. Arquitectura de PVNet 59

A.2. Estimación de la pose 59

A.2.1. Estimación de los keypoints 60

A.2.2. Estimación de la matriz rotación-traslación 60

B. Validación del modelo **63**

C. Métrica de evaluación de la red neuronal **67**

C.1. Métrica de la traslación 67

C.2. Métrica de la rotación 68

C.3. Media y varianza de los resultados 68

Capítulo 1

Introducción y objetivos

1.1. Objetivo y alcance del proyecto

El principal objetivo de este proyecto es dotar a un sistema de la capacidad para comprender, detectar y localizar los diferentes tejidos internos de un cuerpo humano. En este trabajo se pretende establecer las bases, comenzando por el desarrollo de una herramienta para la estimación de la posición y orientación de órganos en el interior del cuerpo humano a partir de secuencias de vídeo de laparoscopia, con la posibilidad de poder superponer información relevante durante una intervención quirúrgica. Concretamente, este trabajo se centra en la estimación de la *pose* de hígados en el interior de la cavidad abdominal, que se utilizará para la representación robusta mediante realidad aumentada de los elementos anatómicos de interés, como vasos sanguíneos o tumores.

Los cambios anatómicos sufridos durante una intervención laparoscópica, debidos a la introducción de CO_2 y a los cortes y cauterizaciones realizados con el bisturí, sumado al humo y a la posible oclusión de la imagen por la superposición de las herramientas laparoscópicas, suponen los grandes retos a la hora de desarrollar este tipo de herramientas.

El alcance del proyecto se puede definir como la fusión entre herramientas de medida y técnicas semánticas, mediante la utilización de redes neuronales convolucionales y técnicas de visión por computador para la estimación de la *pose* del hígado de forma automática. Para el entrenamiento de la red neuronal se han utilizado imágenes de hígados sintéticos en distintas posiciones, obtenidas a partir de un simulador quirúrgico desarrollado en el *Applied Mechanics & Bioengineering Group* (AMB) del Instituto de Investigación en Ingeniería de Aragón (I3A) [1] [2]. Para crear un sistema robusto, se han empleado distintos fondos de laparoscopia [3] para un entrenamiento lo más realista y cercano posible a las condiciones reales de uso de la herramienta. Posteriormente, se ha integrado la *pose* y la segmentación del hígado con ORB-SLAM, un programa *open*

source también desarrollado en el I3A [4] [5].

Con el funcionamiento conjunto de la red neuronal y ORB-SLAM se consigue la localización del órgano en tiempo real, así como una nube de puntos en 3D del hígado, que se podrá utilizar en un futuro para implementar un modelo de comportamiento mecánico.

1.2. Estado del arte

Uno de los campos de aplicación con mayor interés de la inteligencia artificial (IA) es la medicina. Durante la última década, se han desarrollado nuevos algoritmos y técnicas con el objetivo de asistir al médico y proveerle de mayor información, ya sea durante el diagnóstico o durante la intervención quirúrgica.

En [6] se describen los diferentes algoritmos desarrollados para asistir al médico en las diferentes tareas que desempeña. Uno de los grandes retos a los que se enfrenta la IA es el registro y la interpretación de las imágenes preoperatorias con las imágenes intraoperatorias en tiempo real durante una intervención quirúrgica.

En [7] se realiza un análisis de las diferentes técnicas de registro de imagen médica, utilizando algoritmos de *Deep Learning* (DL). Se describen tres tipos de redes que se suelen utilizar para realizar esta tarea: redes neuronales convolucionales (CNN), redes de transformación espacial (STN), y redes adversarias generativas (GAN). Las más utilizadas en entrenamiento supervisado son las CNN, mientras que las más utilizadas en entrenamiento no supervisado (o mínimamente supervisado) son las GAN. El uso de redes neuronales en la estimación de las transformaciones espaciales permite acelerar el proceso de registro (comparado con algoritmos tradicionales).

Una CNN utilizada para el registro 3D de imágenes de resonancia magnética (MRI) es 3DPose-Net [8]. La arquitectura de esta red (ResNet) es una de las más utilizadas para estimar la *pose* de objetos (no sólo en el campo de la medicina) [9], y se basa en reformular las capas de la red como funciones residuales. Este tipo de redes son más fáciles de optimizar, y se puede obtener una gran precisión en los resultados si la red es lo suficientemente profunda.

Otra arquitectura de red ampliamente utilizada en el campo de la medicina es U-Net [10], aunque está limitada a la segmentación de imágenes. La principal característica de esta red es que permite la modificación de las imágenes utilizadas para entrenar, obteniendo más datos de entrenamiento. Esto le confiere una gran capacidad para entrenar el modelo con pocas imágenes, algo muy común en medicina. Su arquitectura también está basada en ResNet, y consiste en un *encoder-decoder* que realiza la extracción de características de las imágenes.

La gran mayoría de las redes utilizadas para la estimación de la *pose* están orientadas a un uso general, no centrado en la medicina. Sin embargo, su arquitectura las hace idóneas para desarrollar una herramienta como la descrita en este proyecto, ya que las imágenes utilizadas provienen de cámaras monoculares estándar.

En [11] se desarrolla una CNN de estimación de *pose* (denominada BB8), que se basa en la creación de una caja de 8 vértices alrededor del objeto, mediante la cual se realizan correspondencias 2D-3D para estimar la traslación y la rotación del objeto mediante el algoritmo *Perspective-n-Point* (PnP) [12]. PoseCNN es una red neuronal convolucional [13] en la que se realiza la estimación de la *pose* del objeto sin ningún algoritmo clásico de visión por computador adicional. DPOD [14] se basa en mapas de correspondencia para establecer la relación 2D-3D del objeto, aplicando posteriormente los algoritmos RANSAC [15] y PnP.

Por otro lado, la red PVNet [16] tiene un funcionamiento muy similar a DPOD, pero en vez de utilizar mapas de correspondencia para establecer la relación 2D-3D del objeto, predice vectores unitarios que representan las direcciones desde cada píxel del objeto segmentado hacia varios puntos del objeto, denominados *keypoints*. Esto le confiere una mayor robustez ante oclusiones y truncamientos. Por todo lo anterior, PVNet es la red que se ha elegido para la creación de la herramienta descrita en este proyecto. En el Anexo A se presentan más detalles de su funcionamiento.

Una vez localizado el objeto de interés (en este caso el hígado) con la red neuronal, es necesario realizar un seguimiento, mediante la detección de puntos 3D de la escena. De esta forma, se puede estimar la verdadera escala del objeto, junto con la información aportada por la red neuronal, y una nube de puntos 3D del mismo, que dará la posibilidad de implementar en un futuro un modelo mecánico que soporte deformaciones del objeto.

Una de las tecnologías más utilizadas para el seguimiento de puntos de un objeto es SLAM (*Simultaneous Localization and Mapping*). En [17] se desarrolla un método SLAM para la estimación y el seguimiento de la *pose* de objetos a partir de un algoritmo híbrido, creado a partir de métodos bayesianos y técnicas de medición inversa. Se diseñó con el objetivo de utilizarse en entornos donde no se puede obtener información a priori, como la reparación de satélites o entornos submarinos.

En [18] se aplican las técnicas SLAM sobre entornos quirúrgicos de laparoscopia. Para la localización de la cámara y el seguimiento de los puntos se utiliza un algoritmo EKF-SLAM [19], que hace uso de un filtro Kalman extendido en conjunto con SLAM. A diferencia de lo desarrollado en [18], en este proyecto se hace uso de redes neuronales.

En [20] se desarrollan técnicas SLAM para la localización de la cámara en cirugías de laparoscopia ejecutadas con robots quirúrgicos. Los resultados obtenidos se comparan

con ORB-SLAM [4], obteniendo errores del mismo orden de magnitud.

ORB-SLAM [4] [5] es el algoritmo que se ha utilizado en este proyecto para realizar el seguimiento de los puntos y la discriminación de los puntos del objeto en el mapa 3D. Se trata de un algoritmo SLAM de reconocimiento de puntos que utiliza las mismas características para todas las tareas: seguimiento, mapeo, relocalización y cierre de bucle. Se utiliza un descriptor ORB [21] para obtener las características de la imagen.

En [22] se desarrolla un modelo para el seguimiento de objetos deformables basado en ORB-SLAM. Mediante el seguimiento de los puntos, se obtienen los resultados de desplazamientos en tiempo real, resolviendo el problema de hiperelasticidad. Para su uso en tiempo real, se utilizan métodos de reducción de orden, que permiten reducir la dimensión del problema hiperelástico y su coste computacional. Los resultados obtenidos en el artículo serían el futuro trabajo a realizar a partir de este proyecto, que se limita a la localización del hígado como sólido rígido.

1.3. Contenido del proyecto

En el segundo capítulo se presentan todos los pasos previos al entrenamiento de la red neuronal: la calibración de la cámara, la estructura de los datos y la elaboración del conjunto de datos para el entrenamiento de la red neuronal.

En el tercer capítulo se describen los entrenamientos realizados con la red neuronal hasta llegar a la mejor aproximación posible, y que más se ajusta a las condiciones de uso de la herramienta. También se presentan los resultados del último entrenamiento realizado con la red neuronal, que se utiliza para diseñar la herramienta en conjunto con ORB-SLAM.

En el cuarto capítulo se presenta el diseño de la herramienta para el uso conjunto de PVNet con ORB-SLAM. Se definen los sistemas de referencia utilizados para posicionar el modelo 3D sobre la imagen en realidad aumentada, se describe la obtención de la nube de puntos en 3D del objeto y se presentan los procesos de optimización y registro, necesarios para obtener una buena aproximación de la *pose* del objeto.

En el quinto capítulo, se presentan los resultados finales en formato vídeo e imagen, pudiéndose comprobar la representación en realidad aumentada de un tumor y de la vena porta sobre el hígado.

Capítulo 2

Preproceso

Antes de comenzar la elaboración de la base de datos que se utiliza para entrenar la red neuronal es necesario realizar varios pasos previos, como la calibración de la cámara y la replicación de la jerarquía de archivos, utilizada originalmente para el entrenamiento de la red.

2.1. Calibración de la cámara

La red neuronal con la que se va a desarrollar el proyecto (PVNet)[16] está preentrenada con un conjunto de datos (*dataset*, en inglés) ya existente, denominado LINEMOD, que está compuesto por fotografías de diferentes objetos, además de varios datos adicionales para entrenar a la red [23].

Para una correcta estimación de la *pose*, es necesario conocer los parámetros de la cámara con la que se ha entrenado la red. Los parámetros de referencia utilizados son los correspondientes a la matriz intrínseca de la cámara de la red neuronal.

Cuando se utiliza otro dispositivo para capturar las imágenes de entrenamiento de la red neuronal, es necesario ajustar los parámetros de la cámara. En este trabajo, en ausencia de grabaciones *in vivo* de hígados reales, se ha hecho uso de un simulador de cirugías [1] [2], donde la matriz intrínseca de la cámara está expresada en un formato diferente (OpenGL), mientras que en la red neuronal los parámetros de la cámara se expresan como una matriz intrínseca estándar. Dicha matriz intrínseca (también denominada matriz de Hartley-Zisserman [24]) se compone de los siguientes elementos:

$$\mathbf{K_{int}} = \begin{pmatrix} f_x & s & x_0 \\ 0 & f_y & y_0 \\ 0 & 0 & 1 \end{pmatrix} \tag{2.1}$$

donde f_x y f_y representan la distancia focal de la cámara en términos de las dimensiones de píxel en las direcciones X e Y, respectivamente. El parámetro s expresa la distorsión de los ejes o píxeles, siendo un valor nulo en la mayoría de casos. Los parámetros x_0 e

y_0 representan las coordenadas del punto del eje principal de la cámara.

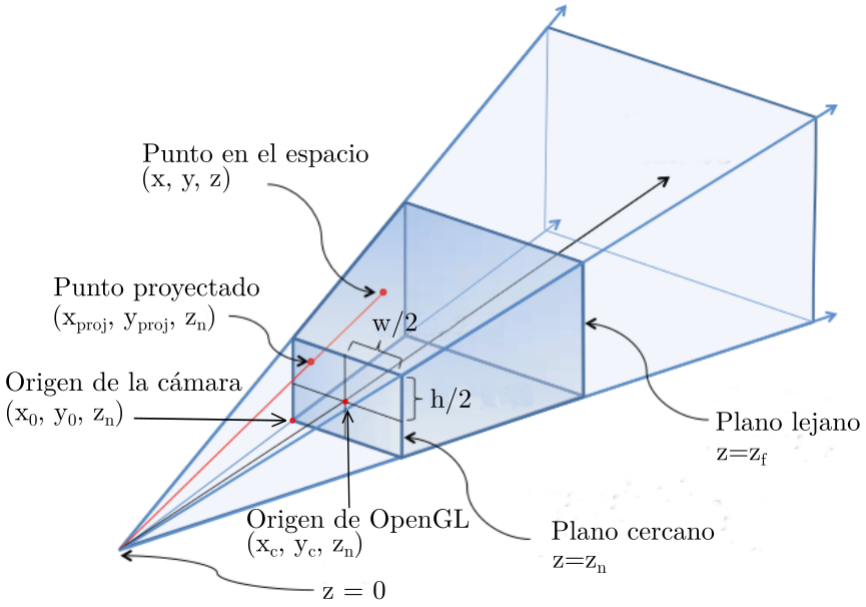


Figura 2.1: Representación gráfica de los diferentes parámetros que componen la matriz de la cámara en OpenGL. Figura obtenida de [25].

La cámara del simulador en formato OpenGL se expresa:

$$\mathbf{K_{GL}} = \begin{pmatrix} \frac{2f_x}{w} & \frac{-2s}{w} & \frac{w-2x_0+2x_c}{w} & 0 \\ 0 & \frac{2f_y}{h} & \frac{-h+2y_0+2y_c}{h} & 0 \\ 0 & 0 & \frac{-z_f-z_n}{z_f-z_n} & \frac{2z_f z_n}{z_f-z_n} \\ 0 & 0 & -1 & 0 \end{pmatrix} \tag{2.2}$$

donde w y h representan el ancho y alto de imagen, respectivamente. x_c e y_c expresan el origen de la cámara en OpenGL, siendo normalmente $(0, 0)$. z_f y z_n representan los planos lejano y cercano de la cámara, respectivamente. En la Figura 2.1 se representan gráficamente los parámetros que componen la matriz de la cámara en OpenGL.

Conociendo todos los valores de la matriz intrínseca de la de la cámara con la que se han tomado las imágenes de entrenamiento de la red neuronal,

$$\mathbf{K_{int}} = \begin{pmatrix} 572{,}41 & 0 & 325{,}26 \\ 0 & 573{,}57 & 242{,}05 \\ 0 & 0 & 1 \end{pmatrix} \tag{2.3}$$

junto con los parámetros adicionales de la cámara,

$$
\begin{aligned}
w &= 640\, pixels \\
h &= 480\, pixels \\
z_n &= 0{,}001 \\
z_f &= 100 \\
x_c &= 0 \\
y_c &= 0
\end{aligned}
\tag{2.4}
$$

se han obtenido los valores de la matriz en OpenGL del simulador, equivalente a la matriz intrínseca de la red:

$$
\mathbf{K_{GL}} = \begin{pmatrix} 1{,}789 & 0 & -0{,}016 & 0 \\ 0 & 2{,}389 & 0{,}085 & 0 \\ 0 & 0 & -1{,}00002 & -0{,}002 \\ 0 & 0 & -1 & 0 \end{pmatrix} \tag{2.5}
$$

Estos parámetros han permitido elaborar un *dataset* propio con imágenes de hígado en distintas posiciones, asegurando que la estimación de la *pose* con estos parámetros de la cámara sea consistente con los mismos parámetros del *dataset* original, empleado para preentrenar la red PVNet.

2.2. Jerarquía de los datos

Para comenzar a elaborar el *dataset*, se ha tenido que replicar la jerarquía de archivos original de la red. En la Figura 2.2 se resume dicha jerarquía utilizada para el posterior entrenamiento de la red con el *dataset* propio.

Las funciones de cada uno de los archivos y carpetas son las siguientes:

- *corners.txt*: Representa los valores máximos y mínimos en los ejes X, Y y Z en coordenadas locales de la malla 3D del hígado. Al representar el resultado, quedarán 8 puntos (las esquinas de una caja) que rodea al hígado. Al realizar el test, estos 8 puntos son los que se representarán visualmente, para poder comprobar si la estimación de la *pose* es correcta.

- *dense_ pts.txt*: Nube de puntos obtenida a partir de la malla 3D del objeto (*.ply) con el programa CloudCompare [26]. A partir de la malla, se han obtenido 100.000 puntos.

- *liver.ply*: Malla 3D del hígado, en formato (*.ply).

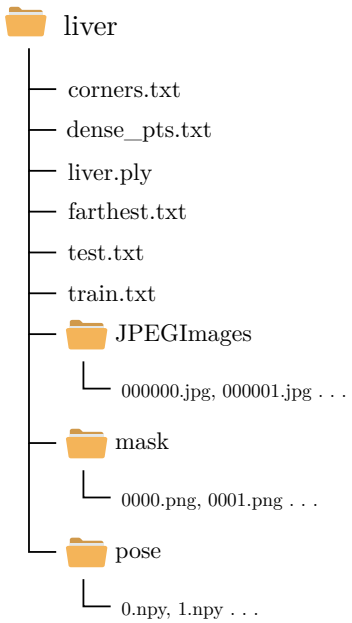


Figura 2.2: Jerarquía de los datos, necesarios para entrenar a la red.

- *farthest.txt*: Contiene los 8 *keypoints* del objeto en coordenadas locales, generados por el algoritmo *Farthest Point Sampling* [27] para su posterior detección con la red neuronal.
- *train/test.txt*: Contienen los nombres de las imágenes que se han utilizado para el entrenamiento y el test de la red, respectivamente.
- *JPEGImages*: Carpeta que contiene las imágenes en formato (*.jpg) del hígado en distintas posiciones.
- *mask*: Carpeta que contiene la segmentación del hígado en cada una de las posiciones, en formato (*.png).
- *pose*: Carpeta que contiene las matrices extrínsecas de cada una de las posiciones del hígado.

