Universidad de Alcalá
Escuela Politécnica Superior

**Máster Universitario en Ingeniería Industrial**

**Trabajo Fin de Máster**

# Reconstrucción 3D de solidos deformables mediante el uso de Redes convolucionales



**Autor:** David Fuentes Jiménez

**Tutor:** Daniel Pizarro Pérez

2018

# UNIVERSIDAD DE ALCALÁ

ESCUELA POLITÉCNICA SUPERIOR

**Máster Universitario en Ingeniería Industrial**

**Trabajo Fin de Máster**

**Reconstrucción 3D de solidos deformables mediante el uso de Redes convolucionales**

Autor: David Fuentes Jiménez

Tutor: Daniel Pizarro Pérez

**Tribunal:**

**Presidente:** Luis Miguel Bergasa

**Vocal 1º:** Pedro Gil Jiménez

**Vocal 2º:** Daniel Pizarro Perez

Calificación: .....

Fecha: .....

*"Lo mismo que el hierro se oxida por falta de uso y el agua estancada se vuelve putrefacta, la inactividad destruye el intelecto"*

Leonardo Da Vinci

# Agradecimientos

*El presente es suyo, el futuro para el que trabajo, es mío...*

(Nikola Tesla-Un gran inventor que siempre fue infravalorado)

A todos aquellos que han estado conmigo durante estos 7 años de esfuerzo y sufrimiento: Estos 7 años de grado y máster han sido realmente horribles, asignaturas completamente innecesarias e inútiles, que no me han válido ni me van a valer nunca para nada, diría que solo un 40-50 por ciento del grado y el máster me han servido para algo, lo cúal es una gran pena.

Durante todo este tiempo he aprendido, he luchado y he dedicado muchas horas de trabajo a aprender, he tenido mis máximos y mis mínimos en una función a lo largo del tiempo.

Me siento afortunado de poder decir que en todos esos momentos buenos y malos, Mi familia ha estado a mi lado acompañándome. En los momentos más estresantes de la vida del estudiante, en los cúales estas frustrado y a veces lo pagas con los que menos lo merecen. Y es que a ellos les debo todo lo que soy. Gracias a mis padres Roberto y Encarnación, que me han criado con mucho esfuerzo, a mi hermano Roberto por cuidar siempre de mí y a Eva, mí novia por estar siempre ahí aguantándome y ayudándome.

En el día a día de mi trabajo como investigador debo agradecerle cada uno de los días que hemos pasado a mi compañero de trabajo David Casillas Pérez y a mi tutor Daniel Pizarro Pérez. Siempre han estado ahí y me han ayudado en todo momento para sacar todo. De David debo destacar, que la camaradería que he tenido con él y el apoyo que hemos tenido el uno con otro ha sido de los mejores que he tenido a lo largo de mi carrera. Por otro lado, de Dani debo destacar que ha sido un tutor increíble, ha tenido todo lo que esperaba como tutor y ha estado practicamente viniendo a vernos y a ver como estábamos todos los días. Siempre ha tratado de ayudar, aportar ideas, información y tratando de ofrecernos todos los medios económicos a su alcance. Muchísimas gracias a los dos.

Aunque a día de hoy ya están ausentes o trabajando, debo agradecerle a tres personas en especial todo el apoyo y ayuda que me han dado durante todos estos años. El primero de todos y más importante porque junto a David Casillas han sido como mis Obiwan para un Padawan que empezó en segundo de grado, es Jose Velasco Cerpa, que estuvo conmigo en todo momento ayudándome y aguantándome desde mis principios. Le dedico este pequeño tributo en mi carrera estudiantil. Por otro lado aunque no menos importantes, agradecer a Juan Ignacio Fraile y a Marcos Baptista Ríos el haber estado ahi para ayudarme mientras estuvieron trabajando conmigo.

Por último debo agradecer también su apoyo y ayuda a mis compañeros actuales del laboratorio, que siempre están dispuestos a echar una mano y cooperar para aprender todos juntos: Leticia, Juanma, Fran, Rober(mi aprendiz de TFG), Sara y Sergio.

# Resumen

Este trabajo de fin de master tiene por objetivo la reconstrucción 3D de sólidos deformables mediante la utilización de redes neuronales convolucionales, en este caso con una arquitectura encoder-decoder.Dicha aplicación busca una futura aplicación en realidad aumentada .

**Palabras clave:** SFT,CNN,Warp,CMI,NRSFM.

# Abstract

This master thesis aims to achieve 3D reconstruction of deformable solids through the use of convolutional neural networks, in this case with an encoder-decoder architecture. This work looks for a future application in augmented reality .

**Keywords:** SFT,CNN,Warp,CMI,NRSFM.

# Índice general

# Índice de figuras

# Índice de tablas

# Lista de acrónimos

| | |
|---|---|
| BBS | Bicubic Best-Spline. |
| CH17 | Chhatkuli17. |
| CH17R | Chhatkuli17+Refinement. |
| CNN | Convolutional Neural Network. |
| CNNP | CNN Propuesta. |
| CPU | Computational Proccesing unit. |
| FPS | Frames Per Second. |
| FT | Graphics Proccesing Unit. |
| GEINTRA | Grupo de Ingeniería Electrónica Aplicada a Espacios Inteligentes y Transporte. |
| GPU | Graphics Proccesing Unit. |
| GT | Ground-Truth. |
| GTR | Groundtruth Register. |
| HOG | Histogram of Oriented Gradients. |
| KLT | Kanade-Lucas Tracker. |
| LDOF | Large Displacement Optical Flow. |
| MDH | Maximum Depth Heuristic. |
| MSE | Mean-Squared Error. |
| | . |
| OF | Optical Flow. |
| OFR | Optical Flow Register. |
| R50F | Resnet50+Fully Connected. |
| RMSE | Root-Mean-Squared Error. |

.
.

| | |
|---|---|
| SGD | Stochastic Gradient Descent. |
| SOCP | Second Order Cone Programming. |
| | |
| TOF | Time Of Fligh. |

# Capítulo 1

# Introducción

*La vida es y siempre será una ecuación imposible de resolver, pero tiene ciertos factores que conocemos.*

Nikola Tesla

## 1.1 Introduccion

El objetivo de este Trabajo de fin de máster es la implementación y validación de un sistema para la reconstrucción y registro de objetos que sufren deformaciones a partir de la información proporcionada por una cámara de color convencional.

A día de hoy, un reto importante en la visión computacional consiste en la recuperación de la información 3D de la escena a partir de imágenes capturadas por una cámara convencional. Este problema en concreto se conoce como reconstrucción 3D. En la actualidad existen sensores de profundidad tales como Kinect [6], mediante los cuales se puede obtener directamente un mapa de profundidad de la escena. Sin embargo, el problema de la reconstrucción 3D a partir de cámaras convencionales sigue siendo un tema de gran interés. Esto es debido principalmente a las limitaciones de los sensores de profundidad, ya sea en costes, exactitud, tamaños o sus dependencias de las condiciones de luz, texturizado y movimiento en la escena.

Los objetos que se encuentran en la naturaleza se pueden dividir a grosso modo en rígidos y no rígidos. En el caso de los objetos rígidos, la reconstrucción tridimensional a partir de múltiples vistas se conoce como SfM [7] y se ha estudiado en profundidad en las últimas décadas. A día de hoy existen soluciones para SfM estables y exactas y se integra en numerosos productos y sistemas industriales. En SfM se parte de un conjunto de fotos 2D tomadas de un objeto desde diferentes vistas. Gracias a la suposición de rigidez, es posible expresar el movimiento entre imágenes como rotaciones y traslaciones de la cámara con respecto a la escena. Un ejemplo de SfM se observa en la figura 1.1

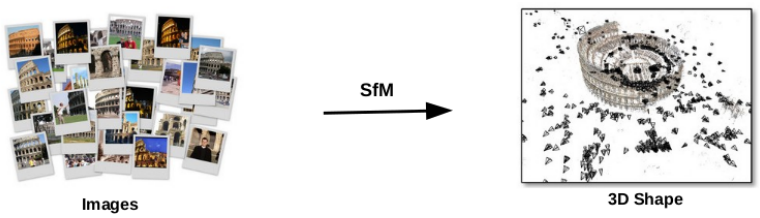



Figura 1.1: Ejemplo de reconstrucción rígida mediante SfM www.cs.cornell.edu/ snavely/bundler

En el caso de los objetos no rígidos o deformables, los métodos SfM convencionales no funcionan correctamente. Dichos objetos pueden sufrir cambios y deformaciones entre imágenes y, por lo tanto, no es posible expresar el movimiento entre imágenes exclusivamente con rotaciones y traslaciones de cámara.

En esta última década se ha estudiado ampliamente el problema de la reconstrucción deformable 3D en más de 100 artículos de investigación. Algunos de estos métodos emplean restricciones de movimiento combinadas con otras restricciones visuales para condicionar correctamente el problema de reconstrucción.

Por ejemplo, [8] combinan el movimiento con los cambios de luz, [9, 10] combinan cambios de luz, movimiento y características de bordes y [11, 12] combinan cambios de luz y texturas. Dichas soluciones no son comparables con SfM en términos de exactitud o estabilidad. La reconstrucción deformable es además un problema de considerable importancia dadas sus aplicaciones científicas y tecnológicas. Un ejemplo de reconstrucción deformable se observa en la figura 1.2.

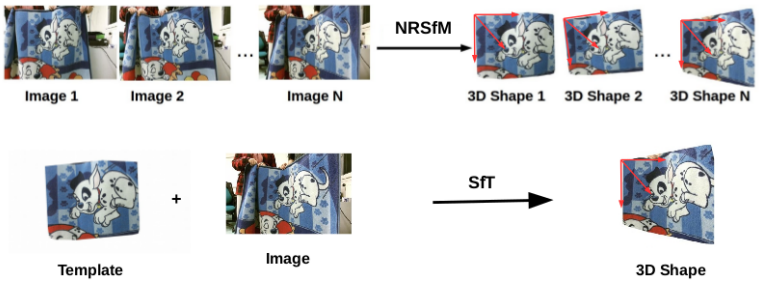



Figura 1.2: Ejemplo de reconstrucción no rígida del trabajo [1]

La reconstrucción de objetos deformables se aplica a diferentes disciplinas. A continuación se muestran algunos ejemplos:

1. **Estudios de impactos:** Dentro de este ejemplo se encuentra [13] que estudia la deformación producida por el impacto de pelotas blandas en superficies, lo cual puede ser aplicado en el mundo del deporte y la fabricación industrial. Un ejemplo de dicha aplicación se observa en la figura 1.3.

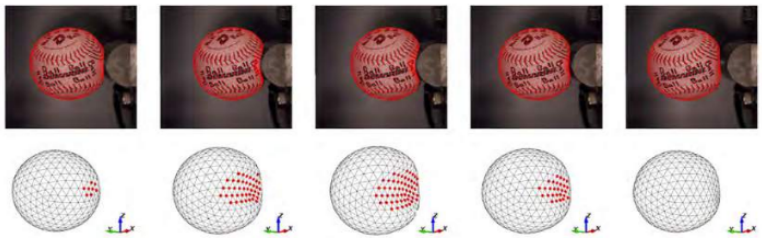


Figura 1.3: Ejemplo de estudio de impacto

2. **Recuperación de deformación objetos en tiempo real:** En este caso destaca [14], el cual recupera la deformación de objetos en tiempo real, permitiendo su uso en diferentes industrias, como la de la moda para probar prendas virtualmente o la realidad aumentada. Un ejemplo de dicha aplicación se observa en la figura 1.4.

3. **Realidad aumentada en cirugía mínimamente invasiva:** En este caso la aplicación de la reconstrucción deformable consiste en crear ayudas visuales en operaciones por laparoscopia mediante realidad aumentada. Mediante estas técnicas se combina información obtenida durante el preoperatorio del paciente (principalmente imagen 3D por resonancia magnética) con la información capturada por una cámara de laparoscopia. Dentro de este campo existen numerosas publicaciones


Figura 1.4: Ejemplo de estudio reconstrucciones

entre las que destacan [2–4, 15]. En las figuras 1.5, 1.6 y 1.7 se muestran algunos resultados de estos métodos.

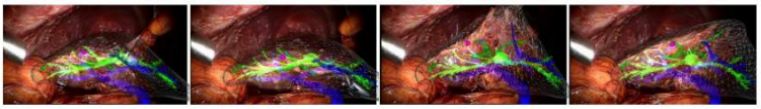


Figura 1.5: [2] Imagen laparoscópica del hígado aumentada con información 3D obtenida mediante resonancia magnética. Se muestran estructuras internas del hígado como pueden ser los tumores y vasos sanguíneos.

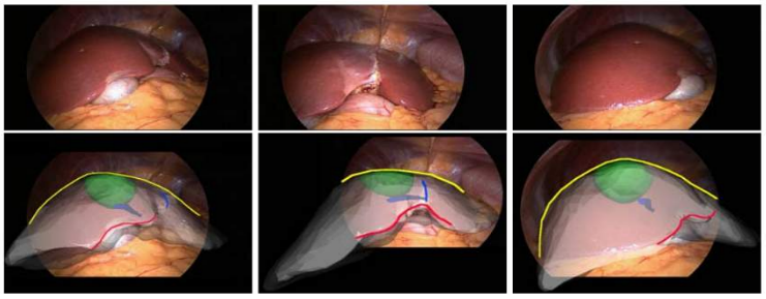


Figura 1.6: [3] Imagen laparoscópica del hígado aumentada con información 3D obtenida mediante resonancia magnética. Se muestran los ligamentos, tumores y bordes de oclusión del hígado.

Los ejemplos anteriormente mencionados constituyen aplicaciones reales y funcionales de la reconstrucción tridimensional de objetos deformables a partir de imágenes convencionales. Sin embargo, muchos de estos ejemplos requieren un alto nivel de preparación experimental e intervención humana. Las técnicas actuales de reconstrucción deformable no permiten funcionar en entornos sin restricciones, como puede ser una mesa de quirófano.

Esta Tesis de máster se centra en resolver el problema de reconstrucción deformable conocido como Shape-from-Template (SfT). Esta técnica en concreto trata de obtener la forma tridimensional de un objeto mediante el registro entre una imagen y una plantilla, que se compone de una forma de referencia del objeto y su correspondiente apariencia o mapa de textura. Un ejemplo de plantilla se contempla en la figura 1.8.

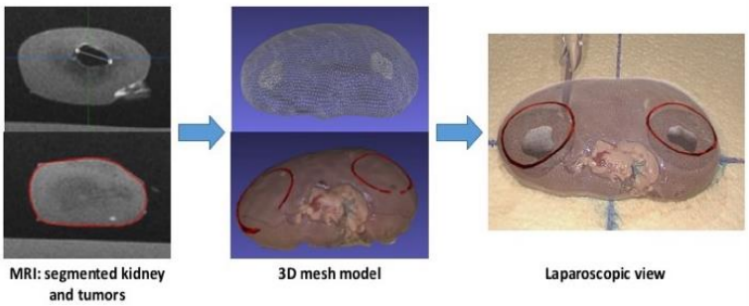



Figura 1.7: [4] Imagen laparoscópica ex-vivo de un riñón de cerdo aumentada con información 3D obtenida mediante resonancia magnética. Se muestran los tumores internos y los límites de corte de los mismos sobre la superficie del riñón.

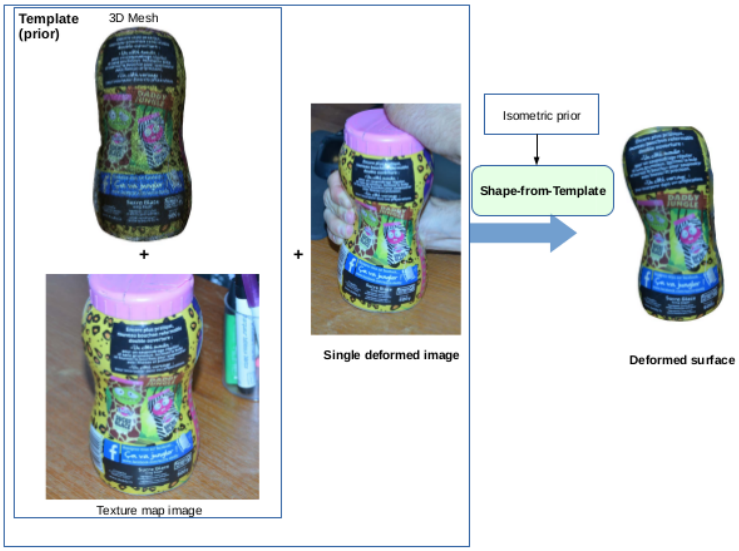



Figura 1.8: Ejemplo de SfT

A continuación se explicarán brevemente cada uno de los elementos involucrados en SfT, los cuáles se observan en la figura 1.8.

1. **Plantilla**: La plantilla esta constituida por una malla tridimensional del objeto de interés y un mapa de textura del mismo. Este mapa de textura bidimensional puede ser una imagen tomada del objeto donde se visualize su apariencia externa. En general la plantilla suele mostrar la forma del objeto en su estado natural sin deformación pero no es condición fundamental para el método propuesto. La forma o malla tridimensional de la plantilla y su apariencia están relacionados mediante una función que mapea las coordenadas de la textura sobre la malla tridimensional.

2. **Imagen con deformación**: Se trata de la imagen de entrada en la que se visualiza el objeto deformado. El objetivo de SfT es obtener la forma tridimensional del objeto deformado que se corresponde a la proyección en la imagen.

3. **Priors o restricciones**: Las restricciones constituyen un factor clave en los algoritmos de SfT. El problema SfT es de naturaleza ambigua y requiere de modelos de deformación y otras condiciones "a priori" que actúen como restricciones en el proceso de reconstrucción. En el caso de este trabajo se explorarán aquellos objetos que sufren deformaciones que se aproximan a una isometría, donde

las distancias geodésicas del objeto se mantienen invariantes a la deformación. Como se comentará en el Capítulo 2 de este trabajo, la isometría permite que el problema SfT esté bien condicionado y por tanto asegura que existe en general una única deformación posible que es compatible con su proyección en la imagen.

4. **Algoritmo SfT**: Dicho algoritmo será el encargado de relacionar las restricciones, la plantilla y la información 2D de la imagen de entrada para obtener la reconstrucción tridimensional y la relación con la plantilla.