2.3. Extracción de datos desde el simulador

Para el entrenamiento se han utilizado imágenes sintéticas de aspecto realista de un hígado con un fondo gris homogéneo, como la representada en la Figura 2.3. Se han obtenido imágenes con diferentes ángulos de giro en los ejes X, Y y Z, para que la red aprenda a reconocer al objeto desde todas las perspectivas posibles.

Para cada una de las imágenes, se ha exportado la máscara o segmentación correspondiente, que se obtiene discriminando el valor RGB (el color, expresado en valores de 0 a 255, y descompuesto en los colores rojo, verde y azul) de cada píxel. Al

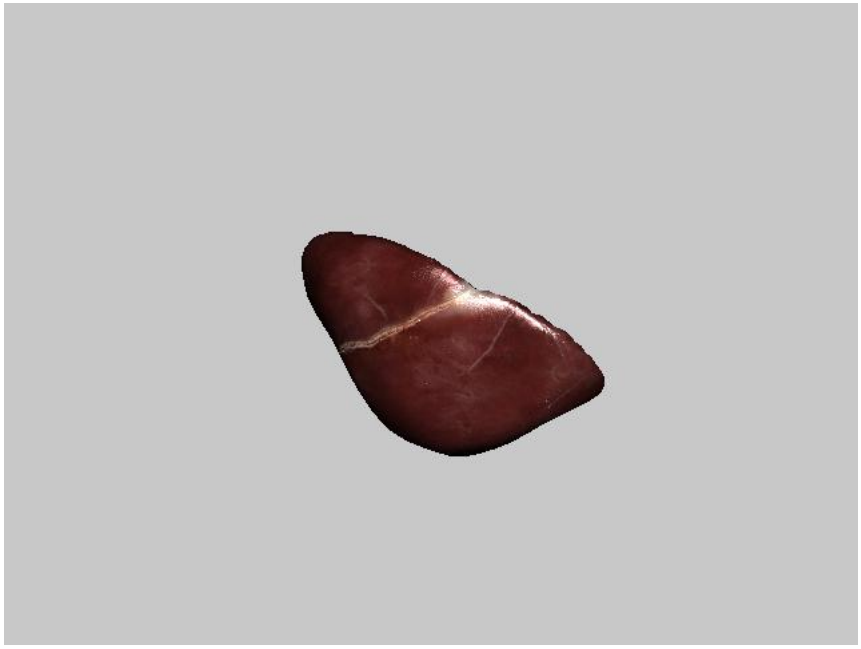


Figura 2.3: Captura del hígado obtenida desde el simulador con fondo homogéneo.

tratarse de un fondo homogéneo, se consideran como parte del hígado aquellos píxeles que no tengan el mismo valor RGB (o color) que el fondo.

Posteriormente, se ha aplicado un filtro de mediana para eliminar el efecto de sal y pimienta que se puede producir en algunas capturas. Este efecto se produce cuando alguno de los píxeles que forman parte del hígado tiene el mismo valor RGB (color) que el fondo homogéneo. Normalmente, y dado que el color del hígado es muy distinto al fondo, esto se va a producir únicamente en píxeles aislados dentro del hígado. Con un filtro de mediana se consigue eliminar estos defectos, creando una máscara homogénea con la que se va a poder entrenar a la red. El resultado de la máscara generada para una captura del simulador se puede observar en la Figura 2.4.

La *pose* se obtiene exportando la matriz extrínseca desde el simulador para cada una de las capturas. El simulador y la red neuronal tienen referencias espaciales diferentes, por lo que es necesario rotar 180^o la matriz extrínseca en el eje X para obtener la *pose* acorde con la referencia de la red neuronal:

$$\mathbf{M_{MV}} = \begin{pmatrix} \mathbf{R_{3x3}} & \mathbf{p_{3x1}} \\ \mathbf{0_{1x3}} & 1 \end{pmatrix} \tag{2.6}$$

donde $\mathbf{M_{MV}}$ representa la *pose* en OpenGL (denominada ModelView Matrix), de dimensiones 4x4. $\mathbf{R_{3x3}}$ representa la rotación del hígado con respecto a la cámara, y $\mathbf{p_{3x1}}$ representa la traslación del hígado con respecto a la cámara. La *pose* en la red neuronal se expresa a partir de una matriz de dimensiones 3x4, por lo que las transformaciones necesarias para obtener esta matriz (denominada $\mathbf{P}$) son las

siguientes:

$$\begin{aligned} {}^{\mathbf{C}}\mathbf{T_O} &= \mathbf{TRotX(180^o)} * \mathbf{M_{MV}} \\ \mathbf{P} &= \begin{pmatrix} \mathbf{R_H} & \mathbf{p_H} \end{pmatrix} \end{aligned} \tag{2.7}$$

donde ${}^{\mathbf{C}}\mathbf{T_O}$ es la matriz homogénea de transformación de la cámara al objeto (el hígado, en este caso), $\mathbf{R_H}$ representa la componente de rotación de la matriz homogénea, de dimensiones 3x3, y $\mathbf{p_H}$ representa la componente de traslación de la matriz, de dimensiones 3x1.

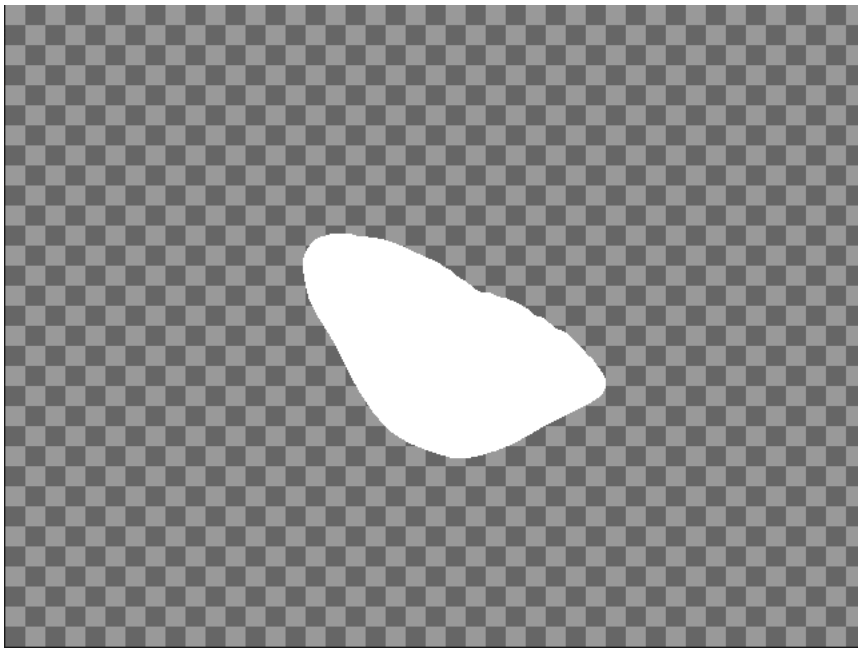


Figura 2.4: Máscara correspondiente a la captura del hígado descrita en la Figura 2.3.

La obtención del *dataset* para un primer entrenamiento se compone de 2666 imágenes del hígado con fondo homogéneo, sin ningún tipo de oclusiones. Se han obtenido capturas del hígado rotado en diferentes posiciones para que la red neuronal sea capaz de reconocerlo en diversos casos.

En la Figura 2.5 se representa el preprocesado de los datos obtenidos del simulador, necesario para que el formato de los archivos sea compatible con la red neuronal.

En la Figura 2.6 se representan ejemplos de las capturas contenidas dentro del *dataset* utilizado para entrenar la red neuronal. Como se puede observar, el objetivo de este primer entrenamiento es que la red sea capaz de reconocer al objeto (hígado) y estime su *pose*.

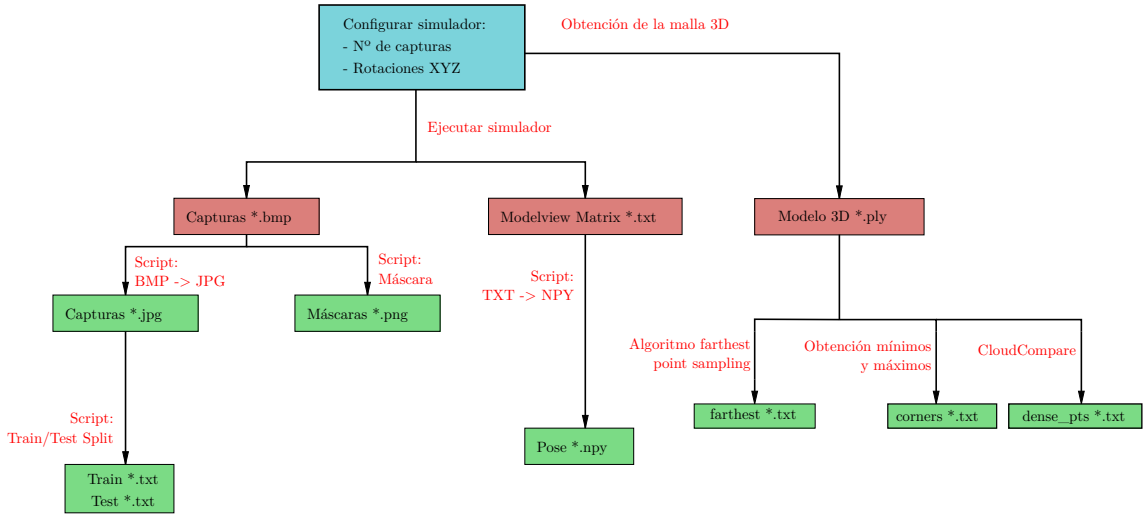


Figura 2.5: Preprocesado de los datos obtenidos del simulador, para adaptar el formato al de la red neuronal.

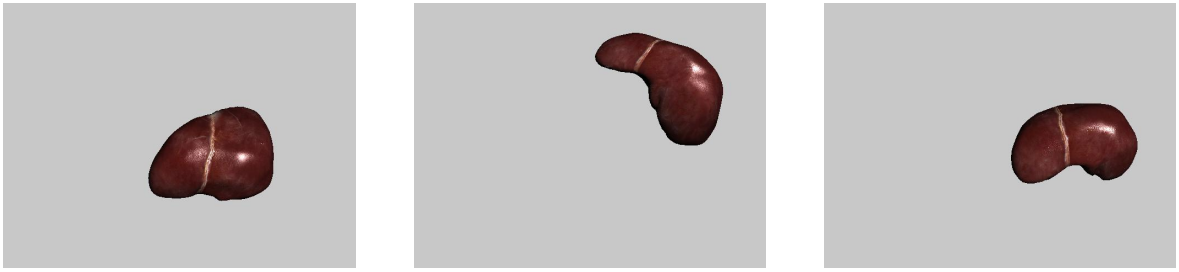


Figura 2.6: Ejemplos de capturas utilizadas para hacer un primer entrenamiento de la red neuronal.

Capítulo 3

Entrenamiento de la red neuronal

3.1. Condiciones de entrenamiento

Previo al entrenamiento, es necesario configurar la red para trabajar con las imágenes específicas generadas anteriormente.

En primer lugar, se ha dividido el conjunto de datos en dos partes, utilizando el 80 % de las imágenes para el entrenamiento de la red (*train*), y el 20 % restante para verificar los resultados (*test*). Mediante esta subdivisión se consigue entrenar el modelo con unos datos, y comprobar sus estimaciones sobre otros datos diferentes, evaluando la capacidad de la red para obtener resultados de forma generalizada.

Se han utilizado dos funciones diferentes para evaluar la capacidad de la red para obtener la *pose* y la segmentación (máscara) del hígado. Estas funciones se denominan funciones de pérdida (o *loss function*). La primera función de pérdida evalúa el error cometido por la red al estimar la máscara del hígado correspondiente (como el descrito en la Figura 2.4). La segunda función de pérdida evalúa el error cometido por la red al estimar los *keypoints*, que se utilizan para calcular la posición del objeto mediante el algoritmo *Perspective-n-Point* (PnP) [12]. En la Figura 3.1 se representa una captura de hígado con los puntos a estimar por la red (*keypoints*), para posteriormente obtener su *pose*.

Durante el entrenamiento se realizan numerosas iteraciones para mejorar la predicción de la red neuronal. Para cada iteración, se utiliza un número determinado de imágenes para realizar la predicción, denominado tamaño de lote (o *batch size*, en inglés). El tamaño de lote utilizado para entrenar la red neuronal es de 5 imágenes, ya que la potencia computacional disponible limita el número de imágenes para cada lote, sin afectar sobre el resultado final del proceso de aprendizaje de la red

El número de repeticiones (o *epochs*, en inglés) se define como el número de veces que la red neuronal recorre el subconjunto entero de datos de entrenamiento para realizar las estimaciones. Para los diferentes entrenamientos que se han realizado, se

ha escogido un número de repeticiones distinto.

Para el entrenamiento del modelo, se ha utilizado un algoritmo deoptimización basado en gradiente descendiente, denominado ADAM [28]. La tasa de aprendizaje es un escalar que multiplica al gradiente en cada iteración, definiendo el cambio en la predicción de la red. La tasa de aprendizaje inicial es de 0,001. Esta tasa disminuye un 50 % cada 5 repeticiones.

Figura 3.1: En azul se representan los *keypoints* a estimar por la red neuronal para una posición determinada del hígado. Estos puntos se utilizan posteriormente para calcular la *pose* del hígado mediante el algoritmo PnP [12].

Para el entrenamiento de la red se ha seguido una estrategia de *transfer learning*. El *transfer learning* consiste en transferir información de una tarea de aprendizaje automático a otra. En este proyecto, para el entrenamiento de la red con las imágenes de los hígados se ha partido de un modelo preentrenado, cuyo objetivo es la detección de objetos en diferentes posiciones, pertenecientes al *dataset* LINEMOD [23]. Se ha elegido utilizar esta estrategia de entrenamiento debido a que el problema es básicamente el mismo, por lo que aprovechamos todo el proceso de aprendizaje previamente realizado.

La decisión de emplear una red previamente entrenada sobre el *dataset* LINEMOD, con una estrategia de *transfer learning*, nos ha permitido emplear un *dataset* bastante reducido, sin necesitar una gran cantidad de datos para entrenar la red neuronal desde cero, reduciendo drásticamente el tiempo de entrenamiento.

3.2. Evolución del entrenamiento

Para un primer entrenamiento se han utilizado capturas del simulador de hígado como las expuestas en la Sección 2.3. El conjunto de imágenes está formado por 2666 capturas del hígado desde todas las perspectivas posibles. En este primer caso, las condiciones de entrenamiento no se aproximan demasiado a las condiciones de uso de la herramienta, pero sirven como punto inicial para obtener los resultados finales. Se han llevado a cabo 40 repeticiones, con el objetivo de obtener resultados con una tasa de error máxima previamente definida.

En la Figura 3.2 se expone un ejemplo de captura utilizada para el entrenamiento, junto con resultados obtenidos para un caso de *test*.

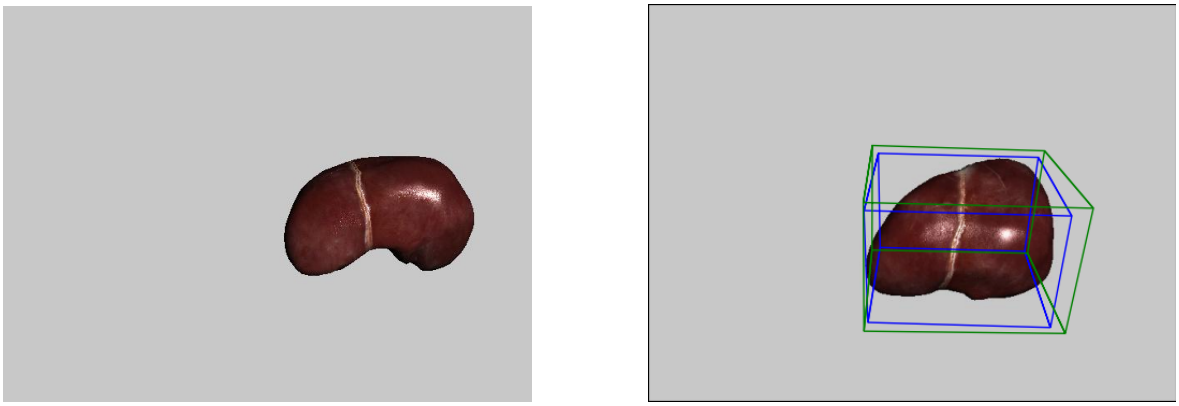


Figura 3.2: A la izquierda se expone una captura utilizada para el primer entrenamiento de la red neuronal. A la derecha, se expone el resultado de la estimación de la *pose* con la red neuronal. La caja azul representa la *pose* verdadera del hígado, mientras que la caja verde representa la estimación de *pose* obtenida mediante la red.

Tras realizar un primer entrenamiento, se ha procedido a validar estadísticamente el modelo, con el objetivo de evitar malas predicciones y comprobar que el entrenamiento de la red se está llevando a cabo correctamente. En el Anexo B se exponen detalles de la validación del modelo realizada con este primer entrenamiento.

El siguiente entrenamiento llevado a cabo parte de la estimación obtenida en el primer entrenamiento. En este caso, se ha elaborado un nuevo conjunto de imágenes con fondos realistas de laparoscopia [3], con el objetivo de entrenar a la red en unas condiciones más cercanas a las condiciones de uso de la herramienta. El nuevo conjunto de imágenes está compuesto por 2923 capturas del hígado, e incluyen fondos que simulan una intervención laparoscópica. Las capturas están tomadas desde numerosos puntos de vista, con el objetivo de que la red neuronal reconozca la *pose* del hígado desde el mayor número de posiciones posibles.

En la Figura 3.3 se expone un ejemplo de captura utilizada para entrenar a la red, junto con una estimación de *pose* realizada por la red neuronal.