Es importante resaltar que el algoritmo SfT requiere resolver al mismo tiempo un problema de registro deformable entre la plantilla y el objeto deformado (por ejemplo a que punto de la plantilla o su textura corresponde cada píxel de la imagen con deformación) y un problema de reconstrucción, que permite obtener la forma tridimensional del objeto deformado. En este sentido, la información obtenida en SfT de la imagen deformada no puede obtenerse con sensores de profundidad directamente ya que éstos no realizan un proceso de registro. Éste es, además, fundamental para muchas aplicaciones como la realidad aumentada.

Este trabajo explora el uso de las redes neuronales convolucionales o CNN para que, a partir de datos de entrenamiento, se relacione directamente la información contenida en una imagen de color con la deformación sufrida por la plantilla del objeto. La solución propuesta permite obtener una solución densa a la reconstrucción y registro del objeto, en tiempo real y sin necesidad de disponer de una secuencia de vídeo. En este trabajo se demuestra experimentalmente que el sistema propuesto permite resolver un problema de considerable complejidad, mejorando significativamente la calidad de la reconstrucción y el registro con respecto a los métodos del estado del arte actual.

Los sistemas basados en CNN se nutren de bases de datos de gran extensión. Sin embargo, las bases de datos de SfT son escasas y requieren de equipamiento no estandar de reconstrucción y registro, como pueden ser marcas artificiales en espectro no visible. Para abordar este problema se ha optado por la creación de una base de datos de imágenes generadas por ordenador mediante el programa de diseño gráfico Blender [16]. Este software permite simular deformaciones elásticas e inelásticas que imitan las deformaciones sufridas por objetos comunes como ropa o tejidos y componer la imagen tomada por una cámara del objeto, incluyendo efectos de sombreado de la luz, interacción con el material y los efectos de proyección. Se han generado imágenes sintéticas a partir de dos plantillas de objeto de tipo “thin-shell”, como podría ser una hoja de papel o tejido, que difieren en la riqueza de la textura que forma su apariencia en la imagen.

Se propone también el reentrenamiento o fine-tunning de la red CNN propuesta con una base de datos de menor tamaño y en la que se utiliza un sensor de profundidad para obtener datos reales de entrenamiento para la red. Este paso permite mejorar considerablemente la calidad de la reconstrucción en imágenes reales. Dentro de las alternativas actuales de sensores de profundidad destacan las cámaras de TOF como la Kinect v2 [6], StarForm 3D [17], Baxler ToF Camera[18] y las cámaras de luz estructurada como Kinect v1 [19], Astra Pro [20], Xtion Pro [21], etc.. En el caso de este trabajo, para la obtención de las medidas de profundidad, ha sido utilizada una cámara TOF Kinect v2 debido principalmente a su alta resolución de imagen de profundidad (512x424 píxeles) y su coste inferior al de otras alternativas.

El presente trabajo se enmarca dentro de las líneas de investigación del grupo de investigación GEINTRA de la Universidad de Alcalá y dentro del proyecto ARTEMISA financiado por el programa Retos de la Sociedad 2016 del MINECO.

## 1.2 Sistema propuesto

Para llevar a cabo los objetivos previamente propuestos, se ha planteado una estructura de trabajo como la que se describe en el diagrama de la figura 1.9.

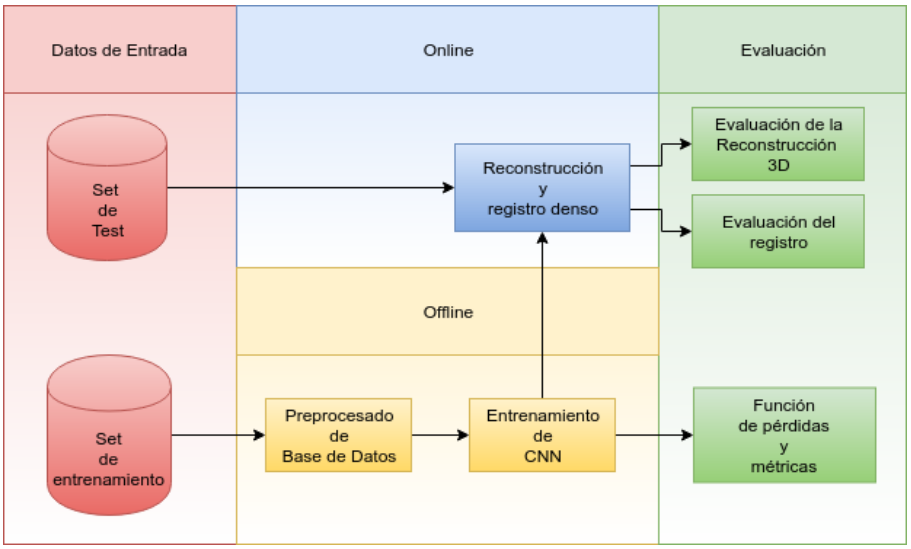



Figura 1.9: Sistema Propuesto

A continuación se comentarán y explicarán brevemente cada una de las fases reflejadas en la figura 1.9:

1. **Grabación y creación de la base de datos**: En esta etapa se realiza la creación de la base de datos, dicha base de datos se puede dividir en dos tramos bien definidos.

   (a) **Creación de una plantilla:** En este paso se definirán las plantillas utilizadas a lo largo de este trabajo y que consistirán en objetos planos definidos por una malla tridimensional y una textura asociada.

   (b) **Base de datos sintética:** Para crear una base de datos sintética de deformaciones y registro se emplea el software de simulación Blender [16] a partir de la plantilla previamente definida. Dicho uso del programa Blender requerirá la creación de una cámara virtual similar al sensor RGB incluido en Kinect v2 y mediante la cual se obtendrán las imágenes de color y los mapas de profundidad del objeto. Mediante Blender se simularán deformaciones cuasiométricas en el objeto. Por último se utilizarán funciones de interpolación basadas en funciones spline bicúbicas BBS para la generación de las funciones de registro entre la textura de la plantilla y la imagen de entrada. Dichas imágenes de registro o "warps" serán, junto con las imágenes de profundidad, los elementos necesarios para realizar el aprendizaje supervisado de la red.

   (c) **Base de datos real:** Una vez entrenada la CNN sobre la base de datos sintética, se empleará la segmentación que realiza la misma, sobre grabaciones reales de la plantilla para crear una pequeña base de datos real que permita realizar una etapa de reentrenamiento o FT sobre la misma y así adaptar la red a datos reales.

2. **Entrenamiento de la CNN**: esta fase constituye un hito crucial para el correcto funcionamiento del sistema completo. En ella se realiza el entrenamiento del sistema de reconstrucción y registro, representado por la CNN, este entrenamiento se puede dividir a su vez en dos subfases:

(a) **Entrenamiento sintético:** En esta primera fase se entrenará la CNN sobre la base de datos sintética previamente creada en Blender. Dicho entrenamiento se realizará sobre unas 23400 imágenes sintéticas que mostrarán diferentes tipos de deformaciones y movimientos, de tal manera que se favorezca la generalización de la red. Se emplearán dos salidas bien definidas, en las cuales la función de pérdidas empleada, será el MSE.

(b) **Entrenamiento real:** Una vez entrenada la CNN sobre la base de datos sintética, se procederá a realizar un reentrenamiento sobre datos reales (FT). Este reeentramiento se realizará únicamente sobre la estimación de profundidad, de tal manera que las salidas de registro se congelarán para no alterarse con respecto al entrenamiento previo.

3. **Reconstrucción y Registro**: Una vez entrenada la CNN obtenida permite realizar reconstrucción y registro de imágenes fuera del conjunto de entrenamiento.

4. **Evaluación de resultados**: esta última fase consiste en un proceso de optimización y comparación de los resultados del algoritmo con los datos etiquetados utilizando métricas que evalúen el desempeño y capacidad de generalización de la misma. Además, se obtendrán los resultados de reconstrucción obtenidos por otros algoritmos del estado del arte que realicen dichas tareas, de tal manera que se pueda comparar con sistemas actuales y medir en que cuantitativamente la mejora que ofrece el algoritmo propuesto con respecto al estado del arte actual.

En la figura 1.10 se muestra con mayor detalle la arquitectura de la CNN implementada. La entrada

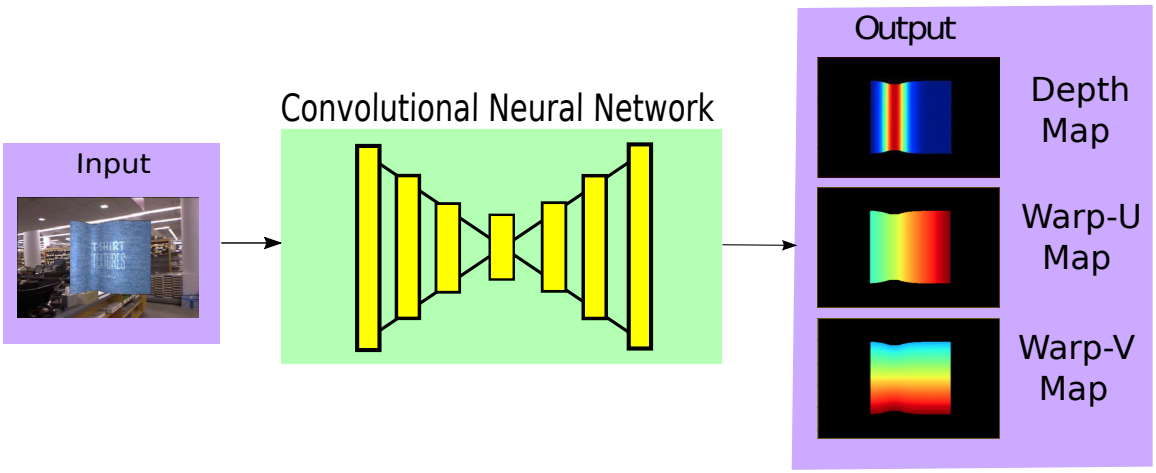



Figura 1.10: Arquitectura propuesta

a la red propuesta consiste en la imagen RGB convencional de la superficie deformada. Se plantea una arquitectura de red de tipo "fully convolutional" formada por sistemas basados en encoder-decoder. Este trabajo demuestra que este tipo de arquitectura es muy adecuada para resolver el problema SfT de manera densa. En concreto esta red basó parte de sus bloques de codificación-decodificación en los bloques convolucionales de la red del estado del arte Resnet [22]. Esta red emplea bloques residuales con normalización de batch [23], que permiten un entrenamiento más rápido y con una pérdida de información menor tras las convoluciones dada la recirculación de residuos. Las salidas de la red consisten en las siguientes imágenes:

1. **Imagen de profundidad:** La primera salida estará constituida por la estimación de una imagen de profundidad correspondiente a la superficie deformada y donde el fondo se debe suprimir, asignando un valor constante a todos los píxeles que corresponden al mismo. Esta salida por tanto representa

la reconstrucción tridimensional y además la detección y segmentación de la superficie de interés con respecto al fondo.

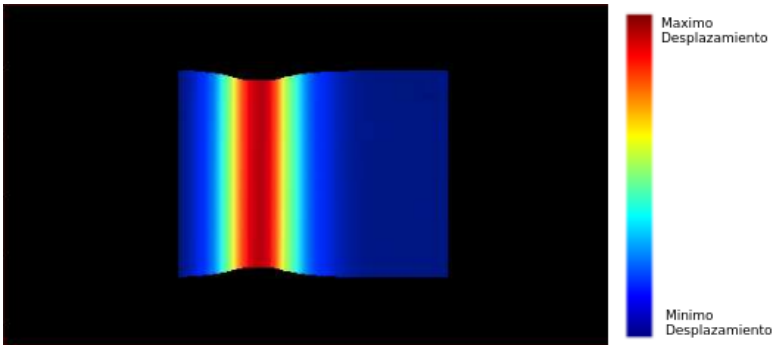



Figura 1.11: Ejemplo de mapa de profundidad de salida, expresado en milímetros

2. **Warp o función de registro de la imagen de la plantilla con la imagen de la superficie deformada:** La función de warp o función de registro dentro del contexto de SfT corresponde al mapa o función que relaciona cada píxel de la imagen de entrada y la textura de la plantilla. En la figura 1.12 se muestra un ejemplo de función de warp.

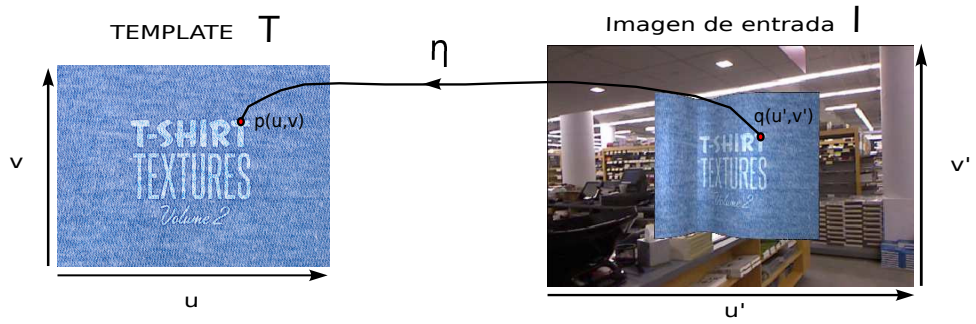



Figura 1.12: Imagen esquemática de la función de warp

En la figura representada anteriormente se observa la imagen de la plantilla plana $T$ y la imagen de la superficie deformada $I$. La función de registro o warp se representa mediante la función $\eta : \mathbb{R}^2 \rightarrow \mathbb{R}^2$. Es decir, esta función es aquella que permite trasladar todos aquellos puntos $q_i = (u'v')$ de la imagen deformada $I$ a sus correspondientes puntos $p_i = (u, v)$ en la imagen de la plantilla plana. La función de registro $\eta$ se obtiene directamente por la CNN en forma de dos funciones escalares que la componen $\eta(u', v') = [\eta_u(u', v'), \eta_v(u', v')]$ y que representan las coordenadas horizontales y verticales de la función de registro en el sistema de referencia de la plantilla plana $T$.

(a) **Warp de la coordenada u:** La segunda salida de la red consiste en una discretización por cada píxel de la función $\eta_u$, componente de la función de warp $\eta$. Es decir, el valor de cada pixel $(u', v')$ de esta salida corresponde con el valor de la coordenada $u$ correspondiente en la plantilla.

(b) **Warp de la coordenada v:** La segunda salida de la red consiste en una discretización por cada píxel de la función $\eta_v$, componente del warp $\eta$. Es decir, el valor de cada pixel $(u', v')$ de esta salida corresponde con el valor de la coordenada $v$ correspondiente en la plantilla.

.

La topología y arquitectura completa de la red propuesta se detallará en el capítulo 4.

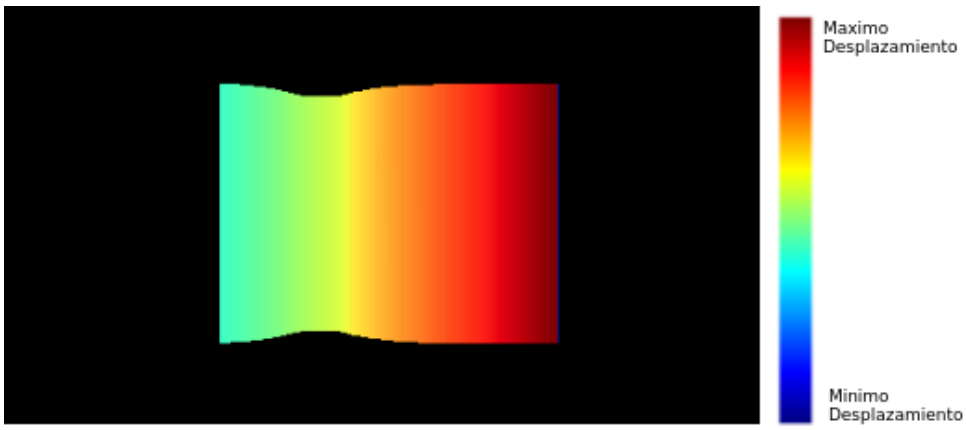



Figura 1.13: Ejemplo de subcomponente $\eta_u$ de la función de registro $\eta$. La intensidad de la imagen corresponde al desplazamiento en píxeles de la coordenada $u$

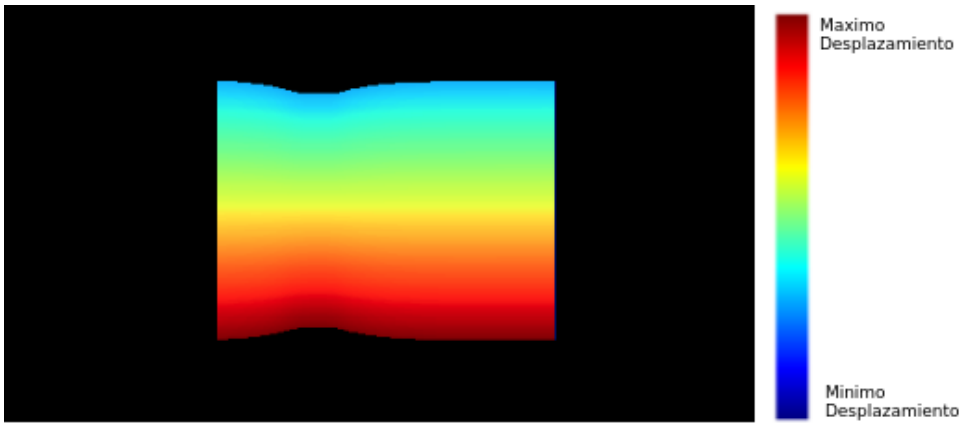



Figura 1.14: Ejemplo de subcomponente $\eta_v$ de la función de registro $\eta$. La intensidad de la imagen corresponde al desplazamiento en píxeles de la coordenada $v$

# Capítulo 2

# Estado del Arte

## 2.1 Introducción

En las últimas décadas, la reconstrucción de objetos rígidos ha sido estudiada con profundidad y se considera un problema principalmente resuelto. Las técnicas SfM permiten reconstruir una escena a partir de un conjunto de imágenes. La condición de rigidez asegura que el problema está bien condicionado y por tanto tiene una solución única salvo un factor de escala global. Este trabajo estudia la reconstrucción de objetos que se deforman y donde las soluciones de SfM no obtienen buenos resultados. La reconstrucción de objetos deformables ha sido estudiada de forma muy activa en los últimos años en más de 100 articulos científicos. Cuenta además con importantes aplicaciones en el mundo industrial y la medicina. Dentro de la reconstrucción deformable destacan dos problemas fundamentales: Non-Rigid Structure-from-Motion NrSfM [24, 25] y Shape-from-Template SfT [26–40].

## 2.2 Non-Rigid Structure From Motion

En NrSfM, se extiende SfM al campo deformable. Se trata de encontrar la forma tridimensional de un objeto deformado a partir de un conjunto de imágenes. Al contrario que en SfT, no se dispone de una plantilla tridimensional del objeto, pero sí de imágenes de la superficie deformada. Esto convierte a NrSfM en un problema considerablemente más complejo que SfT. Una descripción general de este problema se muestra en la figura 2.1.

El problema NrSfM esta mal condicionado debido a que muchas deformaciones diferentes pueden dar lugar a las mismas imágenes. Debido a estos problemas de condicionamiento, es necesario imponer restricciones que desambigüen la forma tridimensional del objeto que se busca obtener y que puede ser diferente en cada imagen de entrada.

En las últimas décadas se han propuesto gran variedad de métodos para aportar una resolución a NrSfM, empleando diferentes restriciones de deformación. Dentro de estos métodos se destacan principalmente dos categorías:

1. **Métodos basados en modelos estadísticos**: En esta primera categoría se asume que el espacio de deformaciones de interés es de baja dimensionalidad. Estos métodos corresponden con los primeros trabajos en NrSfM y siguen siendo hoy en día los más numerosos en la literatura. Los modelos estadísticos permiten recuperar gestos corporales, expresiones faciales y en general deformaciones suaves y simples. Una vez se emplean en objetos con deformaciones de alta frecuencia o atípicas

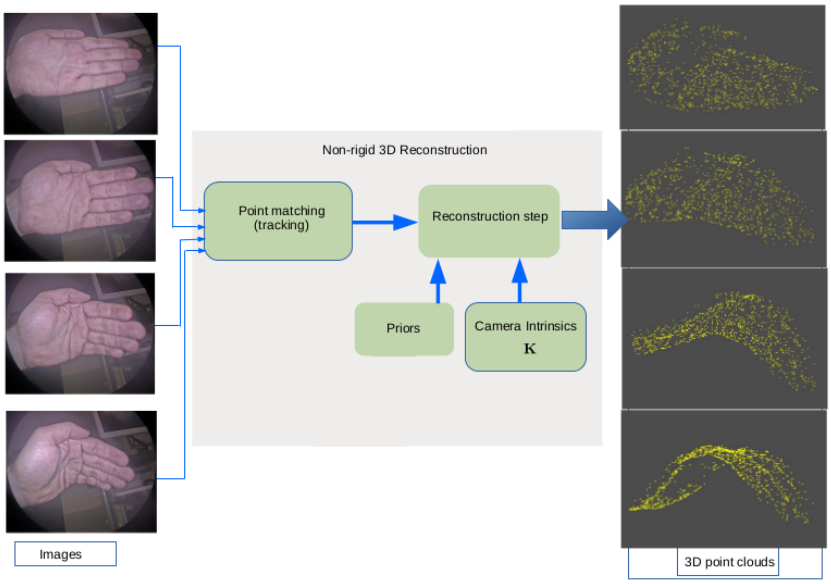



Figura 2.1: Ejemplo de NrSfM.

se obtienen reconstrucciones muy pobres. Por último, es necesario destacar que también sufren problemas frente a oclusiones. Algunos ejemplos representativos de estos trabajos se encuentran [41–43].

2. **Métodos basados en modelos físicos**: Al igual que en SfT, el modelo isométrico es especialmente interesante y preciso para una gran cantidad de objetos reales. En los últimos años se han propuesto trabajos que tratan de resolver NrSfM imponiendo isometría en las deformaciones. Estos trabajos recientes demuestran que la isometría también hace que NrSfM sea un problema bien condicionado en general. Sin embargo, son costosos computacionalmente y presentan problemas a la hora de reconstruir objetos en entornos reales sin restricciones. Dentro de esta categoría se destacan los trabajos [44–46].

## 2.3 Shape From Template

En SfT se trata de obtener la forma del objeto mediante el registro entre una imagen y una plantilla, que se compone de un mapa de textura y una forma de referencia del objeto. La figura 2.2 muestra un ejemplo de plantilla para la reconstrucción en SfT.

Al igual que en NrSfM, en SfT se deben imponer restricciones a la deformación para encontrar una solución única al problema. En la literatura se han propuesto métodos de reconstrucción basados en modelos estadísticos de deformación [41–43] y métodos basados en modelos físicos de deformación [44–46]. En este último grupo, que agrupa la mayoría del estado del arte en SfT, destacan los métodos que restringen las deformaciones con el modelo isométrico [28]. La isometría preserva las distancias geodésicas de la superficie del objeto y son eficaces para modelar las deformaciones de objetos tales como papeles, tejido o ropa.

Recientemente se ha demostrado en [28] que el modelo de deformación isométrico es suficiente para hacer de SfT un problema bien condicionado. Los métodos SfT isométricos han sido muy estudiados en los últimos años y cuentan con sistemas robustos que funcionan en tiempo real [29] en entornos controlados.

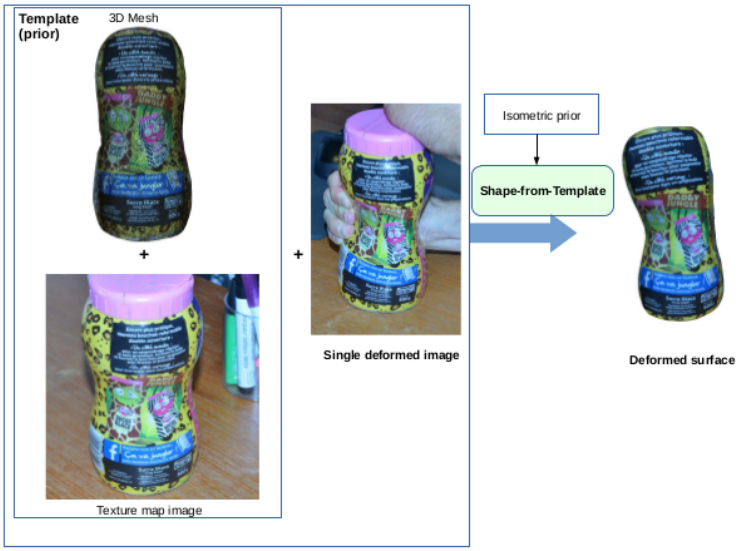



Figura 2.2: Ejemplo de SfT

La organización de los métodos que resuelven SfT isométrico se comentan a continuación y se dividen en función de las restricciones empleadas y el tipo de optimización utilizada:

1. **Métodos de orden cero basados en inextensibilidad** [47, 48].

2. **Métodos estadísticos de refinado de coste óptimo** [49, 50].

3. **Ecuaciones diferenciales parciales cuadráticas** [27, 29].

### 2.3.1 Métodos de orden cero basados en inextensibilidad

Este tipo de métodos representan la superficie como un conjunto de puntos o una malla tridimensional. Dentro de este tipo de sistemas se pueden encontrar por ejemplo [47],[48] que resuelven SfT mediante la maximización de la profundidad (Maximum Depth Heuristic) mientras imponen un límite superior a la distancia entre los puntos vecinos entre sí (modelo de inextensibilidad). La inextensibilidad se emplea como una relajación de la restricción isométrica. Estos métodos, en general, se basan en resolver un problema de optimización convexa [51], reformulando el problema MDH como un programa convexo de tipo "Second-Order Cone Programming"(SOCP) para calcular la profundidad de los puntos de la malla.

Cada vértice de la malla $Q_i$ se obtiene a partir de la siguiente expresión:

$$Q_i = z_i \begin{pmatrix} q_i \\ 1 \end{pmatrix}, \tag{2.1}$$

donde la $z_i$ es la profundidad del punto indexado por $i$ y $q_i$ es la correspondencia normalizada del punto en la imagen. El problema de reconstrucción se resume en el siguiente problema de optimización:

$$\begin{aligned} &\underset{(z_i)}{maximize} \sum_{i=1}^{n} z_i, \\ &subjectto, \forall_i \in (1...n), j \in \aleph(i) \\ &z_i \geq 0 \\ &\| z_i \begin{pmatrix} q_i \\ 1 \end{pmatrix} - z_j \begin{pmatrix} q_j \\ 1 \end{pmatrix} \|_2 \leq d_{ij} \end{aligned} \tag{2.2}$$

En este caso $\aleph(i)$ se refiere al conjunto de índices j para los cuales $Q_j$ esta en el vecindario de un punto $Q_i$. En este caso se maximiza la suma de todas las profundidades y se restringen las distancias entre vértices vecinos de la malla con la condición de inextensibilidad. En la figura 2.3 muestra un esquema de funcionamiento del algoritmo.

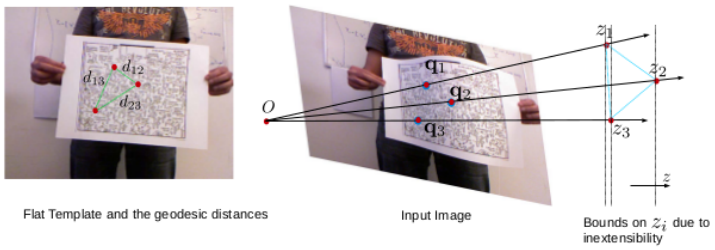



Figura 2.3: Ejemplo de funcionamiento del algoritmo

El principal problema de este tipo de métodos es que imponen isometría de forma aproximada y pueden producir reconstrucciones erróneas. Esto es especialmente problemático cuando la malla tridimensional tiene un número pequeño de vértices. Además, estos métodos son complejos computacionalmente y especialmente sensibles a las condiciones de proyección, siendo muy inestables cuando la imagen muestra poca perspectiva.

### 2.3.2 Métodos estadísticos de refinado de coste óptimo

En el caso de los métodos estadísticos [49], se optimiza una función de coste de máxima verosimilitud que incluye:

- **La restricción de la ecuación diferencial isométrica**
- **La restricción de la reproyección**
- **La restricción de suavidad**

La función de coste a minimizar se compone de los siguientes términos aditivos:

$$\underset{\phi P}{minimize}\, E_{data} + l_{iso} E_{iso} + l_{iso} E_{smooth}, \tag{2.3}$$

La función de reconstrucción se define como $\varphi(p) \in C^2(R^2, R^3)$. Dicha función se parametriza mediante una función BBS y donde el dominio de la misma corresponde a la textura objeto en la plantilla. $E_{Data}$

representa el error de reproyección y generalmente se basa en la extracción de características entre la textura de la plantilla y la imagen de entrada . $E_{Iso}$ es mínimo cuando la función $\varphi$ cumple las restricciones diferenciales de la isometría. $E_{Smooth}$ impone que la superficie sea suave, forzando a que las segundas derivadas de $\varphi(p)$ sean pequeñas. Este funcional se basa en la conocida como "Bending Energy.º energía de flexión de la BBS que representa $\varphi$. Estos términos de error se ponderan mediante el uso de los hiperparámetros $l_{iso}$ y $l_{smooth}$.

Por un lado, optimizar la ecuación 2.3 es un problema de optimización no convexa y requiere de métodos iterativos de optimización. Generalmente se utilizan algoritmos derivados de Gauss-Newton como el algortimo de descenso de Levenberg-Marquardt. Este tipo de refinado requiere de un buen punto de inicialización y un gran numero de iteraciones para converger al mínimo de la función deseado. Además de eso, la optimización requiere una ponderación de las tres restricciones con los hiperparámetros antes citados. Esta ponderación deberá ser optimizada mediante experimentación para obtener buenos resultados. El método de [50] sigue el mismo principio que [49] pero está diseñado para funcionar en tiempo real y resuelve simultáneamente el problema del registro entre la plantilla y la imagen de entrada. La única restricción es que se tome una secuencia de vídeo, donde la diferencia entre imágenes consecutivas sea pequeña, lo que permite mantener el registro. Existen otros métodos en la literatura como [52–54] que operan de forma parecida a [50] y que incluyen mejoras en la función de optimización o el registro. Estos métodos en general requieren que el objeto tenga una textura suficientemente rica como para asegurar el registro. Además, están sujetos a la pérdida de tracking debido a oclusiones parciales o totales del objeto.

### 2.3.3 Ecuaciones diferenciales parciales cuadráticas

Los métodos en esta categoría se conocen como métodos analíticos de reconstrucción y cuyo principal exponente lo componen los métodos propuestos por [27, 55]. Estos métodos asumen que la función de warp entre la plantilla y la imagen de entrada es conocida y es una función diferenciable. A partir de dicha función, y las condiciones diferenciales de isometría, expresan SfT como un sistema no lineal de ecuaciones diferenciales parciales en función de la profundidad de la superficie y sus derivadas. Dicho sistema es redundante y permite calcular la profundidad de forma analítica por cada punto de la superficie. Esta solución se conoce como la solución no holonómica del sistema de ecuaciones diferenciales parciales. Estos métodos requieren por tanto conocer la función de warp.

La naturaleza local de estas soluciones implica que estos métodos son muy rápidos y pueden ser paralelizados eficientemente, dado que la solución de cada punto puede ser encontrada independientemente. Además, estos métodos son una potente herramienta para analizar las propiedades del problema SfT y demostrar que está bien condicionado mediante isometría. Estos trabajos son actualmente los algoritmos más precisos para resolver SfT.

En este trabajo se utilizarán los algoritmos propuestos en [55] para comparar la precisión en la reconstrucción de superficies del método propuesto. Por último se mostrará una imagen ejemplificativa de reconstrucciones llevadas a cabo por este algoritmo, la misma se podrá observar en la figura 2.4.

### 2.3.4 Métodos basados en redes convolucionales

A día de hoy existen muy pocos métodos que utilicen CNNs para resolver la reconstrucción deformable. El más relevante [56] emplea una CNN que se divide en 3 grandes bloques, mediante los cuales estiman un registro en dos dimensiones y una predicción de la profundidad . Dichos bloques son:

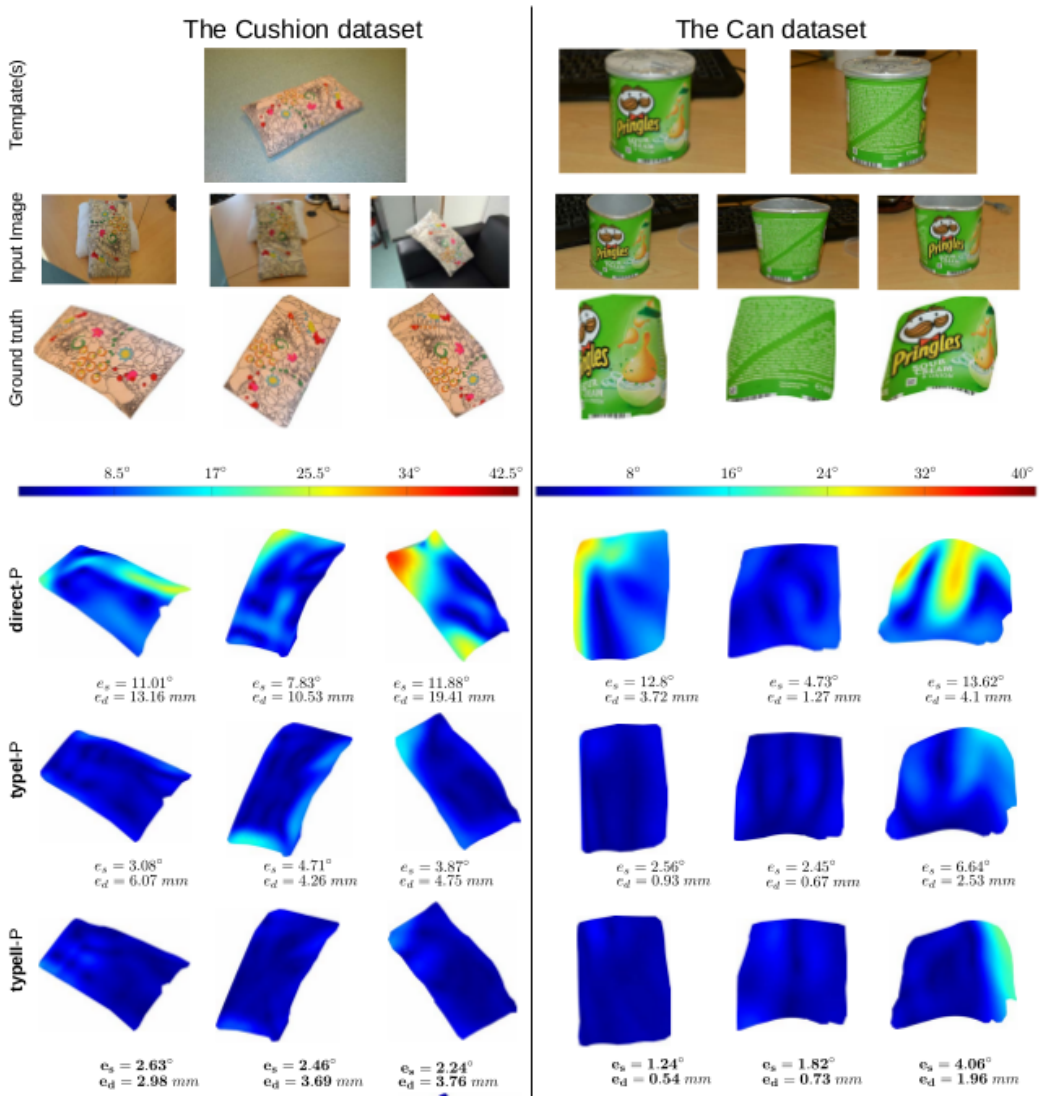



Figura 2.4: Ejemplo de funcionamiento

- **2D Detection Branch:** Este bloque es el encargado de estimar cada una de las posiciones en dos dimensiones de los vertices de la malla del objeto cuya deformacion se quiere estimar. En este caso sus bloques estan basados en las Pose machines [57].

- **Depth Branch:** Por otro lado este bloque se alimentará con la información de entrada y ademas empleará la estimación 2D de cada uno de los vertices de la malla para obtener una regresión de la profundidad de la malla que se emplea.

- **Shape Branch:** Por último se emplea el bloque de shape que es el encargado de emplear la información del mapa de profundidad estimado y del registro de los puntos de la malla para hacer una regresión 3D de los vertices de la malla en el espacio.

Por contra este método emplea una arquitectura muy compleja que no funciona en tiempo real. Además de ello, esta limitado por la estimación y número de vertices de la malla, mostrando resultados para objetos con un máximo de 15x15 vértices.