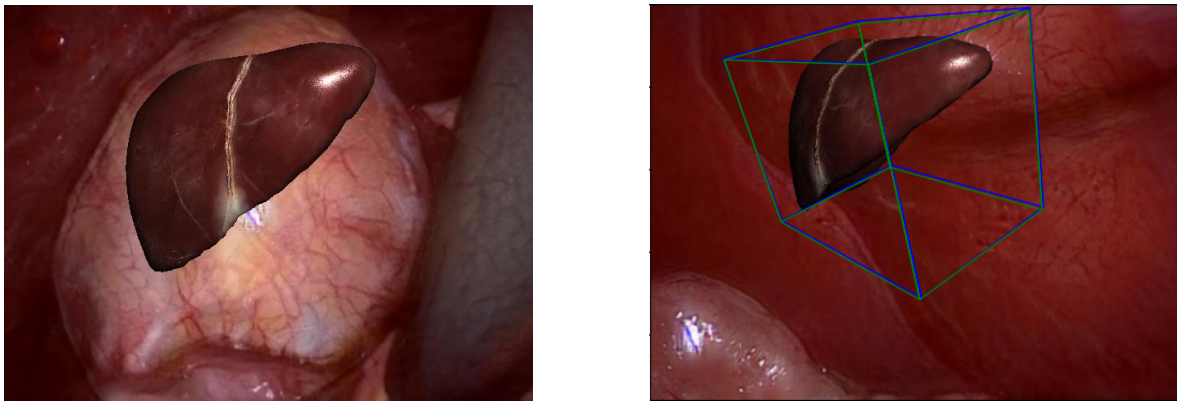


Figura 3.3: A la izquierda se expone una captura utilizada para el entrenamiento de la red neuronal con fondos de laparoscopia. A la derecha, se expone el resultado de la estimación de la *pose* con la red neuronal. La caja azul representa la *pose* verdadera del hígado, mientras que la caja verde representa la estimación de *pose* obtenida mediante la red.

El entrenamiento de la red con este nuevo conjunto de datos se ha dividido en dos partes, con el objetivo de restablecer el valor de la tasa de aprendizaje, y evitar que la red se estanque en algún mínimo local, lo que llevaría a la obtención de resultados con un error permanente. Para el primer entrenamiento, se han llevado a cabo 50 repeticiones, y para el segundo 65 repeticiones. Como se puede comprobar en la Figura 3.3, la estimación de *pose* realizada por la red neuronal tras los dos entrenamientos se aproxima a la posición real del hígado.

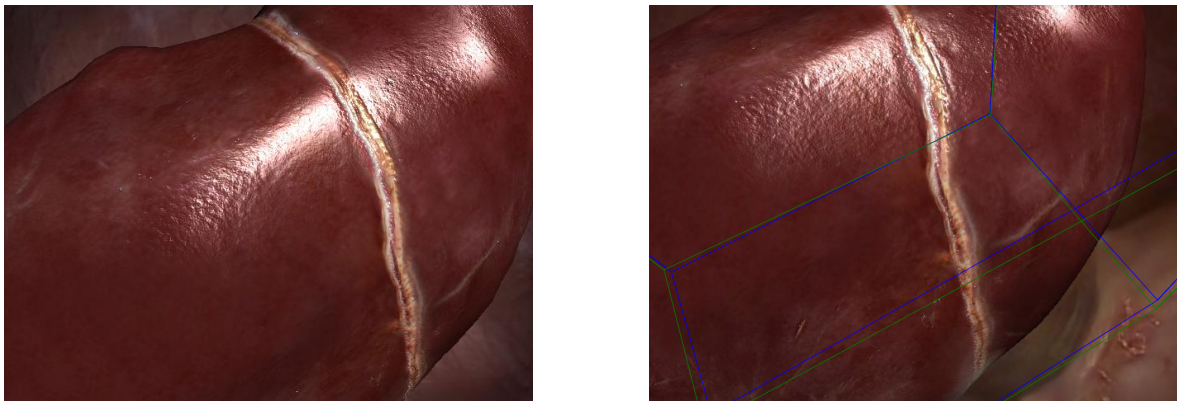


Figura 3.4: A la izquierda se expone una captura utilizada para el entrenamiento de la red en condiciones realistas, con fondo de laparoscopia y el hígado cerca de la cámara. A la derecha, se expone el resultado de estimación de la *pose* con la red neuronal. La caja azul representa la *pose* verdadera del hígado, mientras que la caja verde representa la estimación de *pose* obtenida mediante la red.

El objetivo de estos entrenamientos ha sido comprobar el funcionamiento de la red neuronal tanto con fondos homogéneos como con fondos de laparoscopia, que se aproximan más a las condiciones de uso de la herramienta. Sin embargo, al utilizar una

cámara laparoscópica en condiciones realistas, el órgano se sitúa mucho más cerca de la cámara que en las imágenes anteriores. Además, solamente se puede acceder a la parte frontal del órgano.

Por ello, se han creado dos nuevos conjuntos de imágenes, compuestos por 5100 capturas cada uno de ellos, con la cámara cercana al hígado y solamente obteniendo capturas de la parte frontal, simulando las condiciones de uso de la herramienta. La única diferencia entre los dos conjuntos es que uno de ellos solamente contiene capturas de hígado con fondos homogéneos, y el otro contiene capturas de hígado con fondos de laparoscopia. Este último caso es el más realista y cercano a las condiciones de uso de la herramienta.

El entrenamiento con estos nuevos conjuntos de datos parte del preentrenamiento del gato. Se ha realizado un primer entrenamiento con las imágenes con fondo homogéneo, de 34 repeticiones. El entrenamiento con los fondos de laparoscopia parte de los resultados obtenidos para los fondos homogéneos, y se ha dividido en dos partes, con el objetivo de restablecer la tasa de aprendizaje y evitar los mínimos locales y errores sistemáticos. Para los dos entrenamientos con fondos de laparoscopia, se han realizado 47 repeticiones en el primer entrenamiento, y 48 repeticiones para el segundo. En la Figura 3.4 se expone un ejemplo de captura utilizada para entrenar a la red con fondos de laparoscopia, junto con una estimación de *pose* realizada por la red.

En el siguiente apartado se exponen los resultados obtenidos con el último entrenamiento, que ha sido el que se ha utilizado para diseñar la herramienta, en conjunto con ORB-SLAM.

3.3. Resultados del entrenamiento

Los resultados del último entrenamiento realizado han sido los que se han utilizado para diseñar la herramienta. Este último entrenamiento se ha realizado con el conjunto de imágenes del hígado con fondos de laparoscopia, con la cámara cerca del hígado y utilizando únicamente capturas desde la parte frontal del hígado, simulando las condiciones de uso de la herramienta.

En la Figura 3.5 se exponen las gráficas de entrenamiento de la red, representando diferentes métricas que cuantifican la capacidad de la red durante el entrenamiento.

Las métricas de precisión y recuperación (*precision* y *recall* en inglés, respectivamente) se utilizan en modelos de clasificación. Sin embargo, en [16] se utiliza una métrica que establece la *pose* como correcta si la distancia entre la *pose* predicha por la red y la *pose* verdadera están a una distancia menor a 5 píxeles para la imagen sobre la que se realiza la inferencia. Como se puede observar, tanto la precisión como

la recuperación alcanzan valores cercanos al 100 % a las pocas iteraciones de la red neuronal.

La función de pérdida de segmentación (*segmentation loss* en inglés) evalúa el error cometido por la red al estimar la maścara del hígado. Como se puede observar (al igual que con la precisión y la recuperación), al principio del entrenamiento es algo inestable, pero a las pocas iteraciones ya se consigue un error por debajo del 0,2 %.

Por último, la función de pérdida de vértices (*vertex loss* en inglés) evalúa el error cometido por la red al estimar los *keypoints*, que posteriormente se utilizarán para calcular la *pose* del objeto mediante el algoritmo PnP [12]. La evolución en el error es muy similar a la observada con la segmentación.

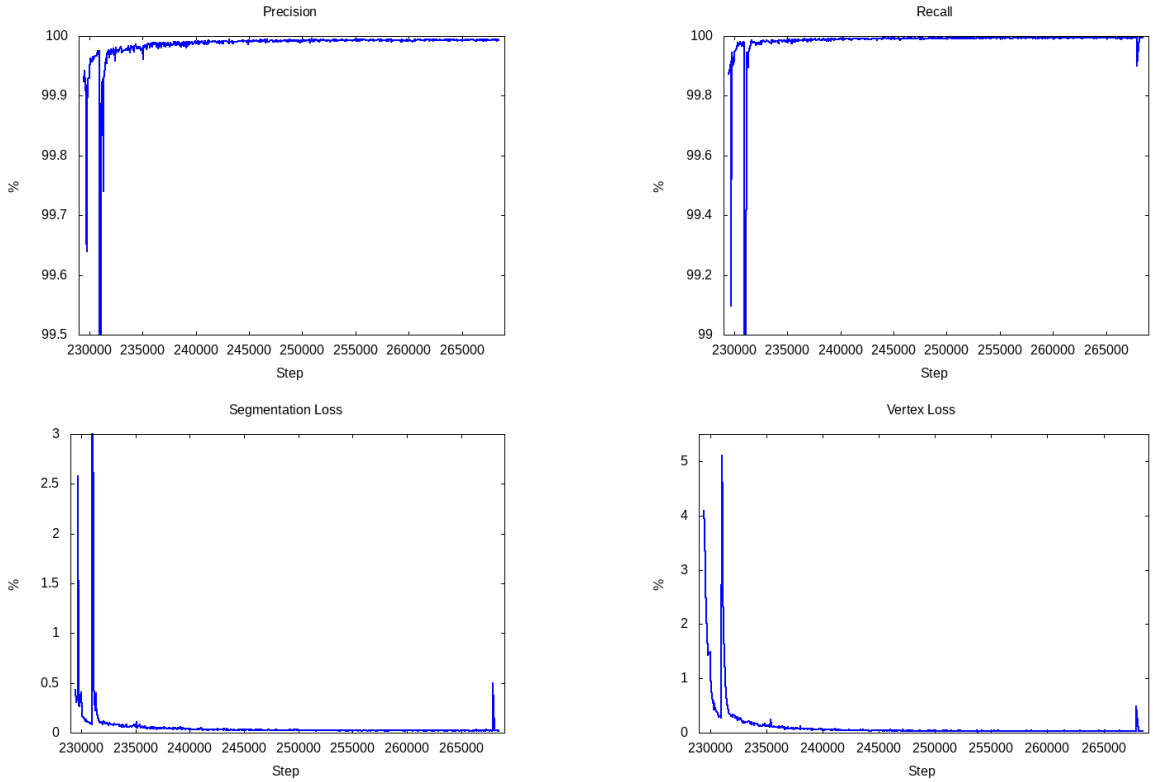


Figura 3.5: Gráficas de entrenamiento de la red neuronal para el último conjunto de imágenes. De izquierda a derecha y de arriba a abajo: Precisión, Recuperación, Función de pérdida de Segmentación y Función de pérdida de Vértices.

En la Tabla 3.1 se exponen las métricas promedio de la última repetición para el entrenamiento (*train*) y el *test*. Como se puede observar, la precisión y la recuperación presentan valores muy similares para el entrenamiento y el test. En cuanto a las funciones de recuperación, el error es bajo en todos los casos, y también se presentan unos valores muy homogéneos tanto para el entrenamiento como para el *test*.

Para un análisis en profundidad de las estimaciones de *pose* realizadas por la red neuronal, se han creado unas nuevas métricas que cuantifican el error de la *pose*

	Train	**Test**
Precision (%)	99,9935627	99,991882
Recall (%)	99,9950886	99,99525
Segmentation Loss (%)	0,0196543	0,022811
Vertex Loss (%)	0,0294318	0,02528

Tabla 3.1: Métricas promedio de la última repetición del entrenamiento y del *test*.

estimada con respecto a la *pose* verdadera. Para ello, se ha obtenido el valor medio de los errores en la traslación y la rotación de los sistemas de coordenadas locales de la *pose* estimada respecto a la *pose* verdadera para el conjunto de imágenes del *test*. Además, se ha calculado la varianza de la traslación y de la rotación para el conjunto de imágenes del *test*. Los detalles de la nueva métrica se especifican en el Anexo C.

En la Tabla 3.2 se exponen los resultados de la nueva métrica sobre todas las imágenes del *test*. Como se puede observar, el error medio en la distancia de la *pose* estimada con respecto a la *pose* verdadera d_{RMS} es de tan solo 0,02 cm. Por tanto, se puede considerar que la traslación de la *pose* estimada es correcta, con un error mínimo. El error medio en la rotación entre la *pose* estimada y la *pose* verdadera es de 12^{o}. El valor de la varianza es más alto, por lo que en el caso de la rotación sí que existe un error considerable.

$\mathbf{d_{RMS}}$ (cm)	0,026078
$\theta_{\mathbf{RMS}}(^{\text{o}})$	12,054376
$\mathbf{d_{s^2}}$	0,000558
$\theta_{\mathbf{s^2}}$	116,595638

Tabla 3.2: Resultados de error en la traslación y la rotación de la *pose* estimada con respecto a la *pose* verdadera para las imágenes del *test*.

Tras analizar las estimaciones realizadas con la red neuronal sobre las imágenes del *test*, se puede afirmar que el error en rotación proviene de una característica propia de la red neuronal, por la cual tiene dificultades a la hora de estimar la *pose* en posiciones simétricas. El hígado visto desde alguna de las perspectivas puede mostrar ciertas simetrías, lo que hace que la red estime la *pose* rotada 180^{o} con respecto a la *pose* verdadera. Aunque esto ocurre únicamente en algunas capturas, la influencia en el uso conjunto con ORB-SLAM es mínima.

Capítulo 4

Integración con ORB-SLAM

Una vez lograda la estimación de la *pose* con la red neuronal, se debe hacer un seguimiento de los puntos del hígado para poder conocer tanto la escena como la posición de la cámara en todo momento y ser capaces de añadir información virtual consistente en el espacio. Para conseguir un seguimiento en tiempo real de la escena es necesario el uso conjunto de la red PVNet con el algoritmo ORB-SLAM [5], el cual permite hacer un seguimiento de los puntos del hígado y de los tejidos que le rodean, creando una nube de puntos en 3D de la escena.

El uso conjunto de la información semántica que aporta la red neuronal y ORB-SLAM permite segmentar los puntos pertenecientes al hígado y obtener su *pose* simultáneamente en tiempo real, lo cual no es posible con el uso por separado de las dos herramientas.

Además, la obtención de una nube de puntos del hígado permitirá desarrollar en un futuro un modelo para el seguimiento del hígado como objeto deformable [22], ya que el hígado es susceptible de sufrir una gran cantidad de deformaciones durante la intervención quirúrgica.

4.1. Nexo entre ORB-SLAM y PVNet

El seguimiento en tiempo real del hígado se consigue mediante el uso conjunto de PVNet y ORB-SLAM. PVNet está escrito en Python, mientras que ORB-SLAM está escrito en C++. Se han valorado diferentes alternativas para conseguir la comunicación entre los dos programas para su uso conjunto.

La alternativa elegida para conectar los dos códigos ha sido ZeroMQ [29], una librería de software libre que utiliza el protocolo TCP para el intercambio de información, y que está disponible para casi todos los lenguajes de programación.

En la Figura 4.1 se representa el esquema de funcionamiento conjunto de PVNet con ORB-SLAM para un fotograma. ORB-SLAM hace uso de la librería OpenCV [30]

para trabajar con las imágenes o vídeos. El primer paso es el envío mediante ZeroMQ del fotograma obtenido por medio de un vídeo almacenado en memoria, o a partir de secuencias en tiempo real (con una webcam o un endoscopio, por ejemplo). Al llegar la imagen a PVNet, se ejecuta la red neuronal y se obtienen las estimaciones de la máscara de segmentación del objeto y la matriz de *pose*. Estos dos elementos se envían de vuelta a ORB-SLAM para aplicar el resultado de la segmentación sobre la nube de puntos 3D, para así fijar aquellos que pertenecen a la geometría del hígado, una información de gran utilidad que solo puede conseguirse gracias al análisis semántico de la red neuronal. Posteriormente, con dicha información ya se puede generar la representación 3D en realidad aumentada sobre el fotograma actual. Estas dos acciones se explican con más detalle en los apartados siguientes.

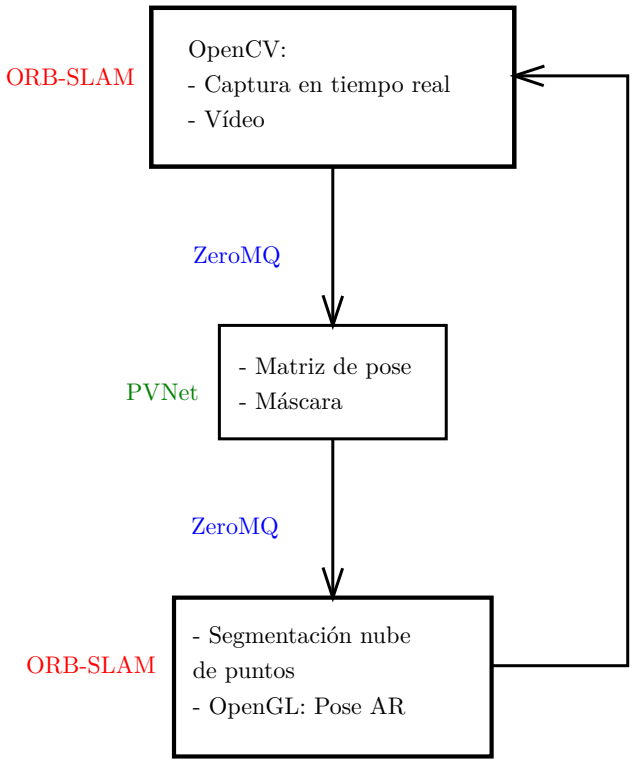


Figura 4.1: Funcionamiento conjunto de PVNet con ORB-SLAM para un fotograma, haciendo uso de la librería ZeroMQ [29].

4.2. Calibración de la webcam

Para validar el método propuesto y reducir las posibles fuentes de error se ha hecho uso de uno de los objetos empleados por los creadores de la red PVNet. Puesto que los autores han compartido la geometría del modelo de un gato empleado en sus experimentos, se ha procedido a imprimirlo mediante una impresora 3D para poder llevar a cabo las pruebas pertinentes.

Todas las imágenes del gato han sido tomadas con una *webcam*. Por tanto, previamente a la grabación de las secuencias ha sido necesario calibrar la cámara. Para ello, se ha utilizado una herramienta de calibración [31], con la que se pueden obtener todos los parámetros de la matriz intrínseca, además de los parámetros de distorsión de la lente de la cámara, necesarios para eliminar la distorsión de los píxeles de las imágenes.

La matriz intrínseca de la *webcam* estimada tras la calibración es:

$$\mathbf{K_{int}} = \begin{pmatrix} 754{,}02 & 0 & 303{,}97 \\ 0 & 757{,}93 & 287{,}23 \\ 0 & 0 & 1 \end{pmatrix} \tag{4.1}$$

Mientras que los diferentes parámetros de corrección de la lente de la cámara son:

$$\begin{aligned} k_1 &= 0{,}0208 \\ k_2 &= 0{,}2468 \\ p_1 &= 0{,}0083 \\ p_2 &= -0{,}0068 \end{aligned} \tag{4.2}$$

donde estos parámetros expresan las distorsiones radiales y tangenciales de la lente de la cámara. En [32] se realiza una descripción más detallada de la calibración de la cámara, además de todos los parámetros utilizados.

4.3. Segmentación de la nube de puntos

Con ORB-SLAM se crea una nube de puntos *sparse* de la escena en tiempo real. Los puntos pertenecientes a la nube se obtienen a partir de la triangulación a lo largo de distintos frames de los puntos detectados por el descriptor ORB. Si éstos son consistentes de fotograma a fotograma, se guardan y se incluyen en la nube de puntos. Sin embargo, la nube de puntos representa únicamente la escena, sin hacer distinción entre los diferentes objetos o características de la imagen capturados.

Mediante el uso conjunto de ORB-SLAM y PVNet se puede conseguir una discriminación en la nube de puntos, distinguiendo el objeto de interés del resto de la escena de forma automática.

Se ha impreso en 3D el modelo de un gato, y se han grabado secuencias de vídeo, con el objetivo de utilizar el método propuesto con una cámara monocular estándar en una escena del mundo real. En la Figura 4.2 se representa el gato impreso en 3D, que servirá como modelo para diseñar el funcionamiento de esta parte del sistema.

Como se ha expuesto anteriormente, PVNet es capaz de estimar la máscara del objeto deseado a partir de la imagen plana capturada. Aplicando esta máscara

Figura 4.2: Modelo del gato impreso en 3D, utilizado para diseñar el conjunto PVNet-ORBSLAM.

a los puntos detectados en cada fotograma por ORB-SLAM se puede hacer una discriminación entre los puntos 3D pertenecientes al objeto y a la escena.

En la Figura 4.3 se representa un fotograma junto a la máscara predicha por la red neuronal, además de la segmentación del mapa de puntos de la escena, obtenida a partir de las máscaras de fotogramas consecutivos. Los puntos rojos representan los puntos de la escena en 3D, mientras que los puntos verdes representan aquellos correspondientes al objeto de interés.

4.4. Seguimiento del objeto

Para realizar el seguimiento del objeto con ORB-SLAM es necesario establecer los sistemas de referencia del objeto (el gato, en este caso), la cámara y la referencia mundo. En la Figura 4.4 se representan los diferentes sistemas de referencia establecidos, junto con las matrices de rotación-traslación utilizadas para realizar la transformación de coordenadas entre los sistemas de referencia.

Al enviar la imagen desde ORB-SLAM hacia PVNet, la red neuronal realiza la inferencia sobre la imagen, devolviendo la matriz de rotación-traslación del objeto con respecto a la cámara ${}^{\mathbf{C}}\mathbf{T}_{\mathbf{O}}$, además de su máscara.

4.4.1. Ajuste de la escala del modelo

El principal reto a resolver en esta parte del proyecto es la escala. La red neuronal ha sido entrenada con imágenes de un gato, cuyo tamaño corresponde con el tamaño de la malla 3D del modelo. En el caso de estudio, se ha impreso el modelo en 3D a una escala diferente a la original, por imposición del tamaño máximo de impresión debido

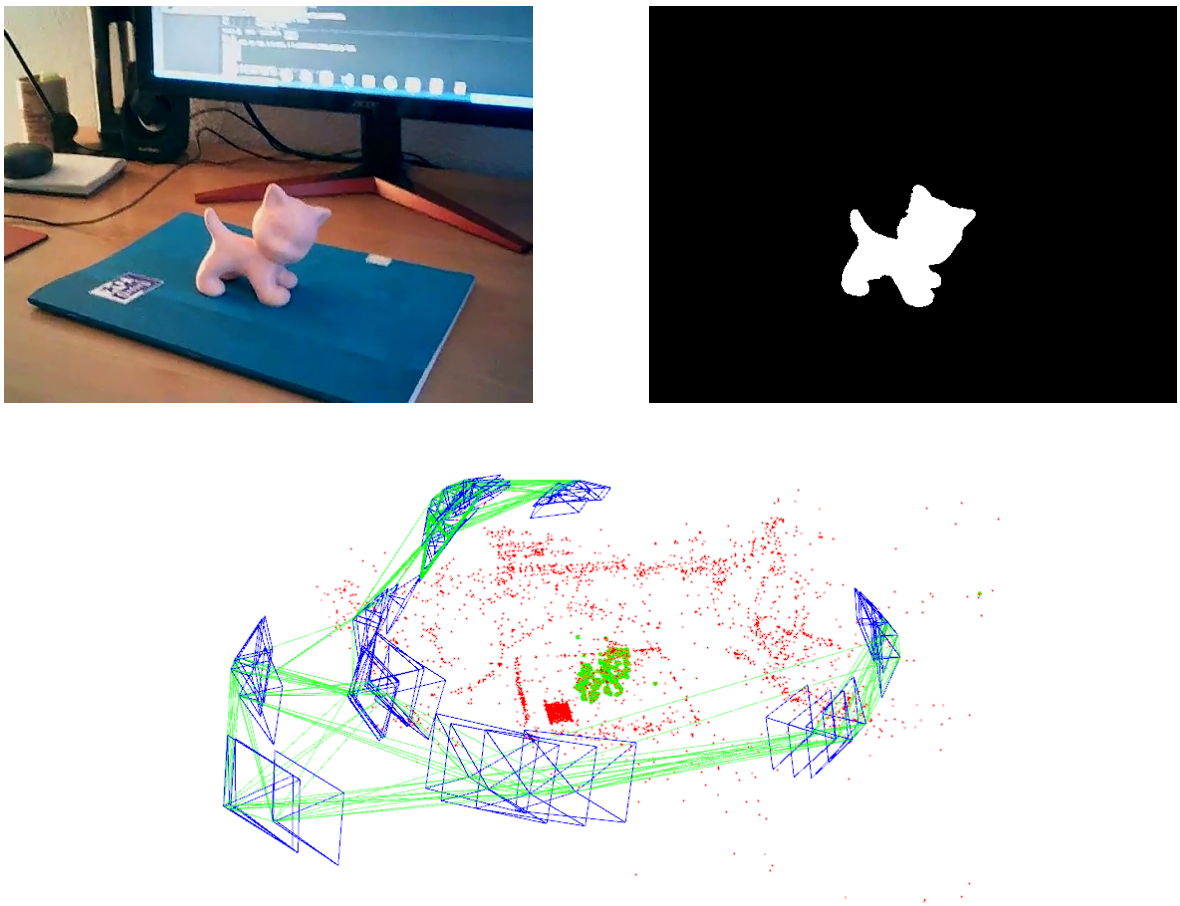


Figura 4.3: Máscara del objeto y segmentación de la nube de puntos. En las dos primeras imágenes se representa el fotograma, junto con la máscara predicha por la red neuronal. En la imagen inferior se representa la nube de puntos de la escena, generada por ORB-SLAM. Los puntos pertenecientes al objeto se representan en verde, mientras que los puntos de la escena se representan en rojo.

a la impresora empleada. La red neuronal estima la posición del objeto sin tener en cuenta dicho cambio de escala debido a la impresión. La resolución de este reto supone una ventaja en el uso final de la herramienta, ya que permite su escalabilidad, pudiendo adaptar las predicciones al tamaño de hígado real.

Para la representación del objeto en realidad aumentada, desde el punto de vista del usuario (cámara) es necesario calcular la matriz de transformación mundo-objeto $^{\mathbf{W}}\mathbf{T}_{\mathbf{O}}$, ya que se asume que el gato no varía de posición en el espacio con el tiempo, por lo que las referencias mundo (fijada de forma aleatoria por ORB-SLAM al iniciar el programa) y objeto son fijas, y por tanto $^{\mathbf{W}}\mathbf{T}_{\mathbf{O}}$ es constante en el tiempo. Para calcular $^{\mathbf{W}}\mathbf{T}_{\mathbf{O}}$ es necesario conocer la matriz de transformación cámara-objeto $^{\mathbf{C}}\mathbf{T}_{\mathbf{O}}$ (obtenida mediante la red neuronal), y la matriz de transformación cámara-mundo $^{\mathbf{C}}\mathbf{T}_{\mathbf{W}}$, calculada para cada fotograma por ORB-SLAM al hacer el seguimiento de los puntos de la escena. La matriz $^{\mathbf{C}}\mathbf{T}_{\mathbf{W}}$ se obtiene para cada fotograma, y la matriz $^{\mathbf{C}}\mathbf{T}_{\mathbf{O}}$

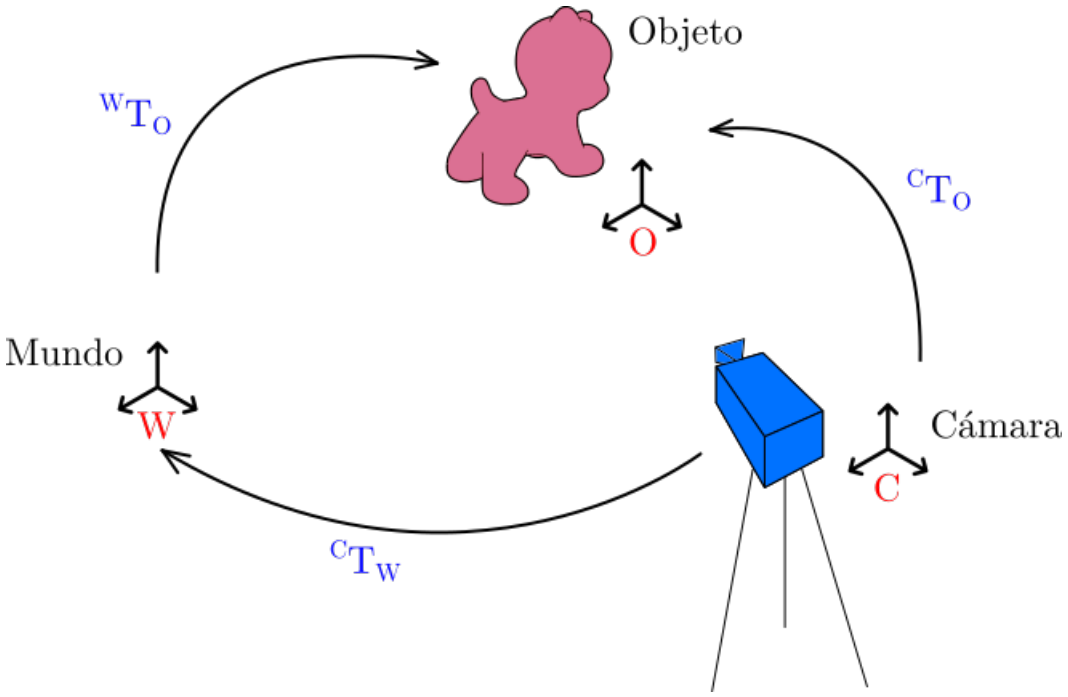


Figura 4.4: Sistemas de referencia mundo (W), cámara (C) y objeto (O). En azul, se representan las matrices de transformación entre los diferentes sistemas de referencia establecidos.

se obtiene cada 4 fotogramas, debido a la elevada potencia de cálculo que requiere la red neuronal. La matriz $^{\mathbf{W}}\mathbf{T}_{\mathbf{O}}$, sin tener en cuenta la escala del objeto, se calcula como:

$$^{\mathbf{W}}\mathbf{T}_{\mathbf{O}} = (^{\mathbf{C}}\mathbf{T}_{\mathbf{W}})^{-1} * {}^{\mathbf{C}}\mathbf{T}_{\mathbf{O}} \tag{4.3}$$

En el caso del gato, al ser el modelo impreso distinto a la malla 3D del modelo, y por la ausencia de información de la profundidad de la escena al utilizar una cámara monocular estándar, se produce un error sistemático en la estimación de la *pose*. Al seguir los puntos del objeto con ORB-SLAM y haber fijado la posición, al cambiar la perspectiva el error aumenta. En la Figura 4.5 se puede observar el error que se produce al fijar la *pose* para un fotograma determinado y enfocar al objeto desde otra perspectiva.

Para la corrección de la *pose* del objeto, es necesario conocer la relación de escala entre el modelo impreso capturado por la cámara y la malla 3D. Aunque el valor de esta constante no va a ser la solución final del problema, sí que disminuye el error a la hora de corregir la *pose* del objeto. En el caso del hígado, las medidas reales del órgano se pueden obtener a partir de imágenes preoperatorias como resonancias magnéticas (MRI) o tomografías computerizadas (CT).

En la Figura 4.6 se representa la medida que se ha tomado como referencia para establecer la escala del gato, medida sobre la malla (expresada en metros) y sobre el modelo impreso en 3D con una regla. Las medidas obtenidas para la malla y el modelo

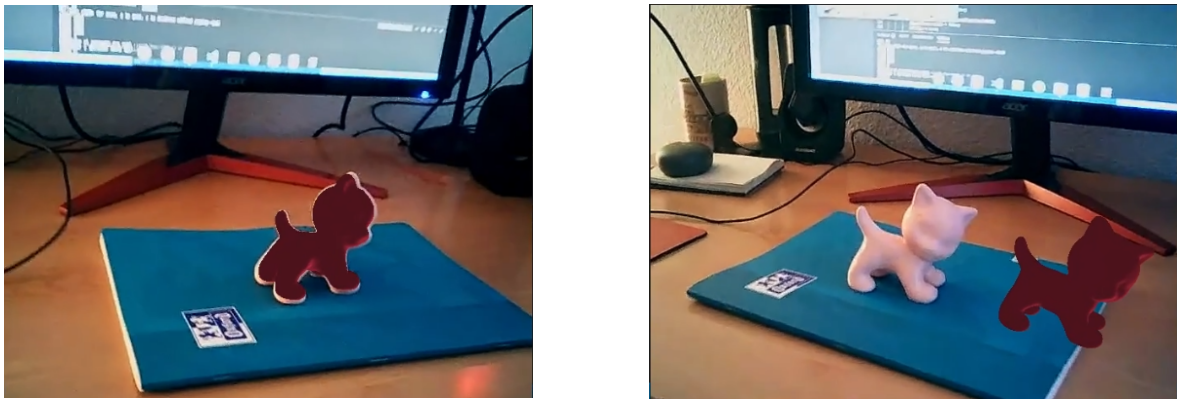


Figura 4.5: Error de estimación de la *pose* al utilizar la red neuronal en conjunto con ORB-SLAM. En la imagen izquierda se representa el fotograma en el que se ha fijado la *pose*, con el modelo 3D representado en realidad aumentada. En la imagen derecha se representa el objeto desde otra perspectiva, tras haber fijado la posición.

impreso 3D son, respectivamente:

$$\begin{aligned} d_{malla} &= 7{,}55cm \\ d_{real} &= 5{,}9cm \end{aligned} \tag{4.4}$$

Por tanto, la relación de escala k_{sc} es:

$$k_{sc} = \frac{d_{real}}{d_{malla}} = 0{,}7814 \tag{4.5}$$

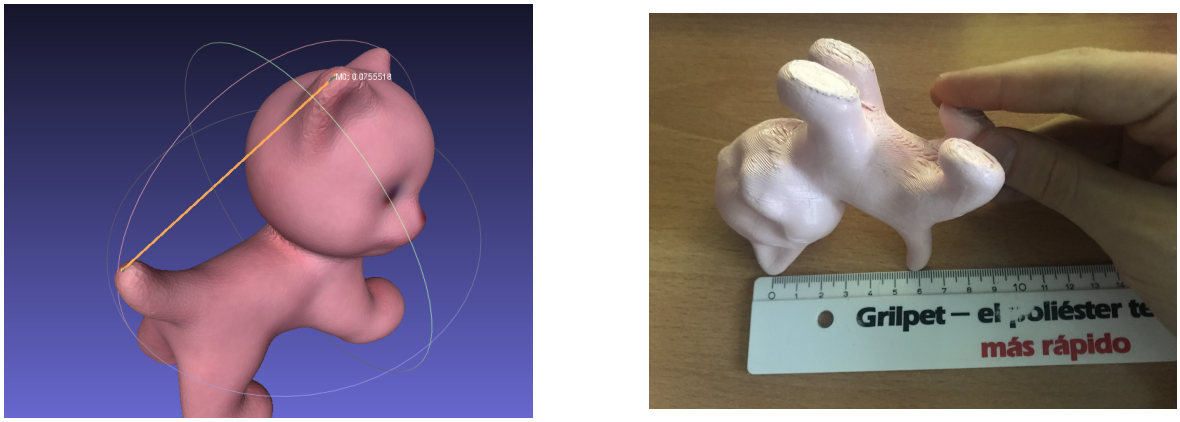


Figura 4.6: Medida tomada para establecer diferencias de escalas entre el objeto virtual (usado para entrenar la red) y el objeto impreso. A la izquierda, medida de la distancia en metros sobre la malla 3D del gato. A la derecha, medida de la distancia sobre el objeto impreso en 3D con una regla.