## 2.4 Métodos de registro deformable

La mayor parte de los algoritmos que solucionan SfT que se han comentado anteriormente asumen que la función de registro entre la plantilla y la imagen es conocida. Existen diversos métodos para realizar el registro deformable. Se pueden dividir en:

- **Métodos basados en características:** Dentro de los mismos se encuentran por ejemplo [58] y [59]. En los primeros se extraen características invariantes entre la plantilla y la imagen, como los populares SIFT [60], SURF [61], KAZE [62], ORB [63]...etc. Dichas características son filtradas de correspondencias erróneas y utilizadas para calcular una función de warp. Estos métodos tienen problemas para capturar el detalle de deformaciones de alta frecuencia pero permiten trabajar en condiciones "wide-baseline".
- **Métodos densos:** Dentro de los mismos se encuentran por ejemplo [64] que se basa en flujo óptico. En este grupo, su uso para SfT requiere contar con una secuencia de vídeo que parta de la plantilla y termine en la imagen de la deformación. Estos métodos son densos ya que permiten capturar miles de correspondencias entre imágenes de la secuencia y generar "tracks" de puntos entre la plantilla y la imagen. Estos métodos se conocen como métodos de flujo óptico denso.

En esta Tesis de máster se utiliza un algoritmo de registro denso [64] basado en flujo óptico como base para obtener el registro deformable necesario en los algoritmos del estado del arte. Sin embargo, hay que destacar que nuestro sistema es capaz de trabajar en condiciones de "wide-baseline" ya que se aplica a imágenes independientes y obtiene tanto el registro deformable como la reconstrucción de manera densa.

### 2.4.1 Flujo óptico denso de gran desplazamiento

La técnica LDOF de resolución del flujo óptico, es una técnica variacional que integra la asociación discreta de regiones alrededor de puntos entre imágenes dentro de una formulación basada en minimización de una función de energía.

Dicha asociación de regiones puede ser sustituida por otros descriptores tales como HOG. Estas características más sencillas de encontrar permiten implementar el sistema de resolución variacional y el sistema de correspondencias sobre una GPU de forma eficiente. El funcional de energía que se pretende minimizar consiste en la siguiente función de energía:

$$\begin{aligned} E(\mathbf{w}) &= \int_{\Omega} \Psi_1\left(|I_2(\mathbf{x}+\mathbf{w}(\mathbf{x})) - I_1(\mathbf{x})|^2\right) \\ &+\gamma \int_{\Omega} \Psi_2\left(|\nabla I_2(\mathbf{x}+\mathbf{w}(\mathbf{x})) - \nabla I_1(\mathbf{x})|^2\right) d\mathbf{x} \\ &+\beta \int_{\Omega} \delta(\mathbf{x})\ \rho(\mathbf{x})\ \Psi_3(|\mathbf{w}(\mathbf{x}) - \mathbf{w_1}(\mathbf{x})|^2) d\mathbf{x} \\ &+ \int_{\Omega} \delta(\mathbf{x})\ |\mathbf{f_2}(\mathbf{x}+\mathbf{w_1}(\mathbf{x})) - \mathbf{f_1}(\mathbf{x})|^2 d\mathbf{x} \\ &+\alpha \int_{\Omega} \Psi_S\left(|\nabla u(\mathbf{x})|^2 + |\nabla v(\mathbf{x})|^2\right) d\mathbf{x} \end{aligned} \tag{2.4}$$

donde $w = (u \quad v)^T$ y $\psi_*(s)^2$ son funciones de penalización generales cuya derivada deben cumplir $\psi_*^{'}(s)^2 > 0$. En la resolución del flujo óptico, los dos puntos clave son la precisión y la velocidad de la resolución. Dado que uno de los puntos claves es la velocidad, se puede sacar gran partido de los potentes procesados que existen hoy en día, en particular de la paralelización de los mismos en sistemas

GPU. Dicha paralelización no es sencilla dado que requiere el empleo de algoritmos capaces de funcionar simultáneamente y que son de naturaleza secuencial.

En este caso la paralelización de la búsqueda y asociación de características se puede implementar fácilmente, dado que los puntos característicos pueden ser buscados en paralelo sin dependencia entre ellos. La tarea que parece más complicada en dicha paralelización es la de la implementación de un sistema de resolución variacional en paralelo, que es lo que se tratará a continuación.

La resolución variacional se realizará minimizando el funcional 2.4 y escribiendo las ecuaciones de Euler-Lagrange para poder resolverlas de forma iterativa. De ello resultan una secuencia de resoluciones de sistemas lineales donde cada píxel corresponde a dos ecuaciones acopladas en el sistema lineal.

En CPU se suelen resolver los sistemas mediante la relajación de Gauss-Seidel que garantiza convergencia, pero esta técnica es secuencial y no puede ser paralelizada facilmente GPU.

Por ello se recurre al método del gradiente conjugado, que requiere como condición matrices simétricas definidas positivas para poder funcionar correctamente. La convergencia de dicho método depende fuertemente del numero de condición de la matriz $k = \frac{\lambda_{max}}{\lambda_{min}}$. En este caso los números de condición obtenidos de las matrices de flujo óptico son grandes y complican la convergencia, por lo cual para mejorar esta situación se trata de precondicionar dichas matrices para reducir el numero de condición de la matriz.

#### 2.4.1.1 Seguimiento de Puntos con el Flujo óptico de gran desplazamiento

El sistema de flujo óptico variacional se puede utilizar para obtener trayectorias de puntos a lo largo de la secuencia de imágenes. Lo primero que es necesario comentar de este método, con respecto a los clásicos métodos de KLT, es que el flujo óptico variacional tiene restricciones de suavidad globales. Esto permite el seguimiento de un numero de puntos mucho mayor y no se restringe solo a unos pocos. Además, permite el seguimiento de objetos rápidos con respecto a los métodos convencionales.

Inicialmente un conjunto de puntos es inicializado en el primer frame con todos los píxeles disponibles, eliminando aquello puntos que no muestren ninguna estructura o textura en su vecindad. Dependiendo de la aplicación de interés pueden buscarse más o menos trayectorias seguidas, de tal manera que este algoritmo incorpora la posibilidad de un "subsampling" espacial que reparta uniformemente los puntos en una rejilla a lo largo de la imagen.

Cada uno de los puntos puede ser seguido durante el siguiente frame gracias al campo de flujo óptico $w = (u \quad w)^T$:

$$(x_{t+1}, y_{t+1})^T = (x_t, y_t)^T + (u_t(x_t, y_t), v_t(x_t, y_t))^T \tag{2.5}$$

Dado que el flujo óptico tiene precisión subpixélica, sus coordenadas x e y posiblemente tendrán valores entre los puntos de la rejilla. Para poder inferir el flujo en estos puntos se emplea la interpolación bilineal.

El seguimiento finalizará tan pronto como el punto seguido se ocluya. Dicha oclusión se detecta mediante un análisis de la consistencia del flujo óptico entre frames consecutivos. Dado que siempre existen pequeños errores de estimación en el flujo óptico, se permite un intervalo de tolerancia que permite el incremento lineal de los mismos con respecto a la cantidad de movimiento, tal y como se define en la siguiente expresión:

$$|w + \hat{w}|^2 < 0{,}01(|w|^2 + |\hat{w}|^2) + 0{,}5 \tag{2.6}$$

Otra condición de parada del seguimiento de los puntos serán unos límites de movimiento. Dado que la localización exacta del flujo óptico fluctúa un poco, esto puede producir el mismo efecto que el de

las oclusiones y causar un deslizamiento en los puntos seguidos. Para evitar este efecto se parará el seguimiento acorde a la condicion siguiente:

$$|\nabla_u|^2 + |\nabla_v|^2 > 0{,}01|w|^2 + 0{,}002 \tag{2.7}$$

Por último, para llenar las áreas que queden vacías a causa de la pérdida de esos puntos, en cada frame se inicializan nuevas trayectorias en las áreas vacías. Finalmente se observa un ejemplo del funcionamiento del algoritmo, en el cual se ha realizado un subsampling de 8 píxeles, creando una rejilla homogénea 2.5.

Figura 2.5: Ejemplo de funcionamiento con subsampling de 8 pixeles

# Capítulo 3

# Conocimientos Previos

*Sin sacrificio no hay victoria.*

Eva Avilés Cerviño

## 3.1 Introducción

En este apartado del trabajo de fin de máster se explican en profundidad los fundamentos teóricos necesarios para su elaboración. Así, en los siguientes apartados se detallan el modelo de cámara en perspectiva, a continuación las deformaciones isométricas y por último fundamentos y artículos más relevantes de las redes Convolucionales profundas 3.4

## 3.2 El modelo de cámara de perspectiva

A día de hoy los sistemas de lentes de las cámaras son sumamente complejos, pero en la gran mayoría de los casos se pueden modelar mediante el llamado modelo de cámara "pinhole" o cámara de perspectiva. En este modelo se supone que la cámara tiene por lente una apertura infinitesimal por la que entra la luz, tal y como se observa en la figura 3.1. El modelo de cámara de perspectiva es el modelo de cámara más usado, debido a que describe con gran exactitud las proyecciones de la imagen a partir de pocos parámetros.

Considerando que un punto 3D $Q = [Q_x \quad Q_y \quad Q_z]^T \in \mathbb{R}^3$ es proyectado en el plano imagen. Si se asume que la distancia entre el centro de la cámara y el sensor (distancia focal) es la unidad y que el eje de coordenadas de la imagen esta centrado en el centro físico de la pantalla, el punto proyectado vendrá dado por $q = [q_u \quad q_v]^T = \frac{1}{Q_z}[Q_x \quad Q_y]^T$.

El modelo "pin-hole" simplificado con distancia focal unitaria se completa con los siguientes parámetros de cámara:

1. **Parámetros intrínsecos:** La proyección de los puntos tridimensionales puede diferir dependiendo del posicionado relativo del plano imagen, centro de la cámara y de la forma del plano imagen. Dicho posicionamiento relativo se define mediante cinco parámetros intrínsecos de la cámara: distancia focal en la dirección del eje horizontal $f_x$ en píxeles, distancia focal en la dirección del eje vertical $f_y$ en píxeles, punto principal $p_c = [c_x \quad c_y]^T$ de la imagen y el parámetro llamado "skew"$s_k$, que es importante si los ejes sobre el plano imagen no son exactamente perpendiculares. Estos son los

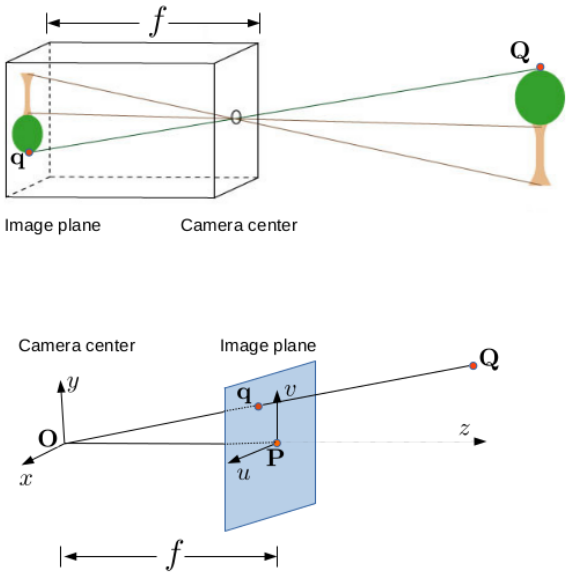



Figura 3.1: Ejemplo de modelo pinhole y su modelado matemático

parámetros que se buscan cuando se realiza una calibración de una cámara convencional. Dichos parámetros se componen en una matriz de 3x3 tal como se observa en la ecuación 3.1. Es necesario comentar que existen otros parámetros importantes no reflejados en la matriz K que tratan de modelar la distorsión de la lente y resultan de gran importancia a la hora de utilizar lentes con gran ángulo de apertura.

$$K = \begin{bmatrix} f_x & s_k & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \tag{3.1}$$

2. **Parámetros extrínsecos:** Para poder definir matemáticamente la proyección 3D de un punto en el plano imagen es necesario que las coordenadas tridimensionales estén en el mismo sistema de coordenadas que el de la cámara. El sistema de coordenadas del objeto se llama el sistema de coordenadas del mundo mientras que por otro lado se tiene el sistema de coordenadas de la cámara. Cuando dichos sistemas no están alineados es necesario alinearlos mediante rotaciones y traslaciones del punto previas a la proyección. Dichos parámetros de rotación y traslación son los parámetros extrínsecos y son los parámetros necesarios para alinear los sistemas de coordenadas. Estos se componen de 6 parámetros que componen una transformación rígida.

Matemáticamente una cámara de perspectiva se define por una matriz de 3x4 que actúa sobre un punto tridimensional. La cámara proyecta cualquier punto 3D $Q \in \mathbb{R}^3$ al plano imagen $q \in \mathbb{R}^2$.

$$s \begin{bmatrix} q \\ 1 \end{bmatrix} = M \begin{bmatrix} Q \\ 1 \end{bmatrix} \tag{3.2}$$

La ecuación 3.2 describe la proyección de perspectiva en coordenadas homogeneas para el punto 3D e imagen. El escalar $s \in R(0)$ se introduce debido a que un punto en un sistema de coordenadas homogeneo permanece igual despues de multiplicarlo por un escalar no nulo. La matriz de proyección codifica los parámetros intrínsecos y extrínsecos de la cámara como:

$$M = K[R \quad t] \tag{3.3}$$

Cuando se consideran superficies no rígidas, a veces, es útil expresar los puntos 3D en el sistema de coordenadas de la cámara $R = I_3$ y t=0. En ese caso la proyección en perspectiva será:

$$s \begin{bmatrix} q \\ 1 \end{bmatrix} = K \begin{bmatrix} I_3 & 0 \end{bmatrix} \begin{bmatrix} Q \\ 1 \end{bmatrix} \tag{3.4}$$

Si se premultiplica por $K^{-1}$, se obtiene:

$$s \begin{bmatrix} q_n \\ 1 \end{bmatrix} = \begin{bmatrix} I_3 & 0 \end{bmatrix} \begin{bmatrix} Q \\ 1 \end{bmatrix} \tag{3.5}$$

La ecuación anterior describe la relación entre las coordenadas normalizadas de los puntos de imagen $q_n$ o puntos en coordenadas de retina. En estas coordenadas, la distancia focal es 1 y el centro físico de la imagen $p = [0 \quad 0 \quad 1]^\top$. La ecuación 3.5 se puede reescribir de manera que se observe mejor la parametrización de un punto 3D tal como se observa a continuación:

$$\begin{bmatrix} Q_x \\ Q_y \\ Q_z \end{bmatrix} = Q_z \begin{bmatrix} q_n \\ 1 \end{bmatrix} \tag{3.6}$$

La ecuación 3.6 implica esencialmente que un punto 3D expresado en el sistema de la cámara puede ser parametrizado por la profundidad y la imagen de retina. Todas las propiedades de este modelo serán importantes recursos a la hora de realizar la reconstrucción 3D aquí propuesta.

## 3.3 Modelo de deformación isométrica

La reconstrucción de objetos deformables a partir de imágenes es un problema mal condicionado, debido a que diferentes deformaciones pueden resultar en las mismas proyecciones. Para poder eliminar estas ambigüedades es necesario emplear ciertas restricciones de deformación, como puede ser la rigidez en el caso de SfM. Una restricción muy utilizada en los métodos de reconstrucción deformable es la isometría.

La restricción de deformación isométrica impone que todas las distancias geodésicas entre puntos de la superficie se mantengan inalteradas con la deformación. Esto implica que la superficie se deforma sin experimentar estiramientos o compresiones.

Dado que muchas superficies naturales se deforman de manera isométrica o cercana a la misma, dicha restricción ha permitido resolver la reconstrucción no-rígida en muchos problemas prácticos. Cabe destacar que la rigidez es un caso particular de las deformaciones isométricas. De hecho, la isometría se puede entender como una rigidez infinitesimal. Sin embargo, la restricción de isometría es mucho más débil que la rigidez.

De forma matemática, se tiene una superficie $\mathcal{T}$ que sufre una deformación isométrica para dar lugar a la superficie $\mathcal{S}$. Dados dos puntos $P_i^{\mathcal{T}}$ y $Q_j^{\mathcal{T}}$ que pertenecen a $\mathcal{T}$ y sus correspondientes puntos $P_i^{\mathcal{S}}$ y $Q_j^{\mathcal{S}}$ en la superficie deformada $\mathcal{S}$, se cumple la siguiente igualdad:

$$G_{(P_i^{\mathcal{T}}, Q_j^{\mathcal{T}})} = G_{(P_i^{\mathcal{S}}, Q_j^{\mathcal{S}})}, \tag{3.7}$$

donde $G_{(P_i^{\mathcal{T}}, Q_j^{\mathcal{T}})}$ y $G_{(P_i^{\mathcal{S}}, Q_j^{\mathcal{S}})}$ son las distancias geodésicas en la plantilla 3D de la superficie y en la superficie deformada respectivamente. La figura 3.2 muestra gráficamente la deformación isométrica de una superficie plana.

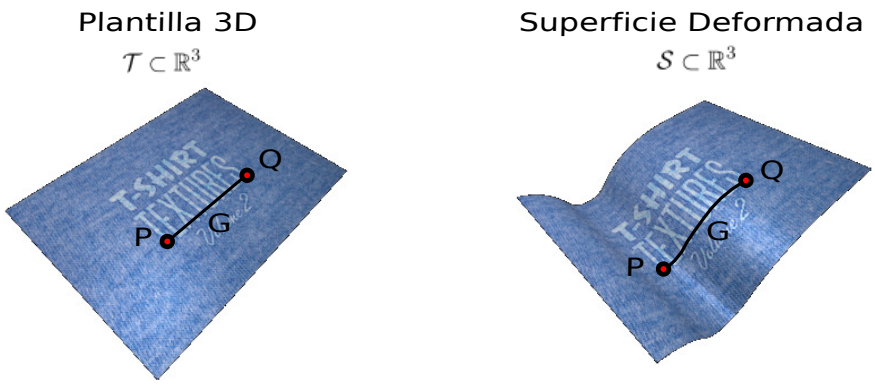



Figura 3.2: Ejemplo de aplicación de la isometría

En general, las distancias geodésicas en superficies son difíciles de calcular y por tanto la ecuación 3.7 no se suele utilizar directamente para resolver SfT. De manera alternativa, la isometría se puede modelar con facilidad utilizando herramientas de la geometría diferencial y da lugar a ecuaciones en derivadas parciales no lineales; utilizadas y estudiadas en los métodos SfT diferenciales comentados en el capítulo anterior. Por otro lado, en los métodos de orden cero o basados en mallas tridimensionales, la isometría se aproxima mediante la conservación de las distancias entre vértices vecinos.