Como se puede observar en la Figura 4.5, la red neuronal estima correctamente la rotación del gato, pero falla a la hora de estimar la traslación. Para corregir la escala,

hay que aplicar la relación de escala a la traslación de la matriz de transformación cámara-objeto ${}^{\mathbf{C}}\mathbf{T}_{\mathbf{O}}$:

$$ {}^{\mathbf{C}}\mathbf{T}_{\mathbf{O}} = \begin{pmatrix} \mathbf{R_{3x3}} & k_{sc} * \mathbf{p_{3x1}} \\ \mathbf{0_{1x3}} & 1 \end{pmatrix} \tag{4.6} $$

4.4.2. Corrección de la pose

A partir de este punto, se asume que a la matriz ${}^{\mathbf{C}}\mathbf{T}_{\mathbf{O}}$ se le ha aplicado la relación de escala k_{sc} para todos los fotogramas. Conocer el valor de k_{sc} ayuda a disminuir el error al estimar la *pose* del objeto, pero no constituye la solución final del problema.

Al trabajar con cámaras monoculares estándar (como una webcam o un endoscopio), no existe información acerca de la profundidad de imagen. Por tanto, con la información que se tiene de las matrices de transformación ${}^{\mathbf{C}}\mathbf{T}_{\mathbf{W}}$ y ${}^{\mathbf{C}}\mathbf{T}_{\mathbf{O}}$ se puede plantear el problema de encontrar la verdadera escala como un problema de triangulación. Al observar al objeto desde dos perspectivas distintas, la red neuronal genera dos matrices ${}^{\mathbf{C}}\mathbf{T}_{\mathbf{O,1}}$ y ${}^{\mathbf{C}}\mathbf{T}_{\mathbf{O,2}}$ diferentes. En la Figura 4.7 se representan los diferentes sistemas de referencia establecidos para plantear el problema. Si a las matrices de transformación cámara-objeto (ya escaladas con k_{sc}) se les vuelve a modificar las componentes de traslación $\mathbf{p_1}$ y $\mathbf{p_2}$, multiplicándolas por una nueva constante k_p, se generan nuevas posiciones para el objeto, que varía en profundidad con respecto a la posición de la cámara:

$$ {}^{\mathbf{C}}\mathbf{T}_{\mathbf{O,1}} = \begin{pmatrix} \mathbf{R_{1,3x3}} & k_p * \mathbf{p_{1,3x1}} \\ \mathbf{0_{1x3}} & 1 \end{pmatrix} \tag{4.7} $$

$$ {}^{\mathbf{C}}\mathbf{T}_{\mathbf{O,2}} = \begin{pmatrix} \mathbf{R_{2,3x3}} & k_p * \mathbf{p_{2,3x1}} \\ \mathbf{0_{1x3}} & 1 \end{pmatrix} \tag{4.8} $$

En la Figura 4.8 se representan las posiciones del gato desde dos perspectivas diferentes al variar el valor de k_p. Los rectángulos verdes representan las cámaras para los dos fotogramas, y el sistema de referencia representa la referencia mundo. Los gatos de color rojo se han obtenido para un valor de k_p determinado, y los gatos en color azul para otro valor de k_p. Los diferentes valores de k_p modifican las matrices de transformación cámara-objeto ${}^{\mathbf{C}}\mathbf{T}_{\mathbf{O,1}}$ y ${}^{\mathbf{C}}\mathbf{T}_{\mathbf{O,2}}$, por lo que al calcular la posición del objeto respecto a la referencia mundo ${}^{\mathbf{W}}\mathbf{T}_{\mathbf{O}}$ se obtienen diferentes posiciones para el gato.

Como se puede observar en la Figura 4.8, se generan dos trayectorias (representadas por dos líneas rojas) al variar el valor de k_p para una posición determinada de la cámara. Para dos fotogramas, estas dos trayectorias se cruzan en un punto determinado del espacio. Idealmente, tanto la matriz de transformación que determina la *pose* del

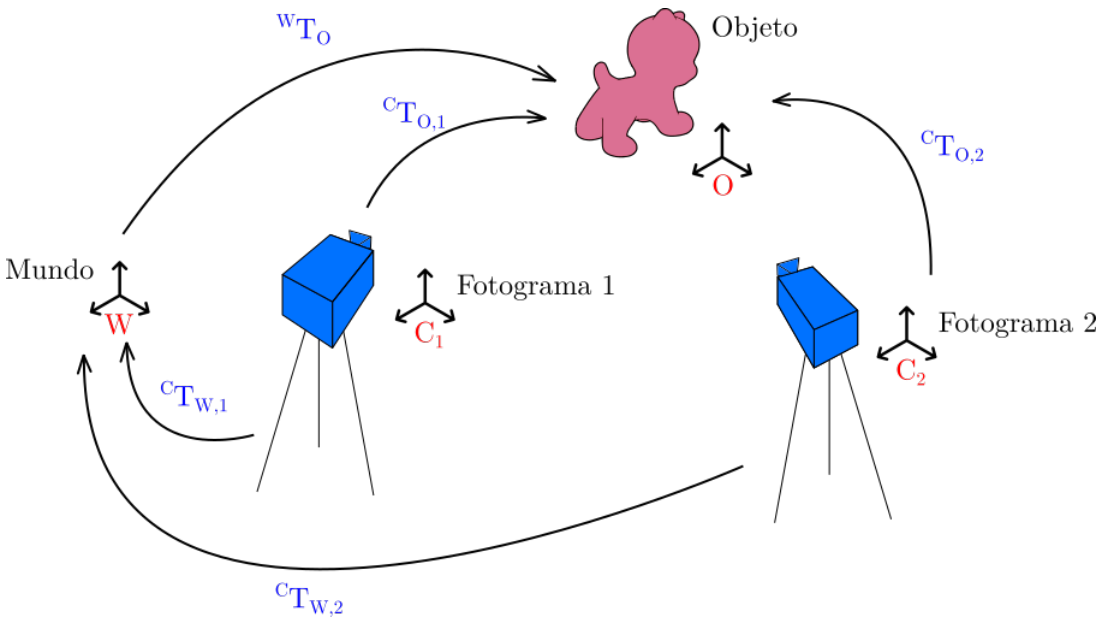


Figura 4.7: Sistemas de referencia establecidos para realizar la corrección de *pose* a partir de dos fotogramas: Referencia mundo (W), fotograma 1 (C_1), fotograma 2 (C_2) y objeto O. En azul se representan las matrices de transformación correspondiente entre todos los sistemas de referencia.

objeto respecto a la referencia mundo ${}^{\mathbf{W}}\mathbf{T}_{\mathbf{O}}$ como el valor de k_p se obtienen mediante la resolución de un sistema de ecuaciones, partiendo de dos fotogramas distintos que cumplen la condición de paralaje (es decir, que existe cierta distancia y cambio de perspectiva de la cámara entre los dos fotogramas).

En el caso de estudio, la red neuronal siempre estima la *pose* para cada fotograma con algo de ruido, por lo que en la gran mayoría de los casos las trayectorias de posición del objeto al variar k_p no se cruzan. En este caso, el sistema de ecuaciones planteado no tiene solución. Para obtener una buena aproximación de ${}^{\mathbf{W}}\mathbf{T}_{\mathbf{O}}$ es necesario utilizar un optimizador, que minimice la distancia entre las posiciones del objeto en dos fotogramas distintos para un valor de k_p determinado. Para el cálculo de la distancia se ha utilizado la métrica de la traslación entre dos sistemas de referencia distintos, descrita en el Anexo C.

El algoritmo utilizado para minimizar la distancia entre dos posiciones del objeto obtenidas en dos fotogramas distintos para un valor determinado de k_p es el algoritmo de minimización Nelder-Mead [33] [34]. Se trata de un algoritmo de optimización sin derivadas, para el que la convergencia en este problema es muy rápida. Se ha elegido este algoritmo porque al trabajar con distintas expresiones matriciales era mucho más sencillo plantear la función a optimizar sin desarrollar todas las derivadas. Además, al trabajar únicamente con una variable k_p se trata de un problema bastante estable.

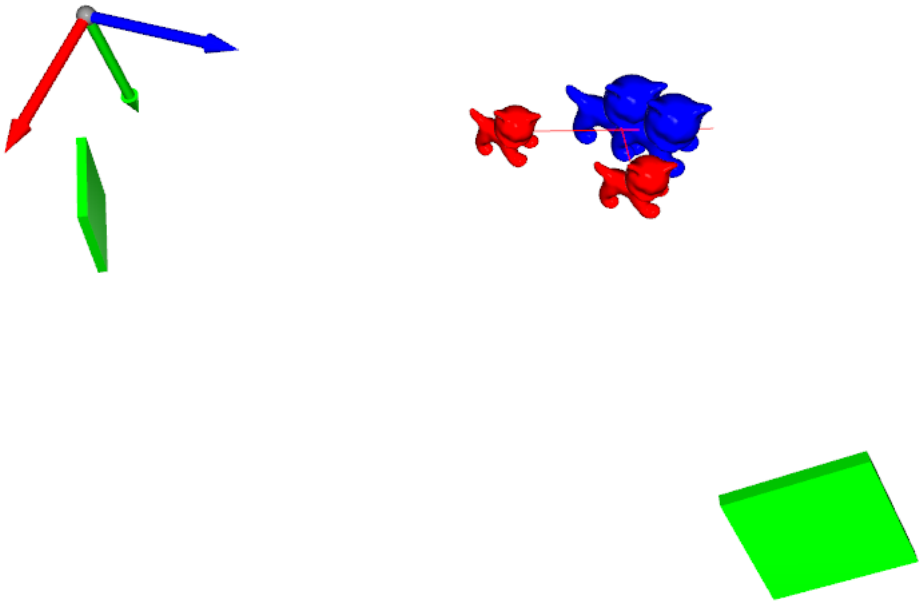


Figura 4.8: Representación de la posición de los gatos desde dos perspectivas distintas para valores de k_p distintos. Las cámaras se representan con rectángulos verdes. En azul se representan los gatos para un valor de k_p determinado, y su proyección a partir de las matrices ${}^{\mathbf{C}}\mathbf{T}_{\mathbf{O}}$ desde las dos perspectivas. En rojo se representan los gatos para un valor de k_p distinto al anterior. Las líneas rojas representan la trayectoria de los gatos al cambiar el valor de k_p para cada una de las perspectivas.

El problema de optimización planteado es el siguiente:

$$min(d_p(k_p)), \qquad k_p \in (0, \infty) \tag{4.9}$$

Es decir, el objetivo es minimizar la distancia d_p entre dos posiciones del objeto ${}^{\mathbf{W}}\mathbf{T}_{\mathbf{O}}$ calculadas desde dos fotogramas distintos para un mismo valor de k_p.

Para un valor de k_p determinado, se calculan las dos matrices de transformación cámara-objeto ${}^{\mathbf{C}}\mathbf{T}_{\mathbf{O},\mathbf{1}}$ y ${}^{\mathbf{C}}\mathbf{T}_{\mathbf{O},\mathbf{2}}$, modificando las componentes de traslación, tal y como se describe en 4.7 y 4.8. A partir de las nuevas matrices de transformación cámara-objeto se calcula la matriz de transformación del objeto con respecto a la referencia mundo. Al trabajar con dos fotogramas, a estas matrices de transformación se les denomina ${}^{\mathbf{W}}\mathbf{T}_{\mathbf{O},\mathbf{1}}$ y ${}^{\mathbf{W}}\mathbf{T}_{\mathbf{O},\mathbf{2}}$, y se calculan a partir de las matrices de transformación mundo-cámara ${}^{\mathbf{C}}\mathbf{T}_{\mathbf{W},\mathbf{1}}$ y ${}^{\mathbf{C}}\mathbf{T}_{\mathbf{W},\mathbf{2}}$ para cada fotograma, obtenidas por medio de ORB-SLAM:

$${}^{\mathbf{W}}\mathbf{T}_{\mathbf{O},\mathbf{1}} = ({}^{\mathbf{C}}\mathbf{T}_{\mathbf{W},\mathbf{1}})^{-1} * {}^{\mathbf{C}}\mathbf{T}_{\mathbf{O},\mathbf{1}} \tag{4.10}$$

$${}^{\mathbf{W}}\mathbf{T}_{\mathbf{O},\mathbf{2}} = ({}^{\mathbf{C}}\mathbf{T}_{\mathbf{W},\mathbf{2}})^{-1} * {}^{\mathbf{C}}\mathbf{T}_{\mathbf{O},\mathbf{2}} \tag{4.11}$$

La distancia entre las dos posiciones obtenidas por las dos matrices de transformación ${}^{\mathbf{W}}\mathbf{T}_{\mathbf{O},\mathbf{1}}$ y ${}^{\mathbf{W}}\mathbf{T}_{\mathbf{O},\mathbf{2}}$ se calcula con la métrica de traslación definida en el Anexo C, y es la función a minimizar a partir del optimizador Nelder-Mead:

$$d_p({}^{\mathbf{W}}\mathbf{T}_{\mathbf{O},\mathbf{1}}, {}^{\mathbf{W}}\mathbf{T}_{\mathbf{O},\mathbf{2}}) \tag{4.12}$$

Una vez obtenida la constante de escalado k_p para la cual la distancia entre las dos posiciones del objeto es mínima, se vuelve a calcular la matriz ${}^{\mathbf{W}}\mathbf{T}_{\mathbf{O}}$ para obtener la mejor aproximación de la *pose* verdadera del objeto. Como la distancia entre la posición de los fotogramas es muy pequeña, el cálculo de la matriz ${}^{\mathbf{W}}\mathbf{T}_{\mathbf{O}}$ se puede realizar indistintamente con cualquiera de los dos fotogramas. Se ha elegido el primer fotograma para realizar el cálculo de ${}^{\mathbf{W}}\mathbf{T}_{\mathbf{O}}$:

$${}^{\mathbf{W}}\mathbf{T}_{\mathbf{O}} = ({}^{\mathbf{C}}\mathbf{T}_{\mathbf{W},\mathbf{1}})^{-1} * {}^{\mathbf{C}}\mathbf{T}_{\mathbf{O},\mathbf{1}} \tag{4.13}$$

El último paso es escalar el modelo 3D, para que al representar el modelo en realidad aumentada coincida la *pose* y el tamaño del objeto con la imagen capturada con la cámara. Para escalar el objeto, es necesario operar las coordenadas de los nodos del modelo 3D del gato (coordenadas locales) por la k_p obtenida anteriormente y por la relación de escala entre el modelo impreso en 3D y la malla k_{sc}, para posteriormente posicionar el objeto ya escalado con la matriz de transformación mundo-objeto ${}^{\mathbf{W}}\mathbf{T}_{\mathbf{O}}$:

$$\mathbf{P}_{\mathbf{esc}} = k_p * k_{sc} * \mathbf{P} \tag{4.14}$$

Donde $\mathbf{P}$ son las coordenadas locales del modelo 3D del objeto, y $\mathbf{P}_{\mathbf{esc}}$ las coordenadas del objeto tras aplicar el escalado. Para posicionar al objeto en realidad aumentada sobre la imagen, hay que realizar la transformación mundo-objeto sobre las coordenadas locales del modelo 3D escalado:

$$\mathbf{P}_{\mathbf{AR}} = {}^{\mathbf{W}}\mathbf{T}_{\mathbf{O}} * \mathbf{P}_{\mathbf{esc}} \tag{4.15}$$

Donde $\mathbf{P}_{\mathbf{AR}}$ se define como los puntos del modelo 3D tras realizar las transformaciones mundo-objeto, que serán los que se representen en realidad aumentada sobre la imagen.

Para comprobar la estimación de *pose* sobre el gato y el hígado tras estos pasos, se han superpuesto las mallas 3D del gato y del hígado sobre la imagen capturada por la webcam (en el caso del gato) y sobre el vídeo del simulador quirúrgico con fondo homogéneo del hígado. En la Figura 4.9 se representa al gato en realidad aumentada desde diferentes perspectivas tras realizar las correcciones descritas anteriormente. En la Figura 4.10 se representa al hígado en realidad aumentada. Se puede observar como

las correcciones aplicadas no son suficientes para obtener una buena aproximación de *pose* y escala.

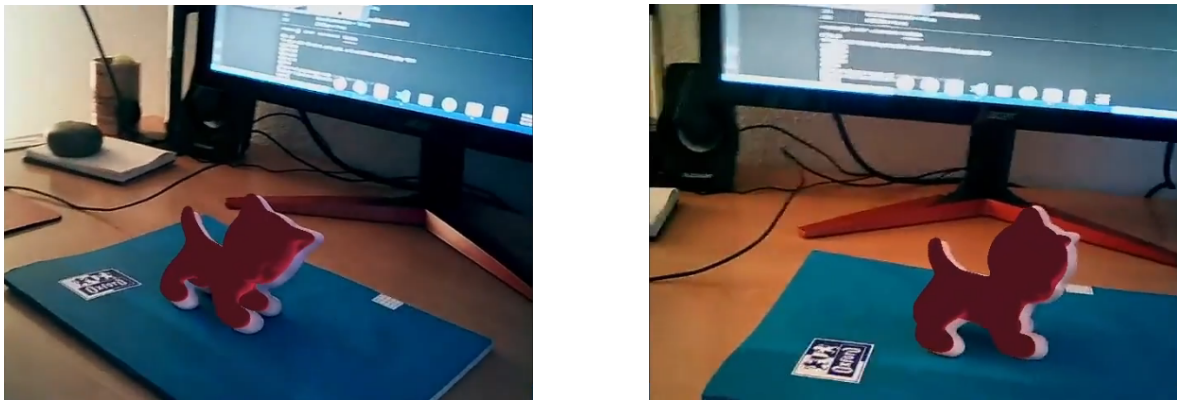


Figura 4.9: Representación del modelo del gato en realidad aumentada sobre un vídeo capturado con la webcam. Se puede observar cómo la posición del gato es consistente desde dos perspectivas distintas.