## 3.4 Redes Neuronales Convolucionales

En esta sección se va a proceder a hablar de los modelos de redes neuronales convolucionales más utilizados y de las principales herramientas que emplean para su entrenamiento y validación.

Las CNN son un tipo de redes neuronales artificiales diseñadas para funcionar de forma muy similar a las neuronas de la corteza visual primaria de un cerebro humano. Estas han resultado ser ampliamente eficaces en tareas fundamentales de la visión artificial como la clasificación y la segmentación de imágenes. Dichas redes están formadas por múltiples capas de filtros convolucionales de una o más dimensiones, tras las cuales se insertan funciones no lineales de activación. Por ejemplo, en el caso de una clasificación clásica mediante una red convolucional, es posible encontrar dos fases bien delimitadas:

1. **Extracción de características:** Esta es la fase inicial y esta compuesta principalmente por neuronas convolucionales que asemejan su procesado al de la corteza visual humana. Cuanto mas se avanza a traves del número de capas convolucionales menos reaccionan estas ante la variación de los datos de entrada y mayor es la abstracción alcanzada por las mismas para reconocer formas mas complejas.

2. **Clasificación:** Se basan en la utilización de capas "Densas" formadas por neuronas convencionales, similares a las utilizadas por los modelos de tipo "perceptron".

### 3.4.1 Fundamentos y capas más importantes

A continuación se explicarán los tipos de capas de neuronas empleados en este tipo de redes neuronales y las partes más importante de las mismas:

- **Capas convolucionales:** Las capas convolucionales operan sobre los datos de entrada mediante el cálculo de convoluciones discretas con bancos de filtros finitos, tal y como se ve en la figura 3.3.

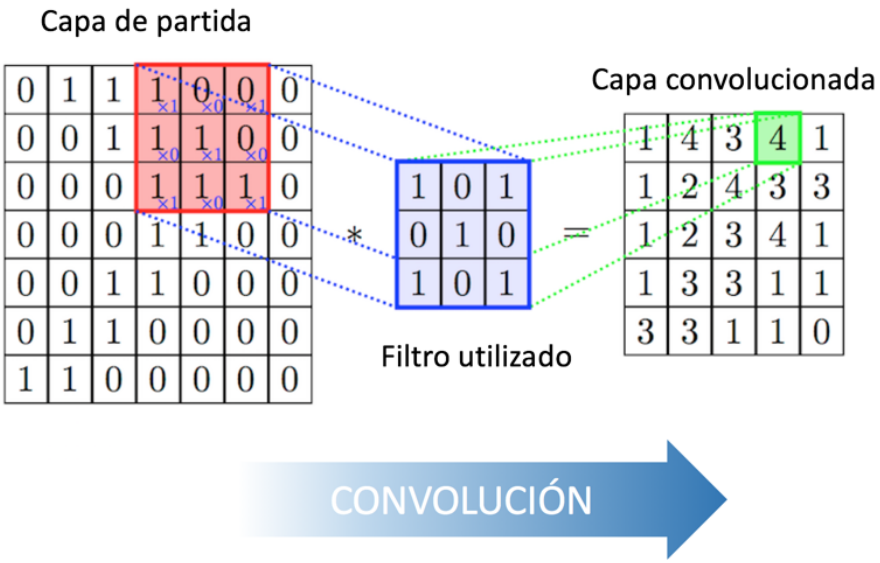



Figura 3.3: Ejemplo de convolucion

En este tipo de capas, las operaciones de convolución permiten obtener características dominantes de la imagen de entrada relacionadas con los objetivos de entrenamiento. De forma experimental se observa que las primeras capas en redes de convolucionales se centran en la búsqueda de características simples, como podrían ser bordes, esquinas o regiones. A medida que se avanza hacia capas más profundas, se aumenta el nivel de abstracción del contenido de la imagen al que se muestran sensibles, tal y como se observa en la figura 3.4.

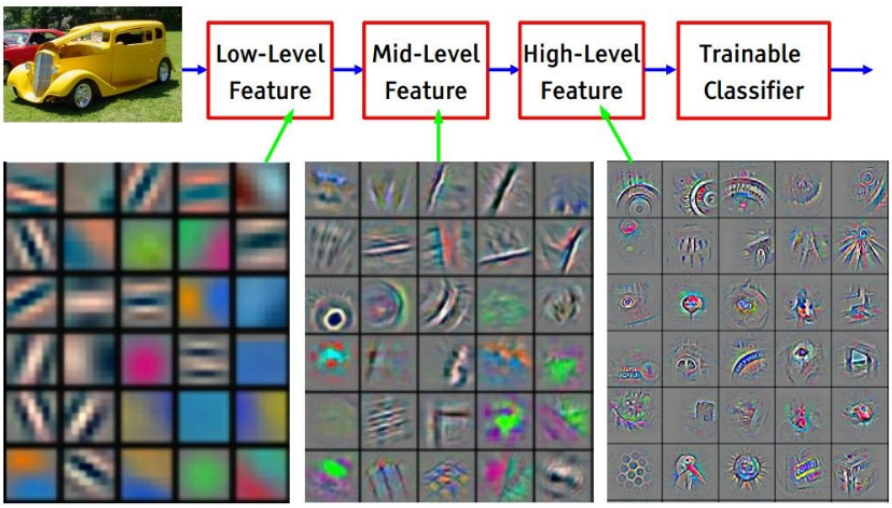



Figura 3.4: Ejemplo de filtros a diferentes niveles de abstracion

- **Densas o Fully Connected:** Este tipo de capas están representadas por las neuronas clásicas empleadas en los ya conocidos perceptrones, tal como se observa en la figura 3.5. Su función suele ser principalmente la de completar el clasificador final, que será el encargado de pasar de mapas de características a valores concretos en función del objetivo de la red (clasificación o regresión).

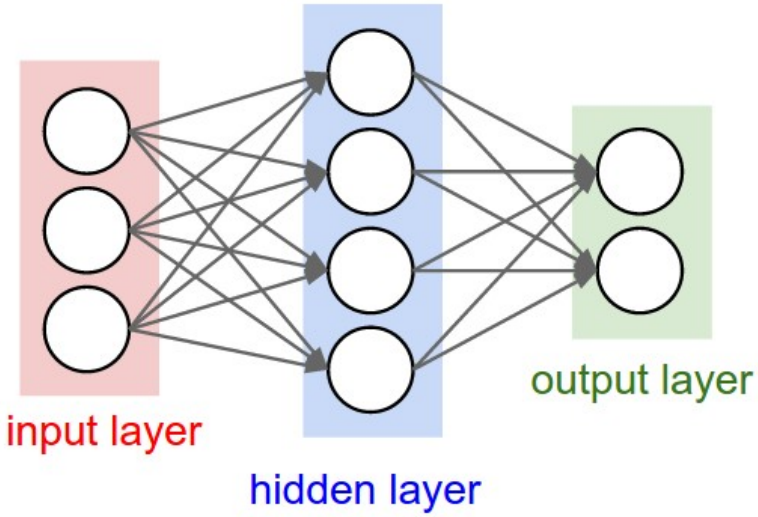



Figura 3.5: Ejemplo de capas Densa

- **Capas de activación:** Estas capas son las encargadas de aportar no linealidad a las funciones generadas por las redes neuronales y de agregar las activaciones de múltiples capas en la salida de la red.

  - **Lineal:** La función lineal es bastante conocida por ser empleada en problemas de regresión y se encuentran generalmente en la salida de la red. Un ejemplo de función de activación lineal se muestra en la imagen 3.6.

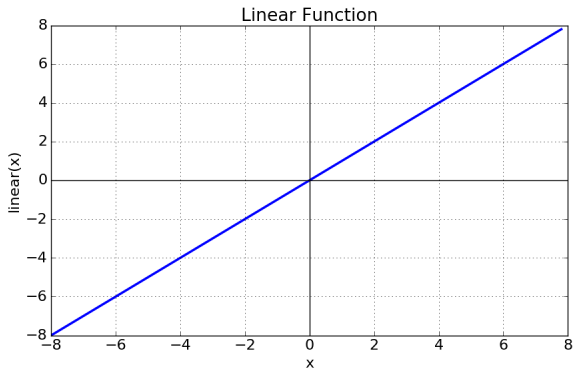



Figura 3.6: Ejemplo de activación lineal

  - **Unidad Lineal Rectificada o Relu:** La función Relu es una función de activación muy utilizada en las redes neuronales actuales. Esto es debido a que se demuestra experimentalmente [65] que dicho tipo de activaciones permiten redes mas profundas y facilitan el entrenamiento de las mismas. La función Relu se muestra en la figura 3.7.

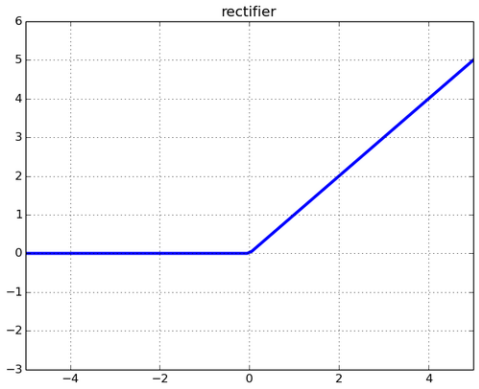



Figura 3.7: Ejemplo de activación relu

Las funcion de activación Relu presenta una serie de ventajas frente a otro tipo de función de activación:

* **Tiene menor numero de problemas de desvanecimiento de gradiente:** la ausencia de saturaciones en la función Relu permite mitigar el problema del desvanecimiento del gradiente que tienen otras funciones (por ejemplo la activación sigmoidal).
* **Tiene una mayor eficiencia computacional:** Esto es debido principalmente a que no requiere operaciones complejas para calcular la función y sus derivadas.
* **Invariante a escala:** Un cambio de escala en la entrada se ve reflejado en un cambio de escala en la parte lineal de la función.

– **Sigmoidal:** La función sigmoidal o funcion de activación logística es la función de activación clásica en las redes neuronales. Es una función diferenciable, monótona y se utiliza especialmente en problemas de clasificacion binaria dado que ofrece valores entre 0 y 1. Un ejemplo de la misma se observa en la figura 3.8.

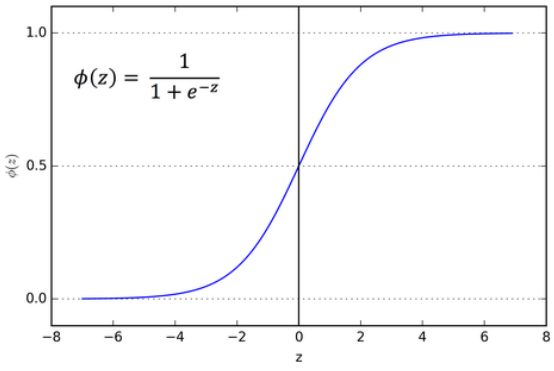



Figura 3.8: Ejemplo de función sigmoidal

– **Softmax:** La funcion softmax es una extensión de la clásica función logística, empleada principalmente para clasificación multiclase.

– **Tangente Hiperbólica:** La tangente hiperbólica es bastante parecida a la funcion sigmoidal pero permite activaciones en el rango -1 y 1. Es una funcion diferenciable y monotona.

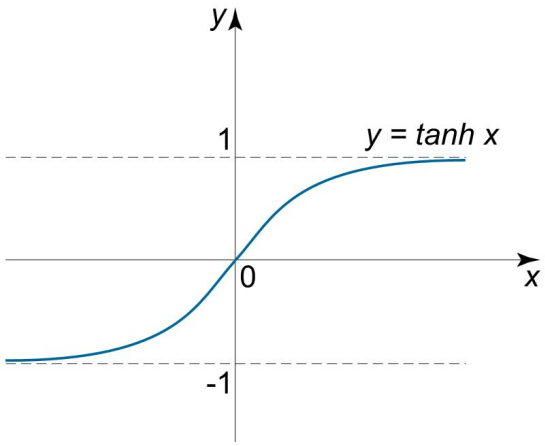



Figura 3.9: Ejemplo de activación de tangente hiperbólica.

### 3.4.2 Capas y Operaciones Adicionales

Las CNN cuentan con capas especiales, algunas de las cuales se activan sólo durante la fase de entrenamiento. Destacan las siguientes:

- **Max-Pooling:** El filtro max-pooling es una forma de reducción del volumen de salida de las capas convolucionales de la CNN y que permite además incrementar el campo de percepción de la red. Su acción sobre el resultado de las capas convolucionales se observa en las figuras 3.10 y 3.11.

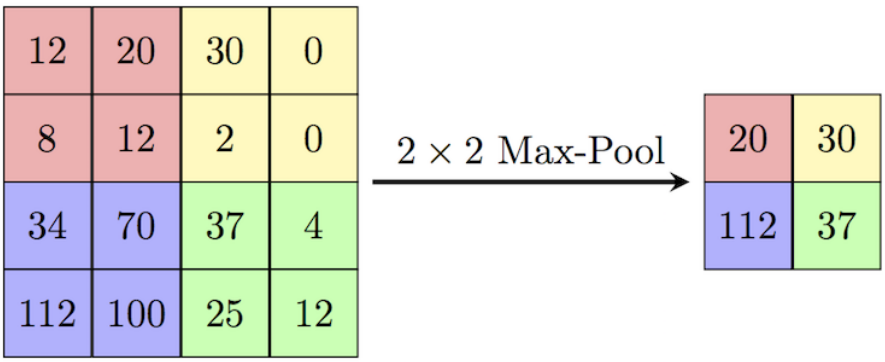



Figura 3.10: Ejemplo de matriz 4x4 en la cual se realiza un maxpooling de 2x2 con un stride de 2 para evitar solapamiento de regiones

- **Dropout:** La capa de Dropout es una capa de regularización muy empleada para evitar el overfitting o sobreentrenamiento en las CNN. Este termino se refiere a la eliminación de las contribuciones de ciertas neuronas junto a sus conexiones de entrada y salida. Dicha eliminación se realiza de forma aleatoria con una probabilidad de eliminación definida previamente. Los efectos del Dropout se estudian ampliamente y se demuestran en el articulo [66]. En la figura 3.12 se muestra el funcionamiento del mismo.

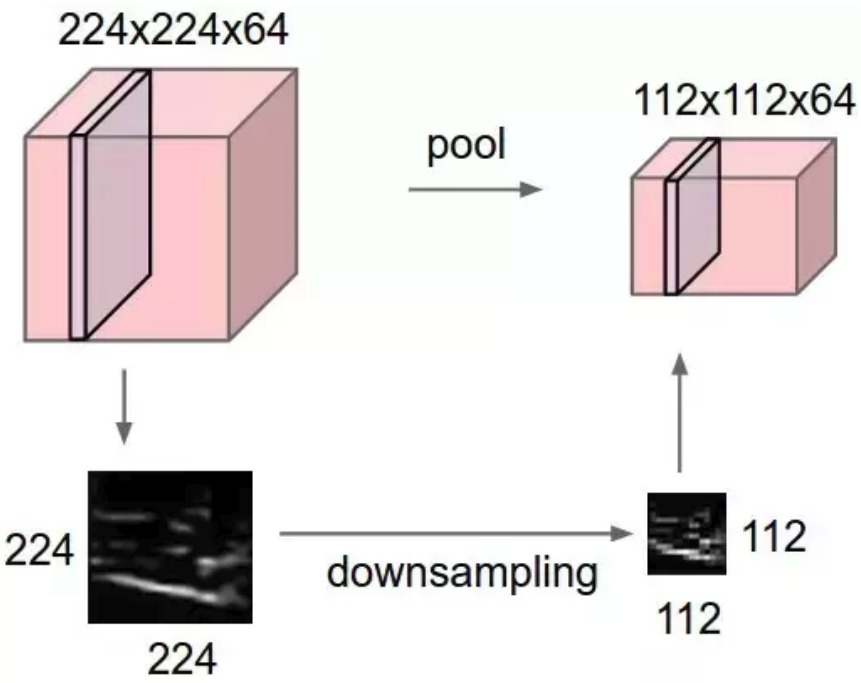



Figura 3.11: Ejemplo real de maxpooling

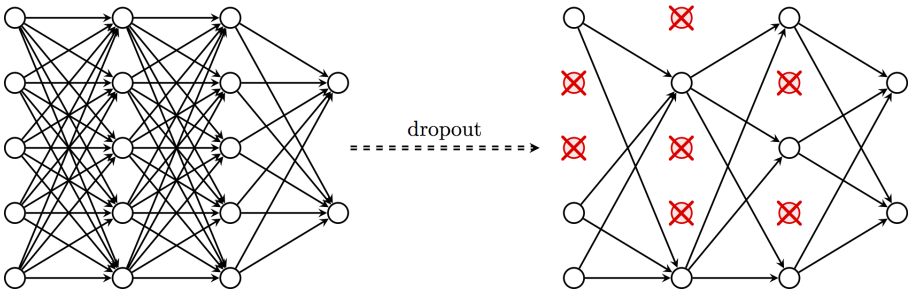



Figura 3.12: Ejemplo de apliación de Dropout

- **Batch Normalization:** La capa de normalización de batch tiene por objetivo aumentar la estabilidad del entrenamiento de la CNN. Esta capa tiene un efecto de regularización en la red mediante la normalización de las salidas de las capas de activación anteriores. Actúa restando a estas la media del batch y dividiendo por su desviación típica. Esta capa requiere de dos parámetros entrenables, de tal manera que los datos normalizados son multiplicados por un parámetro de desviación típica y un parámetro de media. De esta manera, el optimizador puede deshacer la normalización previamente comentada para así asegurar la estabilidad del entrenamiento del sistema. El funcionamiento de dicha capa y los resultados que prueban su eficacia, se estudiaran en mayor profundidad en [23].

### 3.4.3 Optimizadores mas importantes

En el marco de los optimizadores más importantes orientados a las CNN, encontramos una gran variedad de ellos y que derivan del algoritmo de descenso por gradiente estocástico SGD [67]. En este algoritmo se seleccionan conjuntos de entrenamiento de pequeño tamaño de forma aleatoria, también conocidos como “batches” de datos. Estos son utilizados para realizar una iteración de descenso por gradiente para

minimizar la función de pérdidas del entrenamiento. El algoritmo SGD tiene como principal parámetro a ajustar la magnitud del descenso o "learning rate".