Para el modelo del gato, la aproximación de la *pose* en realidad aumentada es lo suficientemente buena tras aplicar el ajuste de la escala y la corrección de la *pose* con el optimizador.

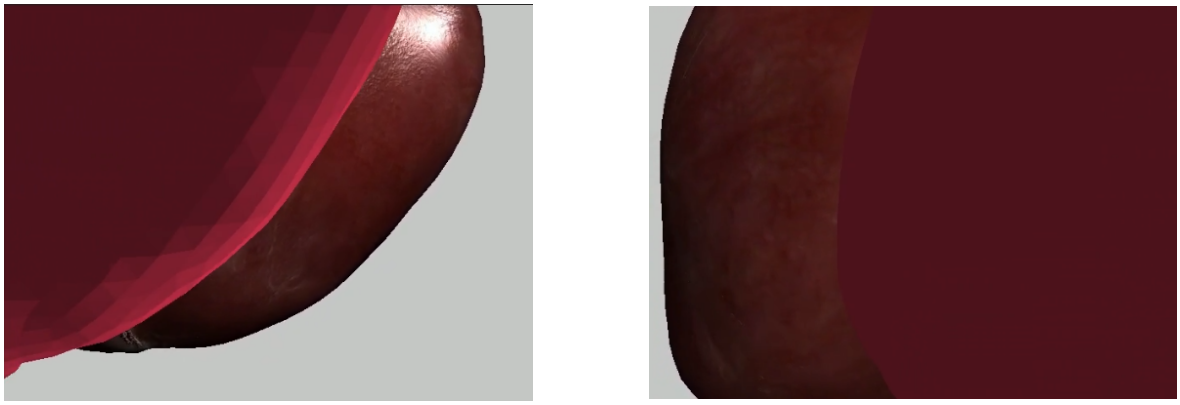


Figura 4.10: Representación del modelo del hígado sobre un vídeo obtenido con el simulador quirúrgico, con fondo homogéneo. En este caso, la posición del hígado es mucho menos precisa, por lo que será necesario realizar unos ajustes adicionales.

Sin embargo, en el caso del hígado es necesario realizar un paso adicional para obtener una buena aproximación de *pose* y poder representar en realidad aumentada tumores y venas sobre el hígado con precisión.

4.4.3. Registro del modelo

El registro y el escalado del modelo con respecto a la nube de puntos es necesario para el hígado. La secuencia que se ha utilizado para diseñar el comportamiento de la herramienta sobre el hígado lo sitúa muy cerca de la cámara, de tal forma que la escala obtenida por medio del optimizador es muy grande, y la transformación espacial necesaria para posicionar el hígado es muy brusca, dando lugar a un error importante si se compara con la secuencia del gato.

Para obtener la escala y la posición del hígado de forma precisa, se han generado unas cajas que engloban a la malla 3D del hígado y a la nube de puntos, se ha calculado un valor global para escalar la malla con respecto a la nube de puntos, y se han alineado las cajas, tomando como referencia una de las esquinas.

El primer paso ha sido calcular los valores máximos y mínimos de los tres ejes con los nodos de la malla 3D del hígado, utilizando las coordenadas del modelo con el escalado y posicionado $\mathbf{P_{AR}}$ de la Sección 4.4.2:

$$\begin{aligned} x_{min,m} &= min(\mathbf{P_{AR}} * \mathbf{i}^T) & x_{max,m} &= max(\mathbf{P_{AR}} * \mathbf{i}^T) \\ y_{min,m} &= min(\mathbf{P_{AR}} * \mathbf{j}^T) & y_{max,m} &= max(\mathbf{P_{AR}} * \mathbf{j}^T) \\ z_{min,m} &= min(\mathbf{P_{AR}} * \mathbf{k}^T) & z_{max,m} &= max(\mathbf{P_{AR}} * \mathbf{k}^T) \end{aligned} \tag{4.16}$$

donde $\mathbf{i}$, $\mathbf{j}$ y $\mathbf{k}$ corresponden a los vectores unitarios en las direcciones X, Y y Z, respectivamente. $x_{min,m}$ y $x_{max,m}$ corresponden a los valores mínimos y máximos de la malla en X , $y_{min,m}$ y $y_{max,m}$ corresponden a los valores mínimos y máximos de la malla en Y y $z_{min,m}$ y $z_{max,m}$ corresponden a los valores mínimos y máximos de la malla en Z.

El siguiente paso es calcular el centroide y realizar un filtrado radial sobre la nube de puntos del hígado, para reducir los nodos redundantes y conseguir una buena estimación de los valores máximos y mínimos en las direcciones X, Y y Z. Esto es muy necesario en el caso del hígado, ya que la secuencia de vídeo recorre solamente la parte frontal del órgano, simulando las condiciones de uso de la herramienta, por lo que un buen filtrado es fundamental para obtener unos resultados precisos.

Para realizar el filtrado, se ha utilizado la librería PCL [35]. El centroide se calcula como:

$$\begin{aligned} x_c &= \frac{\sum(\mathbf{P_H} * \mathbf{i}^T)}{N} \\ y_c &= \frac{\sum(\mathbf{P_H} * \mathbf{j}^T)}{N} \\ z_c &= \frac{\sum(\mathbf{P_H} * \mathbf{k}^T)}{N} \end{aligned} \tag{4.17}$$

donde x_c, y_c y z_c son las coordenadas X, Y y Z del centroide, $\mathbf{P_H}$ son los puntos pertenecientes al hígado en la nube de puntos, y N es el número de puntos pertenecientes al hígado dentro de la nube de puntos.

Una vez calculado el centroide, se establece la condición de filtrado, que consiste en calcular la distancia euclídea desde todos los nodos del hígado en la nube de puntos al centroide. Aquellos puntos que estén a una distancia mayor a un valor determinado, se descartan como pertenecientes al hígado (filtrado radial). En la Figura 4.11 se representa la nube de puntos original del hígado, con su centroide en rojo (imagen izquierda), y la nube de puntos tras el filtrado radial (imagen derecha).

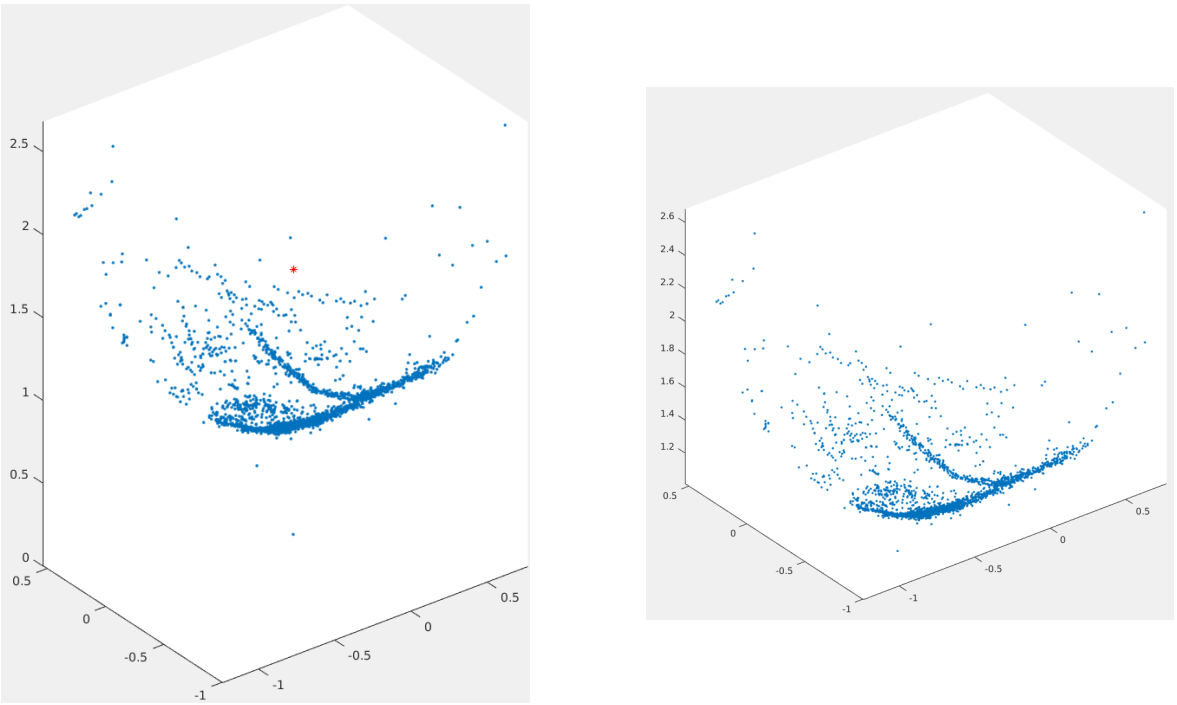
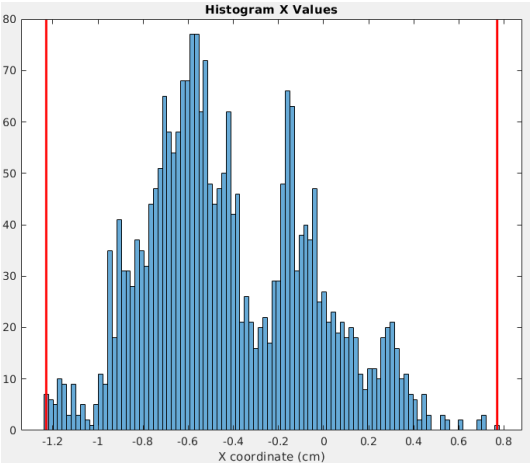


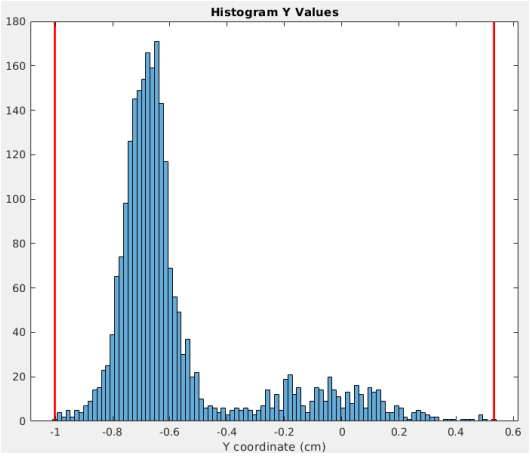
Figura 4.11: Nube de puntos del hígado antes (izquierda) y después de aplicar un filtrado radial (derecha). En la imagen izquierda, el punto rojo representa el centroide de la nube de puntos.

Tras el filtrado radial, se crean tres histogramas con la distribución de los valores en X, Y y Z de los puntos pertenecientes a la nube de puntos. Ajustando los niveles de ruido, se establecen unos límites mínimos y máximos para las tres coordenadas. Estos límites son los que se utilizan para crear la caja que rodea a la nube de puntos del hígado. En la Figura 4.12 se representan los histogramas, agrupando los valores de las coordenadas por rangos equidistantes. En rojo, se representan los límites establecidos mediante los niveles de ruido. Todos los nodos que se encuentran entre estos límites se consideran como pertenecientes al hígado. Para las coordenadas X e Y, se han establecido niveles de ruido que consideran a casi todos los valores como pertenecientes al hígado. En el caso de la coordenada Z, los niveles de ruido son más restrictivos, por

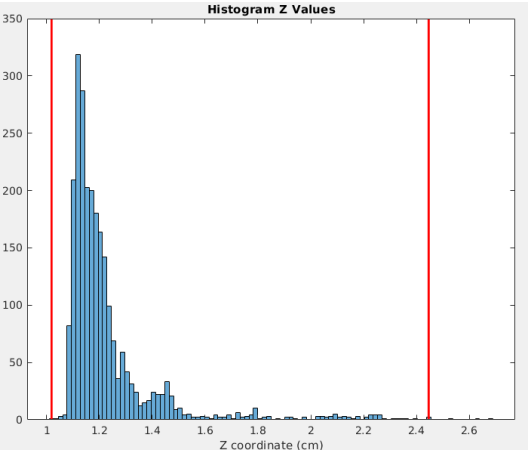
lo que hay valores que se han quedado fuera del rango.



(a) Histograma en el eje X.



(b) Histograma en el eje Y.



(c) Histograma en el eje Z.

Figura 4.12: Histogramas de los puntos pertenecientes a la nube de puntos. En rojo se representan los límites establecidos con los niveles de ruido para considerar a los puntos como pertenecientes al hígado, y por tanto dentro del dominio de la caja.

Con los valores máximos y mínimos obtenidos por medio de los histogramas se crea la caja que rodea a la nube de puntos. En la Figura 4.13 se representa la caja que rodea a la malla, conforme a los valores obtenidos en 4.16 (izquierda), y la caja que rodea a la nube de puntos del hígado (derecha).

Tras obtener las cajas que rodean a la nube de puntos y a la malla del objeto, es necesario escalar la caja que rodea a la malla para posteriormente alinearla con la nube de puntos. Para obtener un valor de escala que aplicar sobre la caja del modelo y sobre la malla, se calcula una relación de escala para cada eje:

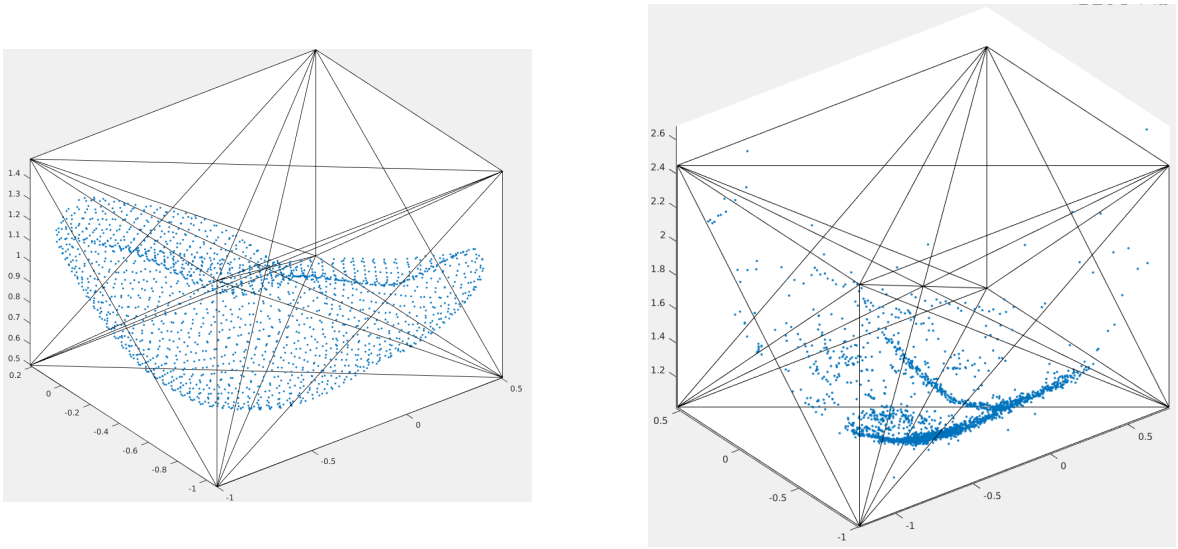


Figura 4.13: Representación de las cajas que rodean a la malla del hígado (izquierda) y a la nube de puntos (derecha), mediante los valores máximos y mínimos en los ejes X, Y y Z.

$$
\begin{aligned}
sc_x &= \frac{x_{max,c} - x_{min,c}}{x_{max,m} - x_{min,m}} \\
sc_y &= \frac{y_{max,c} - y_{min,c}}{y_{max,m} - y_{min,m}} \\
sc_z &= \frac{z_{max,c} - z_{min,c}}{z_{max,m} - z_{min,m}}
\end{aligned}
\tag{4.18}
$$

donde sc_x, sc_y y sc_z corresponden a la relación de escala de las cajas para los ejes X, Y y Z, respectivamente. $x_{min,c}$ y $x_{max,c}$ corresponden a los valores mínimos y máximos de la caja que rodea a la nube de puntos en X , $y_{min,c}$ y $y_{max,c}$ corresponden a los valores mínimos y máximos de la caja que rodea a la nube de puntos en Y y $z_{min,c}$ y $z_{max,c}$ corresponden a los valores mínimos y máximos de la caja que rodea a la nube de puntos en Z.