A partir de SGD se han desarrollado otros métodos que varían dinámicamente el parámetro "learning rate" durante el entrenamiento. Dentro de los optimizadores adaptativos clásicos encontramos los dos principales: Adagrad [68] y RMSprop [69], que a posteriori derivaron en el caso de Adagrad en Adadelta [70] y Adam [71] y, en el caso de RMSprop, en en Nadam [72].

En este trabajo el optimizador prioritario será el optimizador adaptativo Adam, ya que sus características adaptativas y los resultados reportados por diferentes benchmarks de optimización lo sitúan como un buen candidato para el entrenamiento de redes neuronales.

#### 3.4.3.1 ADAM

El optimizador Adam es un algoritmo basado en optimización de primer orden de los gradientes de las funciones objetivo. Este algoritmo se encarga de estimar de forma adaptativa los momentos de ordenes bajos, en este caso primer y segundo orden. Las ventajas de este algoritmo yacen en la simplicidad de su implementación, su eficiencia computacional y su buen funcionamiento en problemas con gran numero de datos y parámetros. Su nombre es derivado de la estimación de momentos adaptativa. Este método solo requiere los gradientes de primer orden y determina el ratio de entrenamiento adaptativo mediante el primer y segundo momento de dichos gradientes. Adam esta diseñado para combinar las ventajas de RMSprop y Adagrad.

Para poder poner en funcionamiento este algoritmo de optimización solo es necesario aportarle 4 datos iniciales, que corresponderían con $l_r$ o "learning rate", $\beta_1$ o ratio de caída del primer momento, $\beta_2$ o ratio de caída del segundo momento y $\theta_0$ o vector de parámetros iniciales. A continuación se muestra un pequeño pseudocódigo ejemplificando el funcionamiento de Adam 3.13:

**Algorithm 1:** *Adam*, our proposed algorithm for stochastic optimization. See section 2 for details, and for a slightly more efficient (but less clear) order of computation. $g_t^2$ indicates the elementwise square $g_t \odot g_t$. Good default settings for the tested machine learning problems are $\alpha = 0.001$, $\beta_1 = 0.9$, $\beta_2 = 0.999$ and $\epsilon = 10^{-8}$. All operations on vectors are element-wise. With $\beta_1^t$ and $\beta_2^t$ we denote $\beta_1$ and $\beta_2$ to the power $t$.

**Require:** $\alpha$: Stepsize
**Require:** $\beta_1, \beta_2 \in [0, 1)$: Exponential decay rates for the moment estimates
**Require:** $f(\theta)$: Stochastic objective function with parameters $\theta$
**Require:** $\theta_0$: Initial parameter vector
  $m_0 \leftarrow 0$ (Initialize 1st moment vector)
  $v_0 \leftarrow 0$ (Initialize 2nd moment vector)
  $t \leftarrow 0$ (Initialize timestep)
  **while** $\theta_t$ not converged **do**
    $t \leftarrow t + 1$
    $g_t \leftarrow \nabla_\theta f_t(\theta_{t-1})$ (Get gradients w.r.t. stochastic objective at timestep $t$)
    $m_t \leftarrow \beta_1 \cdot m_{t-1} + (1 - \beta_1) \cdot g_t$ (Update biased first moment estimate)
    $v_t \leftarrow \beta_2 \cdot v_{t-1} + (1 - \beta_2) \cdot g_t^2$ (Update biased second raw moment estimate)
    $\widehat{m}_t \leftarrow m_t / (1 - \beta_1^t)$ (Compute bias-corrected first moment estimate)
    $\widehat{v}_t \leftarrow v_t / (1 - \beta_2^t)$ (Compute bias-corrected second raw moment estimate)
    $\theta_t \leftarrow \theta_{t-1} - \alpha \cdot \widehat{m}_t / (\sqrt{\widehat{v}_t} + \epsilon)$ (Update parameters)
  **end while**
  **return** $\theta_t$ (Resulting parameters)

Figura 3.13: Pseudocódigo de Adam

Por último, en las figuras 3.14 y 3.15 se muestran comparaciones de Adam con otros optimizadores similares donde se evalúa el entrenamiento de redes neuronales para los challenges Cifar10 [73] y Mnist [74].

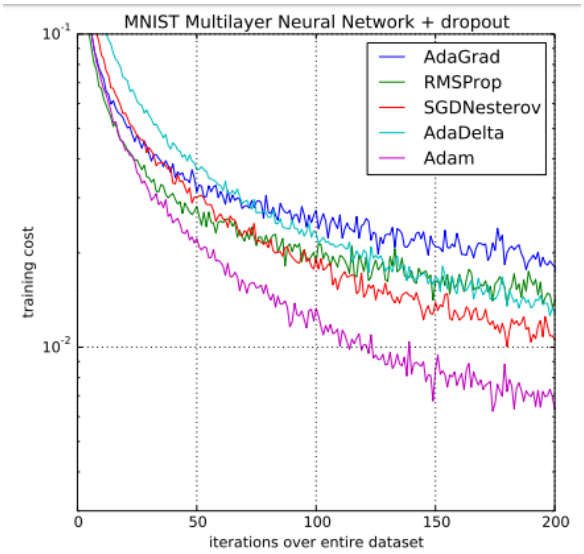



Figura 3.14: Comparativa Mnist

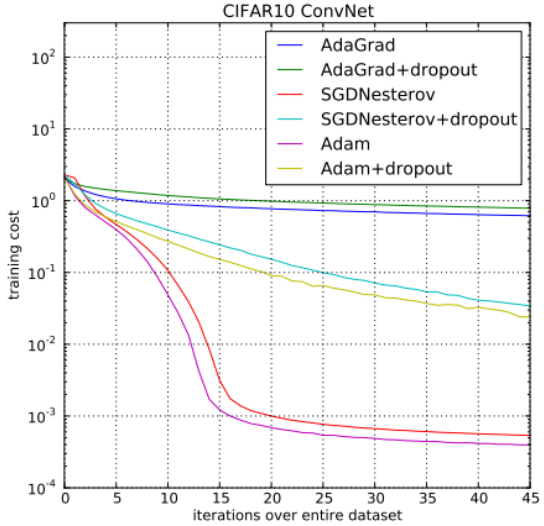



Figura 3.15: Comparativa CIFAR-10

Se observa que Adam es aparentemente el mejor algoritmo, dado que consigue minimizar la función de pérdidas de forma mas eficiente que sus competidores. Cabe destacar que el clásico SGD+Nesterov lo sigue bastante de cerca en el caso de Cifar-10. En todo caso los resultados empíricos aportados sitúan a Adam como un gran candidato para optimizar el problema a tratar en este trabajo. Para una mayor información sobre Adam se deberá remitir al articulo [71].

### 3.4.4 Arquitecturas mas importantes de CNN

Existen determinadas arquitecturas de redes CNN que han sido ampliamente utilizadas en visión artificial como componentes principales en multitud de aplicaciones. Las primeras redes utilizadas fueron la arquitectura Alexnet [75] y las redes Vgg16 y la Vgg19 [76], ya consideradas como redes clásicas. En la actualidad destacan la arquitectura ResNet [22], las redes con convoluciones multiescala Inception v1 [77], v2 [78] y v4 [79] y la red Xception [80] que incluye convoluciones separables.

Se describe a continuación la arquitectura ResNet [22], cuyos bloques residuales serán utilizados en este trabajo para crear la arquitectura de la red propuesta:

- **ResNet:** La arquitectura ResNet [22] es una de las arquitecturas más conocidas a día de hoy debido a sus propiedades residuales. Este tipo de redes residuales permiten un entrenamiento más rápido y estable que las redes clásicas, debido principalmente a que emplean una recirculación de residuos en sus capas. Estos bloques específicos consiguen aumentar en gran medida la profundidad de las redes sin desestabilizarlas. Es necesario comentar que una gran característica de esta red que tambien influye en gran medida en la velocidad de entrenamiento y estabilidad del mismo, es la inclusión de la normalizacion del batch [23]. Tal y como se comento previamente, esta técnica permite normalizar las salidas y tiene un impacto considerable en el entrenamiento de la red.

  A continuación se muestran los bloques que componen las capas ResNet:

  - **Bloques identidad:** El bloque identidad incluye 3 capas convolucionales con filtros de 1x1,3x3 y 1x1. Tras dichas capas se sitúan capas de normalización de batch y por último activaciones de tipo ReLu. Este tipo de bloques permiten procesar la información de entrada y sumarla a la información de salida. El nombre de dichos bloque es identidad dado que en su conjunto las dimensiones de entrada y las de salida serán las mismas, tal y como se observa en la imagen 3.16.

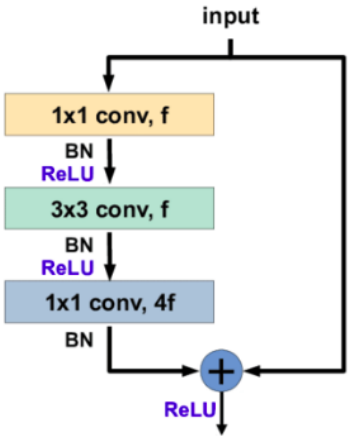



Figura 3.16: Bloque residual de tipo identidad

  - **Bloques convolucionales:** Los bloques convolucionales de la ResNet actúan, en cuanto a tamaños de entrada y salida, como una capa convolucional. Estos bloques incluyen, al igual que los anteriores, una rama con 3 convoluciones de 1x1,3x3 y 1x1, junto a capas de normalización de batch y activaciones ReLu. Estos bloques incluyen una la rama paralela, que en este caso se llama acceso directo, que incluye otra capa convolucional de 1x1. De esta manera, la información recirculada y sumada a la salida no es exactamente la de la entrada, sino que modifica los tamaños como si de una capa convolucional convencional se tratase. Este tipo de bloques permiten el procesado de la información de entrada a diferentes escalas convolucionales, de tal manera que la salida estara compuesta por un conjunto de salidas filtradas a diferentes niveles tal y como se observa en la figura 3.17.

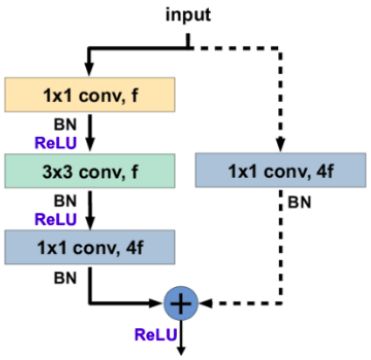



Figura 3.17: Bloque de tipo convolucional

# Capítulo 4

# Solución Propuesta

*La ciencia puede divertirnos y fascinarnos, pero es la ingeniería la que cambia el mundo.*

Isaac Asimov

## 4.1 Definicion del problema

A continuación se realizará una pequeña introducción del problema a resolver, su interés científico y como sería posible realizarlo.

1. **Reconstrucción tridimensional:** La mayor parte de los métodos del estado del arte en SfT proponen resoluciones iterativas o que dificilmente funcionarían en tiempo real. Además funcionan mediante la aplicación de modelos bastante restrictivos que les permiten condicionar bien el problema. Uno de los grandes problemas de dichos métodos es que algunos de ellos requieren de un registro previo de la plantilla con respecto a la superficie ya deformada.

2. **Registro:** El segundo problema a resolver abarca el registro de una plantilla de referencia con respecto a la superficie deformada, de tal manera que se pueda corresponder cualquier punto de la plantilla con su respectiva correspondencia en el frame deformado. La mayor parte de los métodos del estado del arte requieren de una secuencia de video en la cual la variación de movimiento de unos frames a otros sea suave. Las grandes lacras de dichos métodos son:

    - **La necesidad de una secuencia de video**: La gran mayoría de los métodos de registro denso requieren de una secuencia de video en la cual no haya grandes variaciones entre la plantilla y el frame deformado.
    - **Problemas de correspondencia:** En muchos casos surgen grandes problemas que invalidan los métodos existentes, como el "motion blur", los grandes cambios de luz y perspectiva, las oclusiones y auto-oclusiones y otros similares.

El problema aqui planteado busca poder realizar una reconstrucción tridimensional densa al mismo tiempo que un registro denso de la imagen deformada con respecto a una plantilla de referencia. El algoritmo propuesto se basa en el uso de una arquitectura CNN que permita trabajar con deformaciones cuasisométricas y que funcione a una velocidad cercana al tiempo real.

## 4.2 Modelado Matemático

En este apartado se va a proceder a explicar el modelado matemático del problema SfT. La figura esquemática 4.1 representa los pasos y recursos empleados en SfT.

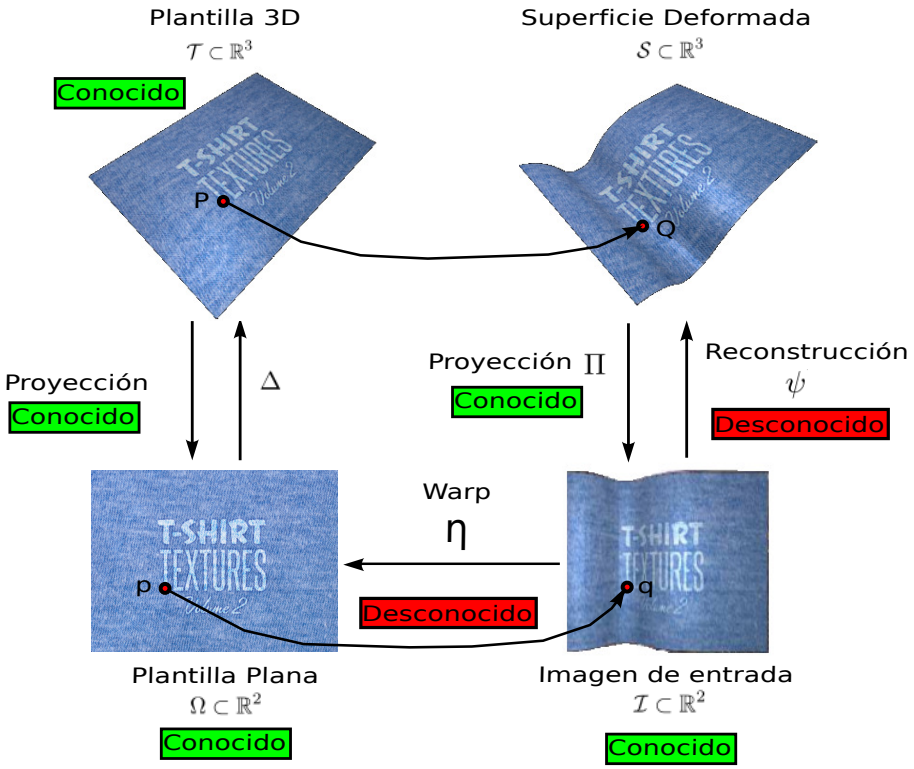



Figura 4.1: Figura de modelado

Dentro del problema SfT los datos que son conocidos para su uso son:

1. **Plantilla 3D**: La plantilla tridimensional $\mathcal{T} \subset \mathbb{R}^3$ es conocida como un dato inicial del algoritmo. Normalmente se representa como una malla tridimensional.

2. **Plantilla Plana**: La plantilla plana se obtiene mediante la función de aplanado $\triangle^{-1}$ de la plantilla tridimensional, cuyo dominio $\Omega \subset \mathbb{R}^2$ se obtiene de $\mathcal{T}$ y se parametriza con $\triangle \in C^1(\Omega, \mathbb{R}^3)$.

3. **Plano de retina de la imagen de entrada**: El plano de retina de la imagen de entrada se obtiene de la propia imagen de entrada $\mathcal{I}$ y los parámetros intrínsecos de la cámara $f_x, f_y, c_x, c_y$.

Es necesario destacar que dichas plantillas en la práctica se obtienen mediante el mapeo de texturas en mallas o simplemente capturando imagenes de los objetos tridimensionales. En este trabajo la función de reconstrucción $\psi \in C^1(\mathcal{I}, \mathbb{R}^3)$ tiene como dominio la imagen de entrada $\mathcal{I}$ y permite obtener la superficie ya deformada $S \subset \mathbb{R}^3$. De manera inversa, mediante los parámetros intrínsecos de la cámara $f_x, f_y, c_x, c_y$ se modela la proyección de $S \subset \mathbb{R}^3$ con la función de proyección de cámara $\Pi$.

La función de registro $\eta$, al contrario que en el SfT convencional, no se asume previamente conocida, sino que se hallará internamente por la CNN, pudiendo así registrar $\Omega$ y $\mathcal{I}$ como $\eta \in C^1(\Omega, \mathbb{R}^2)$.

La plantilla 3D $\mathcal{T}$ y la imagen de la plantilla $\Omega$ serán aprendidas de forma interna por la CNN a través del entrenamiento, permitiendo así eliminar la necesidad de introducir un mayor flujo de datos.

Por último la función de registro o Warp $\eta$ entre las imágenes $\Omega$ y $\mathcal{I}$ no será previamente conocida sino que será estimada por la CNN junto con la reconstrucción.

En este caso las funciones de restricción impuestas al algoritmo para condicionar bien el problema de SfT se introducirán de formas algo diferentes a las convencionales:

1. **Restricción de Deformación**: La restricción de deformación aquí expuesta vendrá dada por el entrenamiento realizado por la CNN con objetos que sufren deformaciones cuasi isométricas.

2. **Restricción de Sombreado**: Durante el entrenamiento se han incluido diferentes tipos de iluminaciones y sombras que provocarán que la CNN aprenda a lidiar con los cambios de luz suaves y a utilizar esa información para obtener reconstrucciones en caso de falta de textura en el objeto .Ejemplos de dichos cambios de luz y sombreados, son la eliminación de la componente difusa, luces rotatorias a lo largo de las simulaciones, la eliminación de la componente especular y las deformaciones con oclusiones con respecto al foco.

3. **Restricción de Reproyección**: La restricción de reproyección se impondrá en este caso mediante la función de coste empleada en el entrenamiento. Dicha función de coste será MSE, la cual minimizará el error tridimensional de la salida de la CNN, lo cual está intimamente relacionado con la proyección dado el modelo de perspectiva de la cámara.

4. **Restricción de Suavizado**: La restricción de suavizado se impondrá en este caso mediante la función de coste durante el entrenamiento con superficies que sufren deformaciones suaves.

Mediante la unión de todas estas restricciones aplicadas aunque de forma indirecta, se condicionará mejor el problema de SfT y se favorecerá que la CNN pueda aprender de forma mas eficaz a representar la superficie deformada $\mathcal{S}$ y el registro entre $\Omega$ y $\mathcal{I}$.

Una vez definidas las restricciones, se definirá matematicamente el formato de las salidas.

1. **Superficie deformada**: La superficie deformada $\mathcal{S}$, que puede ser expresada como $\mathcal{S} = [\mathcal{S}_x, \mathcal{S}_y, \mathcal{S}_z]$, será obtenida en forma de un mapa de profundidad $P \in \mathbb{R}^{WxH}$ cuyas dimensiones $H$ y $W$ serán idénticas a las de la imagen de entrada $\mathcal{I}$. Cada uno de los píxeles $P(u_i, v_i)$, cuyo valor no sea nulo, representará la componente $\mathcal{S}_z$ de uno de los puntos de la superficie deformada $\mathcal{S}$. De esta manera, las componentes $\mathcal{S}_x$ y $\mathcal{S}_y$ se hallarán mediante los parámetros de cámara $f_x, f_y, c_x, c_y$ utilizando la expresión $S_x = (\frac{u-c_x}{f_x})S_z$, $S_y = (\frac{v-c_y}{f_y})S_z$. Una vez halladas las tres componentes que conforman la superficie deformada $\mathcal{S}$, se puede recuperar la misma.

2. **Función de registro o Warp**: La función de registro $\eta$ será hallada directamente por la CNN en forma de dos subcomponentes de la función de warp $\eta = [\eta_u, \eta_v]$. Estas se expresan en forma de mapas $M_u \in \mathbb{R}^{WxH}$ y $M_v \in \mathbb{R}^{WxH}$ cuyas dimensiones de altura $H$ y anchura $W$ serán las de la imagen de entrada $\mathcal{I}$.

## 4.3 Arquitectura Propuesta

En este trabajo se proponen dos arquitectura CNN para resolver el problema SfT. En la primera, denominada arquitectura clásica, se utilizará una arquitectura de Convolucionales+Densas utilizada ampliamente en clasificación [22, 75, 76]. La segunda arquitectura propuesta estará basada en arquitecturas encoder-decoder "fully convolutional". Se demostrará que esta última arquitectura permite mejorar sustancialmente a los métodos del estado del arte en SfT.

### 4.3.1 Arquitectura clásica

La arquitectura clásica previamente comentada ha sido ampliamente empleada para labores de clasificación y determinación de mapas de probabilidad como por ejemplo en los trabajos [22, 75, 76] Esta arquitectura se compone de dos fases bien diferenciadas, como las mostradas en la siguiente figura 4.2:

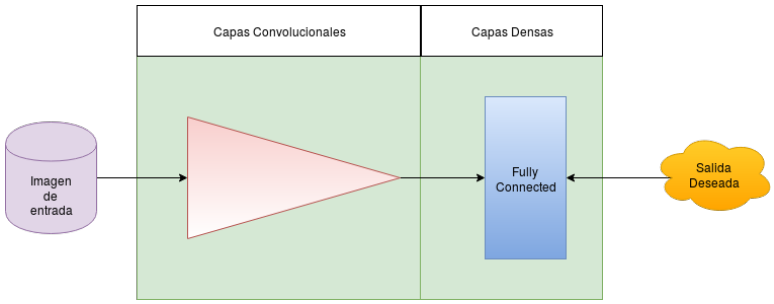



Figura 4.2: Esquema de Arquitectura clásica

En la figura anterior se pueden observar dichas fases, las cuales son:

1. **Fase convolucional:** En esta fase se someterá a la imagen de entrada a un conjunto de capas convolucionales conectadas de manera secuencial. Estas capas convolucionales serán las encargadas de extraer las características principales de la imagen de entrada y permitirán clasificar o realizar una regresión de las salidas indicadas. Esta extracción de características aumentará el nivel de abstracción de las características a medida que aumente la profundidad de la red.

2. **Fase Densa:** en esta fase se utilizarán capas de neuronas "Fully connected" o densas. Estas capas se componen de neuronas clásicas y permiten establecer una relación entre dichas características convolucionales y la salida de la red.

En este caso, para la fase convolucional, se emplearán capas residuales basadas en los bloques de la arquitectura ResNet [22]. Dichos bloques se muestran en la figura 4.3.

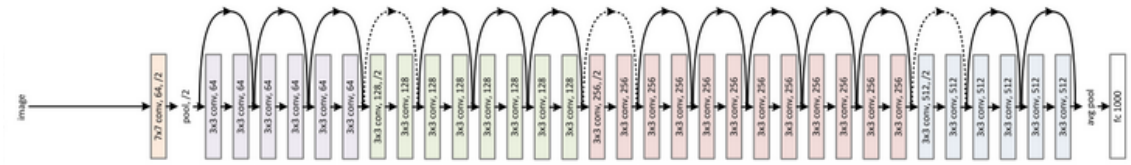


Figura 4.3: Ejemplo de arquitectura residual

Tal como se comento en el capítulo 3, los bloques residuales favorecerán la velocidad de aprendizaje y su convergencia en el entrenamiento, además de permitir añadir mayor profundidad a la red con menor coste computacional.

No obstante se demuestra en este trabajo que la arquitectura clásica compuesta por capas convolucionales+fully connected no es la arquitectura adecuada. Esto es debido a que el numero de parámetros requeridos es mayor y la capacidad de generalización de la misma es mucho más pobre. En este trabajo se ha empleado una configuración de red clásica similar a la de [22] con la excepción de que se empleará una capa fully connected de 200 neuronas y una capa de salida compuesta por la dimensiones de los mapas de registro y profundidad, es decir (480x270x3).

### 4.3.2 Arquitectura propuesta

En este apartado se va a proceder a comentar la arquitectura de red propuesta. Dicha arquitectura se ha diseñado a medida para resolver el problema SfT y su estructura se muestra en la figura 4.4.

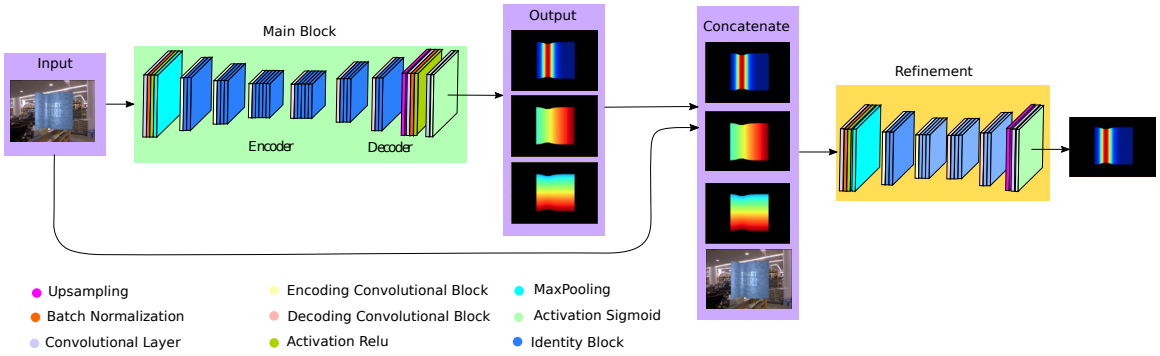



Figura 4.4: Arquitectura propuesta

En la figura anterior se pueden observar dos bloques bien diferenciados: el bloque principal o "Main Block" y el bloque de Refinado o "Refinement Block".

#### 4.3.2.1 ¿Que aporta esta arquitectura?

A continuación se discuten las características más destacables de la arquitectura propuesta:

- **Configuración Encoder-Decoder:** Con respecto a la arquitectura clásica de [22], esta arquitectura trata de emplear una arquitectura más eficiente en número de parámetros y mejor que las clásicas. La arquitectura Encoder-Decoder cumple dicho requisito y en este caso es bastante útil dada la relación espacial entre la entrada a la CNN y las salidas propuestas, que en este caso vienen representadas por los warps u,v y el mapa de profundidad de la superficie deformada.
- **Configuraciones residuales:** Tal y como se ha demostrado a lo largo del tiempo sobre el campo de las CNN, arquitecturas residuales tales como [22], [77] han batido a las clásicas redes no residuales tales como [75] o [76]. La causa de esto proviene de varias ramificaciones destacables.
  - **Pérdida de información**: Por una parte es bastante destacable el hecho de que a medida que se somete a la entrada a diferentes capas convolucionales, estas pierden cierta cantidad de información que aún podría ser de utilidad a la red. El hecho de recuperar la misma permite a la red disponer de una mayor cantidad de datos que le pueden ser útil a la hora de aprender.
  - **Normalización de Batch:** Tal como demuestra el artículo [23], la normalización de los batches empleados durante el entrenamiento permiten a las redes entrenar mucho más rapido, aplicar una pequeña regularización y mejorar los problemas de desvanecimiento de gradiente o saturaciones.
  - **Mayor número de parámetros:** Este tipo de configuraciones residuales permiten que las redes convolucionales dispongan de una mayor cantidad de parámetros entrenables con un menor coste computacional que el obtenido en otras arquitecturas [76].
- **Bloque de refinado:** El bloque de refinado incluido en esta arquitectura permite reutilizar la información expedida por el bloque principal y, además, reutilizar los datos de entrada para poder refinar la estimación del mapa de profundidad. Los bloques de refinado no son algo nuevo en las

redes convolucionales. Diferentes configuraciones de refinado han sido empleadas en ejemplos de redes convolucionales tales como [81], [82] y [83]. En este caso el refinado esta justificado dado que la combinación principalmente de las tres salidas expedidas por el bloque principal puede ser combinada para mejorar el mapa de profundidad obtenido. Ello puede ser reflejado en los métodos del estado del arte que emplean exclusivamente los warps para obtener el mapa de profundidad tales como [55]. En este caso la combinación de dichos warps, el mapa de profundidad expedido por la red y además la entrada de nuevo, pueden ayudar a que la información se complemente entre sí. Esto permite obtener así un mapa de profundidad que se ciña mejor a las deformaciones más complejas, como aquellas que son de alta frecuencia.

#### 4.3.2.2 Main Block

A continuación se va a proceder a hablar del Bloque principal que compone dicha arquitectura. Este bloque está compuesto por un Encoder-Decoder residual basado, tal y como se comentó anteriormente, en los bloques residuales de tipo [22]. A continuación se explicarán en detalle dichos bloques incluyendo diagramas de los mismos.

- **Bloque de Codificación Convolucional o Encoding Convolutional Block:** Este bloque está representado por 2 ramificaciones de cálculo, tal y como se muestra en la figura 3.17. Este bloque expide una salida con un tamaño similar al de una convolución clásica en función del tamaño de los filtros elegidos. Sin embargo, dentro de este bloque se realizan operaciones más complejas. En el caso del Bloque de codificación, se busca que cada uno de los subbloques convolucionales empleados procesen la información y aumenten la profundidad de la red. De esta manera, la segunda ramificación, llamada "shortcut" o acceso directo, emplea un solo subbloque convolucional sobre la entrada y se suma a la información ya procesada de salida. Este tipo de bloques permiten un filtrado de la información a diferentes escalas, una utilización más eficiente de los parámetros y una recirculación de los residuos e información que se puedan haber pérdido en la rama de procesado convolucional, tal y como se observa en la figura 4.5.

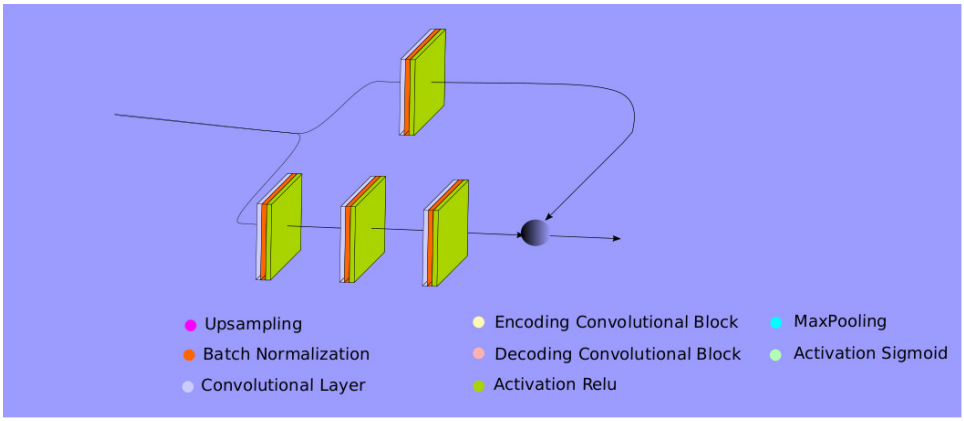



Figura 4.5: Bloque convolucional de codificación empleado

- **Bloque de Decodificación Convolucional o Decoding Convolutional Block:** Este bloque está representado por 2 ramificaciones de cálculo, tal y como se muestra en la figura 4.6. Su actuación es similar a la de el bloque de codificación, pero en este caso tiene un funcionamiento en modo deconvolución. Dado que en este tipo de librerías de redes convolucionales no suele emplear la operación de deconvolución como tal, se empleará en este caso una operación equivalente en tamaños

de salida, como puede ser Upsampling+Convolución 2D con un padding que compense la pérdida de tamaño. Esto se puede observar en la imagen 4.6.

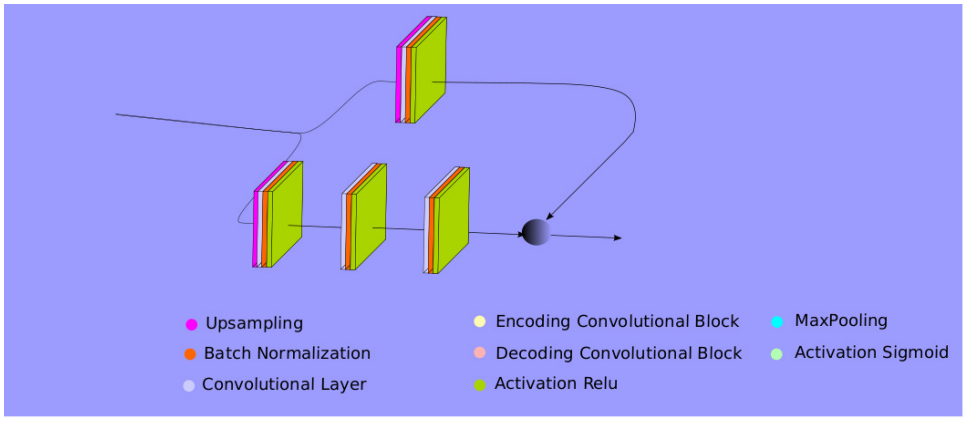



Figura 4.6: Bloque deconvolucional empleado en decodificación

- **Bloque Identidad:** Este bloque presenta 2 ramificaciones de cálculo, tal y como se muestra en la figura 3.16. La función de este bloque será la de procesar toda la información de entrada y recircular la entrada sin procesado. Esto resulta en la obtención de información filtrada junto con la información inicial, enriqueciendo así la entrada a la siguiente capa y evitando la pérdida de información debido a convoluciones. El esquema de dicho bloque se encuentra en la figura 4.6.

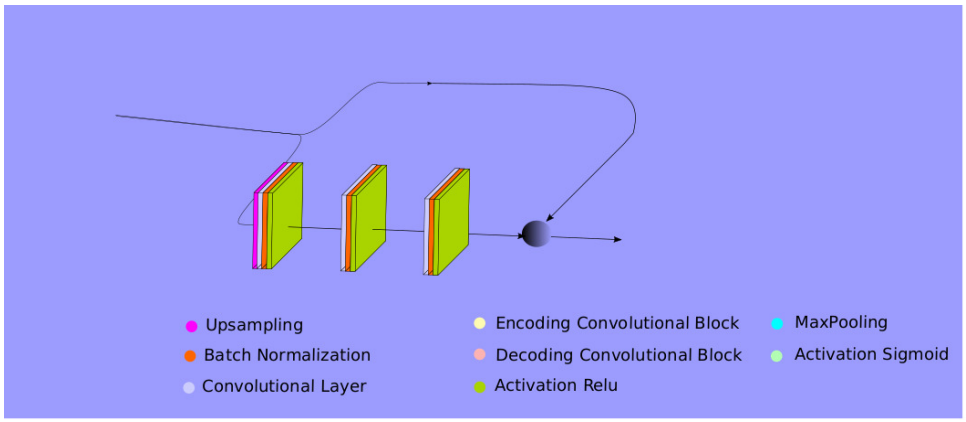



Figura 4.7: Bloque identidad empleado en codificación y decodificación

A continuación, en la tabla 4.3.2.2, se observa la transición del flujo de información a lo largo de las capas.

| Numero | Tipo | Tamaño de salida | Kernel or Activation |
|---|---|---|---|
| 1 | Input | (270,480,3) | – |
| 2 | Convolution 2D | (135,240,64) | (7,7) |
| 3 | Batch Normalization | (135,240,64) | – |
| 4 | Activation | (135,240,64) | Relu |
| 5 | MaxPooling 2D | (45,80,64) | (3,3) |
| 6 | Encoding Convolutional Block | (45,80,256) | (3,3) |
| 7 | Encoding identity Block | (45,80,256) | (3,3) |
| 8 | Encoding identity Block | (45,80,256) | (3,3) |
| 9 | Encoding Convolutional Block | (23,40,512) | (3,3) |
| 10 | Encoding identity Block | (23,40,512) | (3,3) |
| 11 | Encoding identity Block | (23,40,512) | (3,3) |
| 12 | Encoding identity Block | (23,40,512) | (3,3) |
| 13 | Encoding Convolutional Block | (12,20,1024) | (3,3) |
| 10 | Encoding identity Block | (12,20,1024) | (3,3) |
| 11 | Encoding identity Block | (12,20,1024) | (3,3) |
| 12 | Encoding identity Block | (12,20,1024) | (3,3) |
| 13 | Encoding identity Block | (12,20,1024) | (3,3) |
| 14 | Encoding identity Block | (12,20,1024) | (3,3) |

| Numero | Tipo | Tamaño de salida | Kernel or Activation |
|---|---|---|---|
| 15 | Decoding Convolutional Block | (12,20,256) | (3,3) |
| 16 | Encoding identity Block | (12,20,256) | (3,3) |
| 17 | Encoding identity Block | (12,20,256) | (3,3) |
| 18 | Encoding identity Block | (12,20,256) | (3,3) |
| 19 | Encoding identity Block | (12,20,256) | (3,3) |
| 20 | Encoding identity Block | (12,20,256) | (3,3) |
| 21 | Decoding Convolutional Block | (24,40,128) | (3,3) |
| 22 | Cropping 2D | (23,39,128) | (1,1) |
| 23 | Encoding identity Block | (23,39,128) | (3,3) |
| 24 | Encoding identity Block | (23,39,128) | (3,3) |
| 25 | Encoding identity Block | (23,39,128) | (3,3) |
| 26 | Decoding Convolutional Block | (46,78,64) | (3,3) |
| 27 | Zero Padding | (46,80,64) | (0,1) |
| 28 | Encoding identity Block | (46,80,64) | (3,3) |
| 29 | Encoding identity Block | (46,80,64) | (3,3) |
| 30 | Upsampling | (138,240,64) | (3,3) |
| 31 | Cropping 2D | (136,240,64) | (2,0) |
| 32 | Convolution 2D | (272,480,64) | (7,7) |
| 33 | Cropping 2D | (270,480,64) | (2,0) |
| 34 | Activation | (270,480,64) | Relu |
| 35 | Convolution 2D | (272,480,3) | (3,3) |
| 36 | Activation | (270,480,64) | Linear |

Tabla 4.1: Tabla de arquitectura de bloque principal

En la tabla anterior se muestra la arquitectura del bloque principal. Este emplea kernels de 7x7 coeficientes en las primeras capas. Esto reduce la cantidad de información tras obtener las activaciones máximas mediante un Maxpooling de (3,3). A partir de ese punto no se volverá a emplear un Maxpooling, dado que no es necesario segmentar más la información. A continuación del Maxpooling se utilizan bloques convolucionales e identidad, que provocarán una reducción de tamaño, hasta obtener tensores de un tamaño (12,20,256). Se puede observar que, a medida que se aumenta la profundidad, se añaden más bloques identidad. La causa de esto es que permite aumentar aún más el nivel de abstracción que las propias redes de por sí ofrecen. Esto es debido a que procesan aún más la información, obteniendo características más profundas y complejas.