Al obtener la caja que rodea a la nube de puntos mediante el filtrado de ruido con los histogramas, cabe la posibilidad de que las relaciones entre las dimensiones de las aristas de la caja en X, Y y Z no sea la misma que las relaciones de la caja que rodea a la malla del modelo. Por ello, se ha calculado una escala aproximada k_{reg} utilizando el valor medio de las relaciones de escala máximas y mínimas obtenidas en 4.18:

$$
k_{reg} = \frac{max(sc_x, sc_y, sc_z) + min(sc_x, sc_y, sc_z)}{2} \tag{4.19}
$$

Antes de alinear una de las esquinas de las cajas, es necesario escalar la caja que rodea a la malla. Para ello, se ha recalculado la distancia euclídea desde la posición

del hígado estimada en la Sección 4.4.2 hasta los vértices de la caja. Partiendo de la matriz de *pose* $^{\mathbf{W}}\mathbf{T_O}$ calculada anteriormente

$$^{\mathbf{W}}\mathbf{T_O} = \begin{pmatrix} \mathbf{R_{w,3x3}} & \mathbf{p_{w,3x1}} \\ \mathbf{0_{1x3}} & 1 \end{pmatrix} \tag{4.20}$$

donde $\mathbf{p_w}$ corresponde a la traslación desde el sistema de referencia mundo hasta el sistema de referencia objeto. ORB-SLAM establece el sistema de referencia mundo en el origen, por lo que se puede asumir que las tres componentes del vector $\mathbf{p_w}$ son las coordenadas de posición del objeto:

$$\mathbf{p_w} = \begin{pmatrix} p_x \\ p_y \\ p_z \end{pmatrix} \tag{4.21}$$

Por tanto, para obtener las coordenadas de la caja que rodea a la malla del hígado escaladas:

$$\begin{aligned} x'_{min,m} &= p_x + (x_{min,m} - p_x) * k_{reg} \\ x'_{max,m} &= p_x + (x_{max,m} - p_x) * k_{reg} \\ y'_{min,m} &= p_y + (y_{min,m} - p_y) * k_{reg} \\ y'_{max,m} &= p_y + (y_{max,m} - p_y) * k_{reg} \\ z'_{min,m} &= p_z + (z_{min,m} - p_z) * k_{reg} \\ z'_{max,m} &= p_z + (z_{max,m} - p_z) * k_{reg} \end{aligned} \tag{4.22}$$

Tras escalar la caja que rodea a la malla, es necesario calcular la traslación necesaria para alinear las dos cajas. Se ha elegido la esquina de la caja donde los valores de X, Y y Z son mínimos como referencia:

$$\begin{aligned} d_x &= x_{min,c} - x'_{min,m} \\ d_y &= y_{min,c} - y'_{min,m} \\ d_z &= z_{min,c} - z'_{min,m} \end{aligned} \tag{4.23}$$

donde d_x, d_y y d_z representan la traslación necesaria para llevar la caja que rodea a la malla hacia la caja que rodea a la nube de puntos en X, Y y Z, respectivamente. Una vez se ha calculado la traslación necesaria, hay que actualizar la matriz de *pose* $^{\mathbf{W}}\mathbf{T_O}$, con los nuevos valores del vector de traslación:

$$\mathbf{p'_w} = \begin{pmatrix} p_x + d_x \\ p_y + d_y \\ p_z + d_z \end{pmatrix} \tag{4.24}$$

Actualizando el vector de traslación, se obtiene la nueva matriz ${}^{\mathbf{W}}\mathbf{T_O}$:

$$ {}^{\mathbf{W}}\mathbf{T_O} = \begin{pmatrix} \mathbf{R_w} & \mathbf{p'_w} \\ \mathbf{0} & 1 \end{pmatrix} \tag{4.25} $$

El último paso necesario para completar el proceso es el escalado de la nube de puntos. Para ello, únicamente es necesario concatenar la constante de escalado k_{reg} a las calculadas en los apartados anteriores

$$ \mathbf{P_{esc}} = k_{reg} * k_p * k_{sc} * \mathbf{P} \tag{4.26} $$

y relocalizar el modelo con la matriz ${}^{\mathbf{W}}\mathbf{T_O}$ actualizada

$$ \mathbf{P_{AR}} = {}^{\mathbf{W}}\mathbf{T_O} * \mathbf{P_{esc}} \tag{4.27} $$

En la Figura 4.14 se representa la nube de puntos y la malla del modelo antes (izquierda) y después (derecha) de realizar todo el proceso de registro.

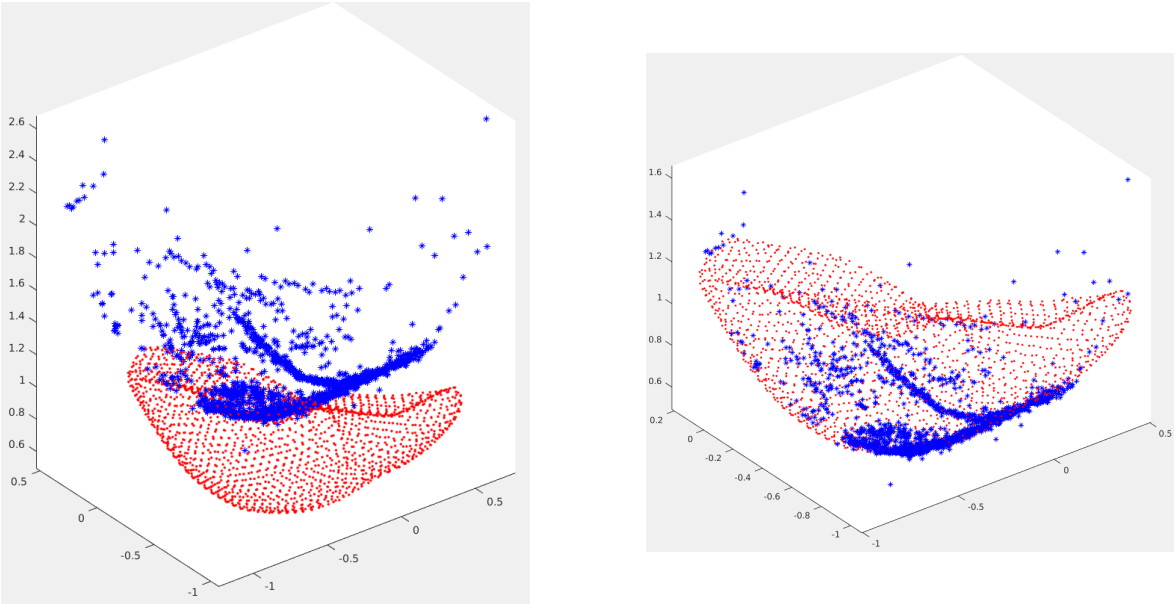


Figura 4.14: Localización espacial de la malla del hígado en rojo y de la nube de puntos del hígado en azul, antes de aplicar el registro (izquierda) y tras aplicar el registro (derecha).

Capítulo 5

Resultados

En este capítulo se exponen los resultados y representaciones en realidad aumentada sobre secuencias de los modelos del gato y del hígado. Para el modelo del gato, se ha comprobado el funcionamiento de la herramienta con una secuencia grabada con una *webcam* del modelo del gato impreso en 3D.

En el caso del modelo del hígado, se han renderizado varias secuencias de vídeo con el simulador quirúrgico, y se ha representado en realidad aumentada un tumor y la vena porta.

5.1. Modelo del gato

Aunque el modelo del gato se ha utilizado como un paso intermedio para el diseño final de la herramienta, utilizar la herramienta en una secuencia grabada con una cámara monocular estándar supone un gran avance, porque demuestra la viabilidad de la herramienta para distinguir al objeto de la escena automáticamente con una cámara estándar, como la de un endoscopio.

En esta secuencia[1], se puede observar cómo la herramienta es capaz de distinguir al gato del resto de la escena automáticamente. Con el uso conjunto de ORB-SLAM con PVNet se consigue representar el modelo del gato superpuesto a la secuencia capturada por la cámara en tiempo real. Como se ha descrito en el capítulo anterior, para la secuencia del gato se utiliza únicamente la corrección de *pose*, tomando dos fotogramas de la secuencia, y obteniendo la *pose* verdadera como un problema de optimización. En la secuencia, esto ocurre a partir del 00:23.

En la Figura 5.1 se representa el mapa de puntos creado por ORB-SLAM de la escena a la izquierda, y a la derecha el gato segmentado con más detalle. Como se puede observar, la red neuronal es bastante precisa al distinguir entre los puntos de la

[1]El texto señalado en azul representa el enlace a Youtube del vídeo que se describe. Para visualizarlo, hacer click sobre el texto.

escena y los pertenecientes al gato.

Figura 5.1: Mapa de puntos creado con la secuencia del gato. A la derecha, mapa de puntos de la escena. Los puntos rojos representan la escena, y los puntos verdes representan los puntos de la escena pertenecientes al gato. En la imagen izquierda, se pueden observar los puntos pertenecientes al gato con más detalle.

En la Figura 5.2 se representa en realidad aumentada el gato sobre la secuencia. La herramienta es capaz de obtener resultados en tiempo real, y como se puede observar, el posicionamiento del modelo sobre el objeto real es bastante preciso, teniendo en cuenta que el vídeo se ha tomado con una webcam con imagen de baja calidad y sin ningún tipo de estabilización de imagen.

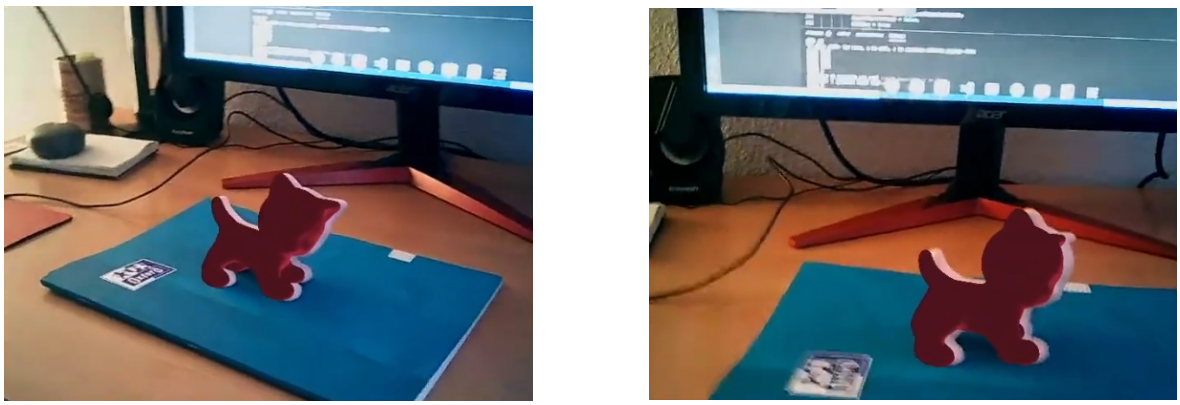


Figura 5.2: Representación en relidad aumentada del gato sobre la imagen capturada con la cámara en diferentes perspectivas.

5.2. Modelo del hígado

En el caso del hígado, se ha representado en realidad aumentada un tumor y la vena porta sobre el modelo del hígado. El tumor y la vena se han obtenido de un *dataset* público, denominado *3Dircadb* [36], y se han modificado para crear una superposición con el modelo del hígado con el que se trabaja en este proyecto. De esta forma, se recrea

la fase de obtención de imágenes preoperatorias del órgano del paciente, que se utiliza posteriormente para generar las representaciones en realidad aumentada durante la operación.

En la Figura 5.3 se representa el tumor modelado con el hígado, y en la Figura 5.4 se representa la vena porta modelada con el hígado. Estos dos modelos son la referencia de la localización del tumor y de la vena con respecto al modelo del hígado, y se comparan con los resultados obtenidos con la herramienta.

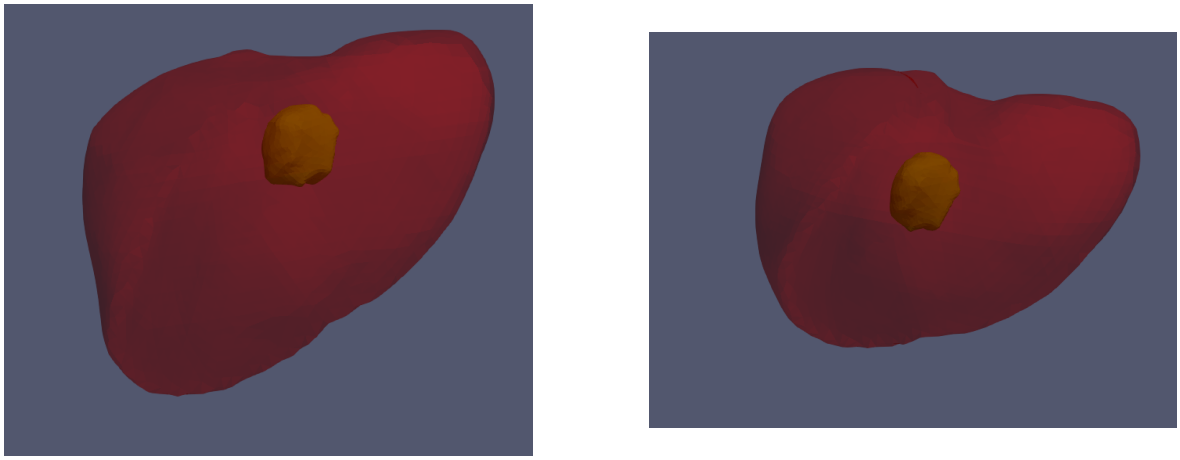


Figura 5.3: Modelo del hígado representado con el tumor, desde diferentes perspectivas. La localización del tumor en estas imágenes representa la posición verdadera, a reproducir con el uso de la herramienta. El tumor se ha obtenido de [36].

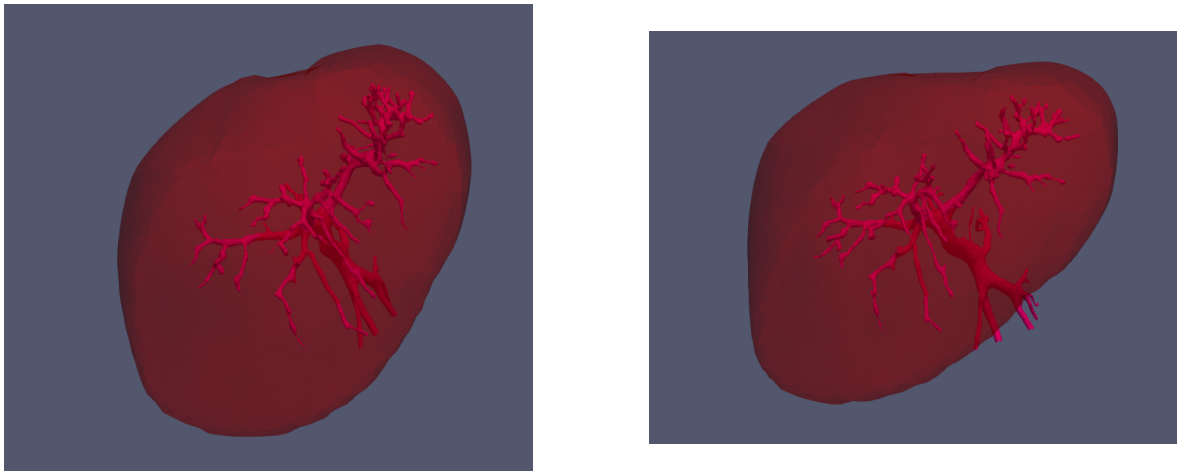


Figura 5.4: Modelo del hígado representado con la vena porta, desde diferentes perspectivas. La localización de la vena porta en estas imágenes representa la posición verdadera, a reproducir con el uso de la herramienta. La vena porta se ha obtenido de [36].

Se ha utilizado una secuencia obtenida con el simulador quirúrgico para representar la vena porta y un tumor sobre el hígado. En la Figura 5.5 se representan los puntos capturados por ORB-SLAM de la escena. En este caso, la herramienta detecta como

pertenecientes al hígado todos los puntos de la escena, ya que el fondo es homogéneo, y no es capaz de encontrar puntos consistentes al avanzar la secuencia de vídeo que no pertenezcan al objeto. Por eso, todos los puntos que detecta son verdes.

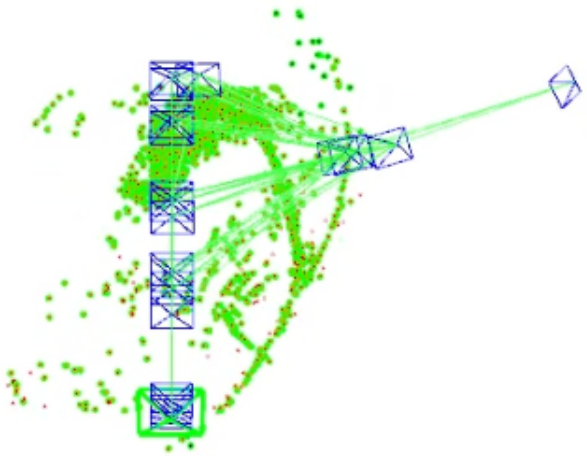


Figura 5.5: Nube de puntos de la secuencia del hígado. Todos los puntos capturados son verdes, y la herramienta los asume como pertenecientes al objeto porque el fondo de la imagen es homogéneo, y ORB-SLAM no es capaz de encontrar puntos consistentes fuera de la geometría del hígado.

En las Figuras 5.6 y 5.7 se representan el tumor y la vena porta sobre el hígado respectivamente, desde diferentes perspectivas. Comparando con las Figuras 5.3 y 5.4, se puede observar cómo la localización del tumor y la vena es bastante precisa.

El proceso llevado a cabo para la representación del tumor y de la vena porta es el mismo: en primer lugar, se fijan los dos fotogramas con los que se estima la primera aproximación de *pose* mediante el optimizador. Por último, se realiza el registro del modelo con el que se obtiene la *pose* y escala verdadera. Como se puede observar en las dos secuencias, hasta el último paso no se consigue la posición verdadera del tumor ni de la vena porta, situándose en los dos casos un poco más cercano a la cámara de lo que está en realidad. Es importante que ORB-SLAM registre una buena cantidad de puntos, que recojan la profundidad del objeto para que el registro se realice correctamente.

Figura 5.6: Representación del tumor en realidad aumentada sobre la secuencia del hígado. Como se puede observar en la secuencia y en las imágenes, la localización es consistente y coincide con la representada en 5.3.

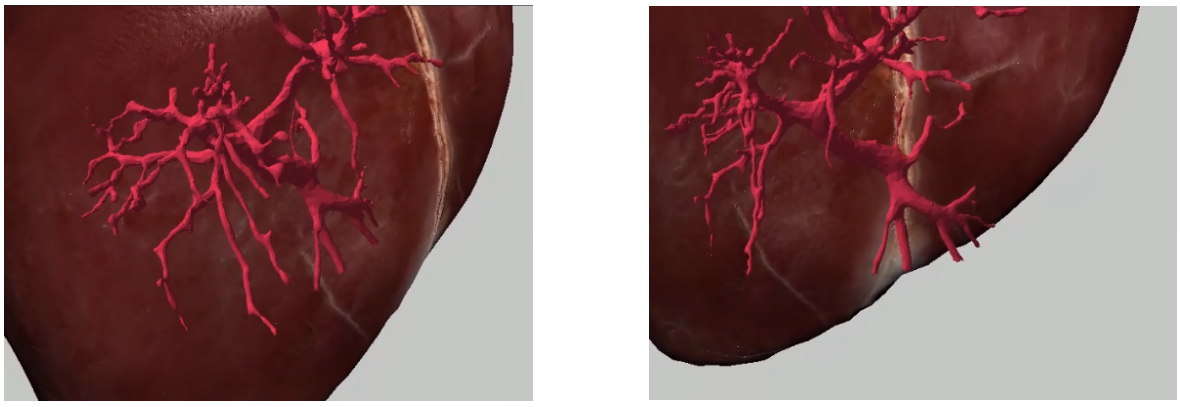


Figura 5.7: Representación de la vena porta en realidad aumentada sobre la secuencia. Comparando el resultado con la referencia en 5.4, la localización coincide con la *pose* verdadera.

Capítulo 6

Conclusiones

En este proyecto se ha desarrollado una herramienta que combina la inteligencia artificial y técnicas de visión por computador para la representación en realidad aumentada de partes internas no visibles, cuyo posicionamiento añade gran información para el cirujano, como pueden ser tumores o venas, sobre imagen laparoscópica de cirugía hepática.

Una de las aportaciones realizadas en este proyecto es la creación y segmentación en tiempo real de la nube de puntos del objeto entrenado por la red neuronal. En este caso, se ha realizado con un gato y con un hígado.

La herramienta también es capaz de obtener la *pose* verdadera del objeto automáticamente y en tiempo real, por lo que existe la posibilidad de utilizar la herramienta con vídeo grabado en vivo, como ocurre durante una intervención quirúrgica.

La combinación de la red PVNet con ORB-SLAM para la representación en realidad aumentada supone una herramienta muy innovadora y polivalente, con la capacidad de uso en diferentes campos, que no se limita únicamente a la ingeniería biomédica.

La nube de puntos creada tiene un tamaño conocido, lo que permite trabajar con la nube del objeto conociendo más datos, que permiten escalar la nube o crear modelos geométricos de dimensiones conocidas.

Como trabajo futuro, se podría reentrenar la red neuronal con hígados de diferentes geometrías, de forma que la herramienta sea más polivalente y con posibilidad de uso en un entorno más realista, más allá de escenas renderizadas.

También se podría probar la herramienta en un entorno laparoscópico completo en 3D modelado en el simulador, lo que supondría un primer paso para el uso de la herramienta en un entorno quirúrgico real.

Otra línea futura a explorar puede ser la de desarrollar un modelo deformable sobre la nube de puntos, que permita el uso de la herramienta en entornos deformables, como la deformación que produce sobre el hígado la introducción de CO_2 en el abdomen del

paciente durante una intervención laparoscópica, o las deformaciones que sufre el hígado al ser manipulado por las herramientas laparoscópicas, como pinzas o bisturís.

de partes internas no visibles, cuyo posicionamiento añade gran información para el cirujano, como pueden ser tumores o venas

Bibliografía

[1] C. Quesada, D. González, I. Alfaro, E. Cueto, A. Huerta, and F. Chinesta, "Real-time simulation techniques for augmented learning in science and engineering," *The visual computer*, vol. 32, no. 11, pp. 1465–1479, 2016.

[2] C. Quesada, I. Alfaro, D. González, F. Chinesta, and E. Cueto, "Haptic simulation of tissue tearing during surgery," *International Journal for Numerical Methods in Biomedical Engineering*, vol. 34, no. 3, p. e2926, 2018. e2926 cnm.2926.

[3] M. Pfeiffer, I. Funke, M. R. Robu, S. Bodenstedt, L. Strenger, S. Engelhardt, T. Roß, M. J. Clarkson, K. Gurusamy, B. R. Davidson, L. Maier-Hein, C. Riediger, T. Welsch, J. Weitz, and S. Speidel, "Generating Large Labeled Data Sets for Laparoscopic Image Processing Tasks Using Unpaired Image-to-Image Translation," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 11768 LNCS, pp. 119–127, 2019.

[4] R. Mur-Artal, J. M. Montiel, and J. D. Tardos, "ORB-SLAM: A Versatile and Accurate Monocular SLAM System," *IEEE Transactions on Robotics*, vol. 31, no. 5, pp. 1147–1163, 2015.

[5] R. Mur-Artal and J. D. Tardos, "ORB-SLAM2: An Open-Source SLAM System for Monocular, Stereo, and RGB-D Cameras," *IEEE Transactions on Robotics*, vol. 33, no. 5, pp. 1255–1262, 2017.

[6] T. Vercauteren, M. Unberath, N. Padoy, and N. Navab, "CAI4CAI: The Rise of Contextual Artificial Intelligence in Computer-Assisted Interventions," *Proceedings of the IEEE*, vol. 108, pp. 198–214, jan 2020.

[7] X. Chen, A. Diaz-Pinto, N. Ravikumar, and A. Frangi, "Deep learning in medical image registration," *Progress in Biomedical Engineering*, pp. 0–31, dec 2020.

[8] S. S. Mohseni Salehi, S. Khan, D. Erdogmus, and A. Gholipour, "Real-Time Deep

Pose Estimation With Geodesic Loss for Image-to-Template Rigid Registration," *IEEE Transactions on Medical Imaging*, vol. 38, pp. 470–481, feb 2019.

[9] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 37, pp. 770–778, IEEE, jun 2016.

[10] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 9351, pp. 234–241, 2015.

[11] V. Lepetit, "BB8 : A Scalable, Accurate, Robust to Partial Occlusion Method for Predicting the 3D Poses of Challenging Objects without Using Depth," *Proceedings of the IEEE International Conference on Computer Vision*, pp. 3828–3836, 2017.

[12] V. Lepetit, F. Moreno-Noguer, and P. Fua, "EPnP: An accurate O(n) solution to the PnP problem," *International Journal of Computer Vision*, vol. 81, no. 2, pp. 155–166, 2009.

[13] Y. Xiang, T. Schmidt, V. Narayanan, and D. Fox, "PoseCNN: A convolutional neural network for 6D object pose estimation in cluttered scenes," *arXiv*, 2017.

[14] S. Zakharov, I. Shugurov, and S. Ilic, "Dpod: 6d pose object detector and refiner," in *International Conference on Computer Vision (ICCV)*, October 2019.

[15] M. A. Fischler and R. C. Bolles, "Random Sample Consensus, A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography," *Communications of the ACM*, vol. 24, pp. 381–395, jun 1981.

[16] S. Peng, Y. Liu, Q. Huang, X. Zhou, and H. Bao, "Pvnet: Pixel-wise voting network for 6dof pose estimation," in *CVPR*, 2019.

[17] S. Augenstein and S. M. Rock, "Improved frame-to-frame pose tracking during vision-only SLAM/SFM with a tumbling target," *Proceedings - IEEE International Conference on Robotics and Automation*, pp. 3131–3138, 2011.

[18] O. Garcia Grasa and J. M. Martinez Montiel, "Visual SLAM for Measurement and Augmented Reality in Laparoscopic Surgery," 2014.

[19] L. M. Paz, J. D. Tardos, and J. Neira, "Divide and conquer: EKF SLAM in O(n)," *IEEE Transactions on Robotics*, vol. 24, no. 5, pp. 1107–1120, 2008.

[20] F. Vasconcelos, E. Mazomenos, J. Kelly, and D. Stoyanov, "RCM-SLAM: Visual localisation and mapping under remote centre of motion constraints," *Proceedings - IEEE International Conference on Robotics and Automation*, vol. 2019-May, pp. 9278–9284, 2019.

[21] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "Orb: An efficient alternative to sift or surf," in *2011 International conference on computer vision*, pp. 2564–2571, Ieee, 2011.

[22] A. Badias, I. Alfaro, D. Gonzalez, F. Chinesta, and E. Cueto, "MORPH-DSLAM: Model Order Reduction for PHysics-based Deformable SLAM," pp. 1–13, sep 2020.

[23] S. Hinterstoisser, S. Holzer, C. Cagniart, S. Ilic, K. Konolige, N. Navab, and V. Lepetit, "Multimodal templates for real-time detection of texture-less objects in heavily cluttered scenes," *Proceedings of the IEEE International Conference on Computer Vision*, pp. 858–865, 2011.

[24] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*. New York, NY, USA: Cambridge University Press, 2 ed., 2003.

[25] D. Shreiner, G. Sellers, J. Kessenich, and B. Licea-Kane, *OpenGL programming guide: The Official guide to learning OpenGL, version 4.3*. Addison-Wesley, 2013.

[26] CloudCompare, "Cloudcompare (version 2.9. 1) gpl software," 2018.

[27] P. Kamousi, S. Lazard, A. Maheshwari, and S. Wuhrer, "Analysis of farthest point sampling for approximating geodesics in a graph," *Computational Geometry: Theory and Applications*, vol. 57, pp. 1–7, 2016.

[28] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[29] P. Hintjens, *ZeroMQ: Messaging for Many Applications*. Oreilly and Associate Series, O'Reilly Media, Incorporated, 2013.

[30] G. Bradski, "The OpenCV Library," *Dr. Dobb's Journal of Software Tools*, 2000.

[31] J.-Y. Bouguet, "Camera Calibration Toolbox for Matlab." http://www.vision.caltech.edu/bouguetj/calib_doc/index.html, 2015.

[32] J. Heikkila and O. Silvén, "A four-step camera calibration procedure with implicit image correction," in *Proceedings of IEEE computer society conference on computer vision and pattern recognition*, pp. 1106–1112, IEEE, 1997.

[33] J. A. Nelder and R. Mead, "A simplex method for function minimization," *The computer journal*, vol. 7, no. 4, pp. 308–313, 1965.

[34] R. t. O'Neill, "Algorithm as 47: function minimization using a simplex procedure," *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, vol. 20, no. 3, pp. 338–345, 1971.

[35] R. B. Rusu and S. Cousins, "3d is here: Point cloud library (pcl)," in *2011 IEEE international conference on robotics and automation*, pp. 1–4, IEEE, 2011.

[36] R. I. against Digestive Cancer, "IRCAD Dataset." https://www.ircad.fr/research/3d-ircadb-01/.

[37] T. Fushiki, "Estimation of prediction error by using k-fold cross-validation," *Statistics and Computing*, vol. 21, no. 2, pp. 137–146, 2011.

Lista de Figuras

2.1. Representación gráfica de los diferentes parámetros que componen la matriz de la cámara en OpenGL. Figura obtenida de [25]. 6

2.2. Jerarquía de los datos, necesarios para entrenar a la red. 8

2.3. Captura del hígado obtenida desde el simulador con fondo homogéneo. 9

2.4. Máscara correspondiente a la captura del hígado descrita en la Figura 2.3. 10

2.5. Preprocesado de los datos obtenidos del simulador, para adaptar el formato al de la red neuronal. 11

2.6. Ejemplos de capturas utilizadas para hacer un primer entrenamiento de la red neuronal. 11

3.1. En azul se representan los *keypoints* a estimar por la red neuronal para una posición determinada del hígado. Estos puntos se utilizan posteriormente para calcular la *pose* del hígado mediante el algoritmo PnP [12]. 14

3.2. A la izquierda se expone una captura utilizada para el primer entrenamiento de la red neuronal. A la derecha, se expone el resultado de la estimación de la *pose* con la red neuronal. La caja azul representa la *pose* verdadera del hígado, mientras que la caja verde representa la estimación de *pose* obtenida mediante la red. 15

3.3. A la izquierda se expone una captura utilizada para el entrenamiento de la red neuronal con fondos de laparoscopia. A la derecha, se expone el resultado de la estimación de la *pose* con la red neuronal. La caja azul representa la *pose* verdadera del hígado, mientras que la caja verde representa la estimación de *pose* obtenida mediante la red. 16

3.4. A la izquierda se expone una captura utilizada para el entrenamiento de la red en condiciones realistas, con fondo de laparoscopia y el hígado cerca de la cámara. A la derecha, se expone el resultado de estimación de la *pose* con la red neuronal. La caja azul representa la *pose* verdadera del hígado, mientras que la caja verde representa la estimación de *pose* obtenida mediante la red. 16
3.5. Gráficas de entrenamiento de la red neuronal para el último conjunto de imágenes. De izquierda a derecha y de arriba a abajo: Precisión, Recuperación, Función de pérdida de Segmentación y Función de pérdida de Vértices. 18

4.1. Funcionamiento conjunto de PVNet con ORB-SLAM para un fotograma, haciendo uso de la librería ZeroMQ [29]. 22
4.2. Modelo del gato impreso en 3D, utilizado para diseñar el conjunto PVNet-ORBSLAM. 24
4.3. Máscara del objeto y segmentación de la nube de puntos. En las dos primeras imágenes se representa el fotograma, junto con la máscara predicha por la red neuronal. En la imagen inferior se representa la nube de puntos de la escena, generada por ORB-SLAM. Los puntos pertenecientes al objeto se representan en verde, mientras que los puntos de la escena se representan en rojo. 25
4.4. Sistemas de referencia mundo (W), cámara (C) y objeto (O). En azul, se representan las matrices de transformación entre los diferentes sistemas de referencia establecidos. 26
4.5. Error de estimación de la *pose* al utilizar la red neuronal en conjunto con ORB-SLAM. En la imagen izquierda se representa el fotograma en el que se ha fijado la *pose*, con el modelo 3D representado en realidad aumentada. En la imagen derecha se representa el objeto desde otra perspectiva, tras haber fijado la posición. 27
4.6. Medida tomada para establecer diferencias de escalas entre el objeto virtual (usado para entrenar la red) y el objeto impreso. A la izquierda, medida de la distancia en metros sobre la malla 3D del gato. A la derecha, medida de la distancia sobre el objeto impreso en 3D con una regla. . . 27
4.7. Sistemas de referencia establecidos para realizar la corrección de *pose* a partir de dos fotogramas: Referencia mundo (W), fotograma 1 (C_1), fotograma 2 (C_2) y objeto O. En azul se representan las matrices de transformación correspondiente entre todos los sistemas de referencia. . 29

4.8. Representación de la posición de los gatos desde dos perspectivas distintas para valores de k_p distintos. Las cámaras se representan con rectángulos verdes. En azul se representan los gatos para un valor de k_p determinado, y su proyección a partir de las matrices $^{\mathbf{C}}\mathbf{T}_{\mathbf{O}}$ desde las dos perspectivas. En rojo se representan los gatos para un valor de k_p distinto al anterior. Las líneas rojas representan la trayectoria de los gatos al cambiar el valor de k_p para cada una de las perspectivas. . . . 30

4.9. Representación del modelo del gato en realidad aumentada sobre un vídeo capturado con la webcam. Se puede observar cómo la posición del gato es consistente desde dos perspectivas distintas. 32

4.10. Representación del modelo del hígado sobre un vídeo obtenido con el simulador quirúrgico, con fondo homogéneo. En este caso, la posición del hígado es mucho menos precisa, por lo que será necesario realizar unos ajustes adicionales. 32

4.11. Nube de puntos del hígado antes (izquierda) y después de aplicar un filtrado radial (derecha). En la imagen izquierda, el punto rojo representa el centroide de la nube de puntos. 34

4.12. Histogramas de los puntos pertenecientes a la nube de puntos. En rojo se representan los límites establecidos con los niveles de ruido para considerar a los puntos como pertenecientes al hígado, y por tanto dentro del dominio de la caja. 35

4.13. Representación de las cajas que rodean a la malla del hígado (izquierda) y a la nube de puntos (derecha), mediante los valores máximos y mínimos en los ejes X, Y y Z. 36

4.14. Localización espacial de la malla del hígado en rojo y de la nube de puntos del hígado en azul, antes de aplicar el registro (izquierda) y tras aplicar el registro (derecha). 38

5.1. Mapa de puntos creado con la secuencia del gato. A la derecha, mapa de puntos de la escena. Los puntos rojos representan la escena, y los puntos verdes representan los puntos de la escena pertenecientes al gato. En la imagen izquierda, se pueden observar los puntos pertenecientes al gato con más detalle. 40

5.2. Representación en relidad aumentada del gato sobre la imagen capturada con la cámara en diferentes perspectivas. 40

5.3. Modelo del hígado representado con el tumor, desde diferentes perspectivas. La localización del tumor en estas imágenes representa la posición verdadera, a reproducir con el uso de la herramienta. El tumor se ha obtenido de [36]. 41
5.4. Modelo del hígado representado con la vena porta, desde diferentes perspectivas. La localización de la vena porta en estas imágenes representa la posición verdadera, a reproducir con el uso de la herramienta. La vena porta se ha obtenido de [36]. 41
5.5. Nube de puntos de la secuencia del hígado. Todos los puntos capturados son verdes, y la herramienta los asume como pertenecientes al objeto porque el fondo de la imagen es homogéneo, y ORB-SLAM no es capaz de encontrar puntos consistentes fuera de la geometría del hígado. . . . 42
5.6. Representación del tumor en realidad aumentada sobre la secuencia del hígado. Como se puede observar en la secuencia y en las imágenes, la localización es consistente y coincide con la representada en 5.3. 43
5.7. Representación de la vena porta en realidad aumentada sobre la secuencia. Comparando el resultado con la referencia en 5.4, la localización coincide con la *pose* verdadera. 43

A.1. Arquitectura de la red PVNet [16] y fases para la obtención de la *pose* del objeto. 60

B.1. Validación cruzada del modelo. El conjunto de datos se divide en 5 subconjuntos de igual tamaño, y se entrena 5 veces distintas de forma independiente. En verde, se representan las subdivisiones del conjunto de datos que se utilizan para el entrenamiento en cada uno de los entrenamientos independientes. En rojo, se representan las subdivisiones del conjunto utilizadas como test para cada uno de los 5 entrenamientos independientes. 64

Lista de Tablas

3.1. Métricas promedio de la última repetición del entrenamiento y del *test*. 19
3.2. Resultados de error en la traslación y la rotación de la *pose* estimada con respecto a la *pose* verdadera para las imágenes del *test*. 19

B.1. Resultados de la validación cruzada en los datos de entrenamiento para los 5 entrenamientos realizados. 64
B.2. Resultados de la validación cruzada en los datos de test para los 5 entrenamientos realizados. 65
B.3. Resultados de la validación cruzada en la estimación de la pose para los 5 entrenamientos realizados. 65