Una vez termina la etapa de codificación, se procede a realizar los pasos inversos mediante bloques de decodificación. En este caso, dado que no modifican los tamaños de salida, los bloques identidad continúan

siendo los mismos. De esta manera se provoca de nuevo que el tamaño de las salidas aumente hasta las originales y se realiza un procesado mayor mediante bloques identidad.

Dado que estos tamaños pueden no ajustarse perfectamente, se aumentarán y disminuirán unidades de dimensión mediante las capas de "Cropping" y "Zero Padding".

Una vez se llega a las capas finales de salida, se llevará la profundidad de la salida a 3, dado que se buscan 3 mapas de salida diferentes. Además de ello se les aplicará una activación lineal, dado que en este caso se busca una regresión, siendo esta la mejor activación para dichos casos.

Es necesario apuntar que se emplearán continuamente normalizaciones de batch, que permitan aumentar la velocidad de entrenamiento y mejorar la estabilidad de la arquitectura. Por otro lado, las activaciones que se utilizan una y otra vez serán las llamadas Relu explicadas en el capítulo 3, excepto en la activación final.

#### 4.3.2.3 Refinement Block

A continuación se hablará del bloque de refinado situado posteriormente al bloque principal. Dicho bloque será el encargado de mejorar el mapa de profundidad obtenido. Esto lo hará gracias al empleo de la información predicha previamente y la recirculación de la información de entrada. La información del registro, junto a la información de profundidad, imponen ciertas restricciones de proyección que permiten la obtención de un mapa de profundidad más fiable. El hecho de recircular la información de entrada permite que la red de refinado se especialice más en hallar las deformaciones de alta frecuencia, dado que el bloque principal hallará el grueso de la información. En la tabla 4.2, se muestran cada una de las capas de la red de refinado.

| Numero | Tipo | Tamaño de salida | Kernel or Activation |
|---|---|---|---|
| 1 | Input | (270,480,3) | – |
| 2 | Convolution 2D | (135,240,64) | (7,7) |
| 3 | Batch Normalization | (135,240,64) | – |
| 4 | Activation | (135,240,64) | Relu |
| 5 | MaxPooling 2D | (45,80,64) | (3,3) |
| 6 | Encoding Convolutional Block | (45,80,256) | (3,3) |
| 7 | Encoding identity Block | (45,80,256) | (3,3) |
| 8 | Encoding identity Block | (45,80,256) | (3,3) |
| 9 | Encoding Convolutional Block | (23,40,512) | (3,3) |
| 10 | Encoding identity Block | (23,40,512) | (3,3) |
| 11 | Encoding identity Block | (23,40,512) | (3,3) |
| 12 | Encoding identity Block | (23,40,512) | (3,3) |
| 13 | Decoding Convolutional Block | (46,80,128) | (3,3) |
| 14 | Encoding identity Block | (46,80,128) | (3,3) |
| 15 | Encoding identity Block | (46,80,128) | (3,3) |
| 16 | Encoding identity Block | (46,80,128) | (3,3) |
| 17 | Decoding Convolutional Block | (92,160,64) | (3,3) |
| 18 | Cropping 2D | (90,160,64) | (2,0) |
| 19 | Encoding identity Block | (90,160,64) | (3,3) |
| 20 | Encoding identity Block | (90,160,64) | (3,3) |
| 21 | Upsampling | (270,480,64) | (3,3) |
| 22 | Convolution 2D | (270,480,1) | (3,3) |
| 23 | Activation | (270,480,64) | Linear |

Tabla 4.2: Tabla de arquitectura de bloque de refinado

Esta red de refinado es bastante similar a la previamente empleada en el bloque principal pero, a diferencia de la misma, es bastante más simple. Esto se lleva a cabo mediante la reducción de la profundidad de la red. De esta manera el bloque de refinado será mucho menos complejo que el principal,

dado que este se encargará de recoger la información procesada por el bloque principal y reutilizarla para mejorar el mapa final de profundidad.

## 4.4 Condiciones de entrenamiento

En esta sección se procederá a esclarecer y comentar todos aquellos puntos concernientes al entrenamiento de la red previamente propuesta. Para ello se empleará una enumeración en la cual, punto por punto, se pueda explicar cada uno de los parámetros empleados y la causa del empleo de los mismos.

1. **Optimizador:** El optimizador que se empleará en este caso será el optimizador adaptativo Adam, tal como se comentó previamente en la subsección 3.4.3. Esto es debido principalmente a que el tamaño de batch posible es lo suficientemente grande como para que un optimizador adaptativo funcione bien y, además, Adam ha demostrado que puede realizar entrenamientos mucho más rápidos y estables que el [67]. Recientemente se ha descubierto que Adam es ideal para un entrenamiento inicial, mientras que SGD permite realizar el reentrenamiento final de forma más estable [84].

2. **Función de pérdidas:** En este caso la función de pérdidas empleada es el MSE, que es una función adecuada para regresiones como las de este problema en concreto. Es necesario destacar que el problema de dicha función de pérdidas es que predomina la tendencia hacia la deformación media, y no penaliza los grandes errores en deformaciones, como podría realizar una función de pérdidas personalizada. Dicha función de pérdidas se muestra en la ecuación 4.1.

$$MSE = \frac{1}{n}\sum_{i=1}^{n} \|y_{real} - y_{predicha}\|^2 \tag{4.1}$$

   Donde $\|x\|$ denota la norma $L_2$ del vector $x$. Es importante destacar que, a la hora de mostrar los resultados, se empleará la raiz de la función $MSE$ para mostrar el error en unidades métricas, tal y como se muestra en el capítulo 5 y en la ecuación 4.2.

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n} \|y_{real} - y_{predicha}\|^2} \tag{4.2}$$

3. **Preprocesado de información:** En el caso del preprocesado de la información de entrenamiento, se hará un preprocesado convencional de la misma. Este preprocesado se realiza llevando su media a un valor de cero y su desviación típica a un valor de uno, de tal manera que la misma oscile entre 1 y -1. Es necesario para evitar saturaciones en la red y pesos excesivos que la desequilibren.

4. **Tamaño de Batch:** El tamaño de batch empleado ha sido de 15 imágenes por batch. Esto representa un batch con un tamaño medio-alto. De esta manera es posible generalizar correctamente y entrenar lo suficientemente rápido.

5. **Epocas:** En este caso se guarda el valor de la red en la epoca en que se obtiene mínimo error en el set de validación.

# Capítulo 5

# Resultados

*La evolucion forjo toda vida consciente en este planeta*
*con una sola herramienta: el error .*

Robert Ford

## 5.1 Introducción

A continuación se explican los diferentes resultados del trabajo de fin de master aquí expuesto. Inicialmente se comenta el entorno de experimentación empleado, para posteriormente proceder a la explicación de las bases de datos empleadas y su origen. Por último se explican las métricas de calidad que se emplean en este trabajo y los resultados experimentales sobre cada una de las bases de datos. En las dichas bases de datos se compara el método aquí propuesto con un método clásico del estado del arte y con una arquitectura clásica de CNN.

### 5.1.1 Bases de datos utilizadas

A continuación se va a proceder a hablar de las bases de datos empleadas en este trabajo de fin de máster. En este caso las bases de datos han sido creadas especificamente para el trabajo aqui mencionado, pudiendo dividir las mismas en dos fases diferentes, necesarias para poder llegar al resultado final del trabajo.

#### 5.1.1.1 Bases de datos sintetica

En primer lugar se va a comentar como se dio lugar la creación de una base de datos sintética de deformaciones para este trabajo.

Dada la gran complejidad para la creación de bases de datos reales en el campo de SfT, se propuso la creación de una base de datos fotorrealista, que permitiera realizar un primer entrenamiento de la red neuronal, para a posteriori realizar un pequeño reentrenamiento sobre una base de datos real mucho más pequeña que la anterior.

Dicha base de datos se crea mediante el programa Blender [16]. Dicho programa es muy empleado en diseño gráfico para realizar animaciones y simulaciones por ordenador. Dado que permite realizar simulaciones de cámaras, modelos de luz y sólidos deformables, encaja perfectamente en el trabajo aquí

expuesto. Mediante este programa se crean entornos virtuales tridimensionales, en los cuales se aplican restricciones a los sólidos de interés, en este caso un póster, que cumplirán modelos cuasisométricos. Dichos sólidos de interés serán sometidos a diferentes fuerzas que imiten el estiramiento de una tela, el ondeo de una bandera, o deformaciones similares. De esta manera se logran resultados como los expuestos en la figura 5.1.

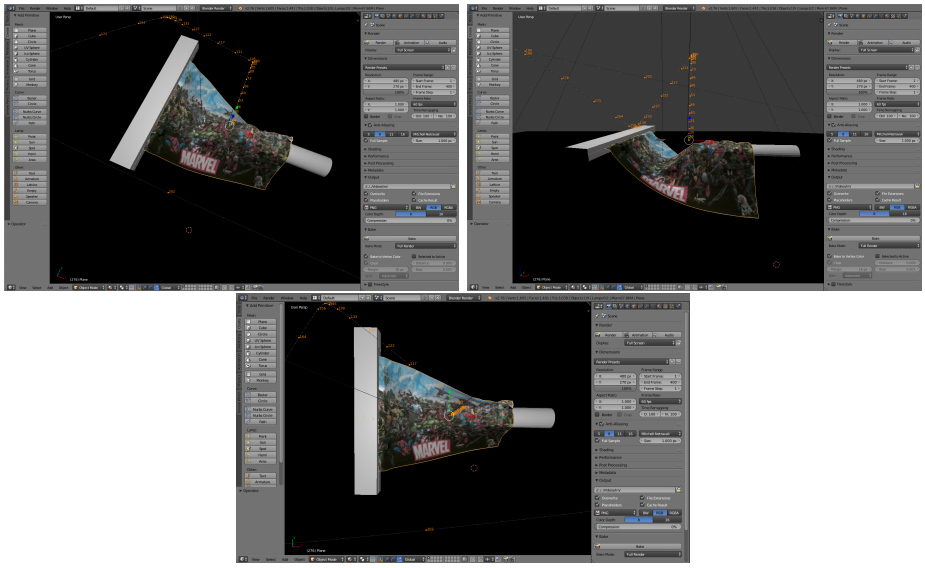


Figura 5.1: Ejemplos de imágenes generadas con el software Blender

Se puede observar en la figura 5.1 que el entorno 3D reproduce fielmente las deformaciones del póster al caer sobre una superficie cilíndrica a causa de la gravedad. Dicha deformación parte del reposo, de tal manera que se renderiza una secuencia entera. A lo largo de esta secuencia, el póster va deformándose poco a poco sobre la superficie. De esta manera, las renderizaciones tendrán como resultado tres salidas principales:

- **Malla**: Con el objetivo de poder hallar a posteriori los warps suavizados, se exportarán las mallas deformadas y sin deformar en formato .obj.
- **Imagen de color:** La imagen de color será renderizada y se le aplicará un modelo de iluminación para lograr un mayor realismo.
- **Imagen de profundidad:** La imagen de profundidad será a su vez renderizada por el buffer de profundidad de Blender y, una vez procesada, representará una de las salidas de la CNN.

Tras la definición de las salidas anteriores, la salida en formato de la imagen de color y la imagen de profundidad tendran un aspecto como el de la figura 5.2.

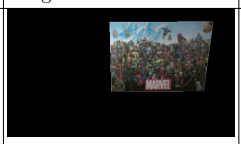
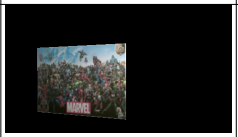
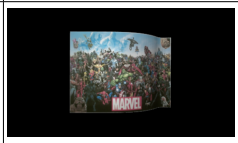
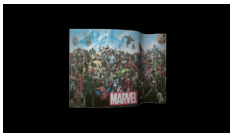
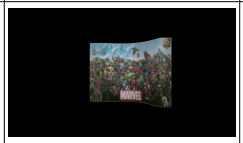
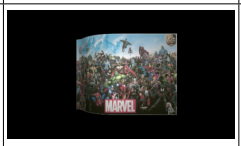
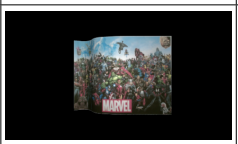
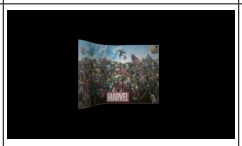
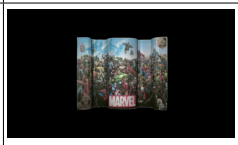
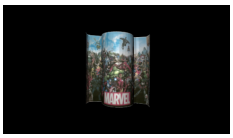
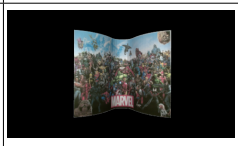
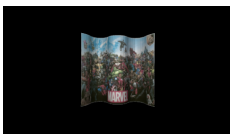
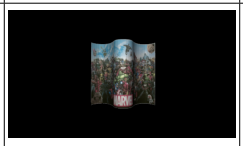
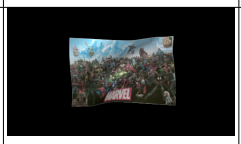
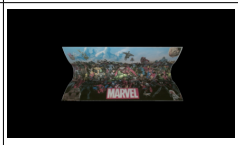
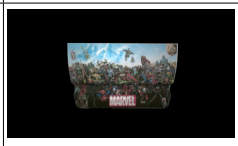
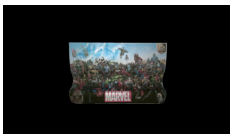
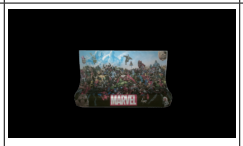
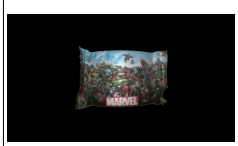
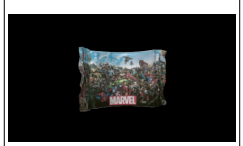
Figura 5.2: Ejemplo de la salida de Blender

Además de lo comentado anteriormente, es necesario comentar que dicha base de datos será creada con dos texturas diferentes. Esta condición será debida a que se buscará comprobar la eficacia de la CNN para funcionar con objetos con diferentes grados de riqueza en textura. Se utilizará una textura muy rica en características y mostrada en la figura 5.3 y una textura repetitiva y pobre en características como la representada en la figura 5.4.


Figura 5.3: Textura rica en características


Figura 5.4: Textura pobre en características

A continuación se van a describir y explicar cada una de las secuencias creadas en la base de datos sintética. Esta base de datos cubrirá un pequeño set de las infinitas deformaciones posibles. En los experiemntos mostraremos que la CNN propuesta tiene capacidad de reproducir deformaciones nunca vistas en el entrenamiento, siempre que éstas no difieran en exceso de las utilizadas en el entrenamiento. Tal como se comentó anteriormente, a pesar de ser las imágenes de mucha textura las representadas en la tabla, la base de datos será creada en las condiciones de poca 5.4 y alta textura 5.3. En la tabla siguiente se explican cada una de dichas secuencias en la tabla 5.1.

| Secuencias | Imagenes | | | Descripcion | Frames |
|---|---|---|---|---|---|
| 1 |  |  |  | Planos en diferentes posiciones y con diferentes ángulos y movimientos | 3000 |
| 2 |  |  |  | Poster con movimiento restringido por un prisma rectangular derecho y que es sometido a fuerzas de viento | 1200 |
| 3 |  |  |  | Poster con movimiento restringido por un prisma rectangular izquierdo y que es sometido a fuerzas de viento | 1200 |
| 4 |  |  |  | Poster con zona central en el eje x restringida por dos prismas rectangulares y sometido a fuerzas que provocan su ondeo | 400 |
| 5 |  |  |  | Poster cuya zona central a lo largo del eje x esta restringida y se le somete a fuerzas que provocan su ondeo y lo deforman | 1200 |
| 6 |  |  |  | Poster cuya diagonal esta restringida y se le somete a fuerzas que provocan su ondeo y lo deforman | 1200 |
| 7 |  |  |  | Poster cuya zona central a lo largo del eje x esta restringida y se le somete a fuerzas que provocan su ondeo y lo deforman | 1200 |
| 8 |  |  |  | Poster cuya diagonal esta restringida y se le somete a fuerzas que provocan su ondeo y lo deforman | 1200 |
| 9 |  |  |  | Poster con movimiento restringido por un prisma rectangular inferior y que es sometido a fuerzas de viento | 1400 |
| 10 |  |  |  | Poster con movimiento restringido por un prisma rectangular en los extremos izquierdo y derecho, sometido a grandes fuerzas | 1200 |
| 11 |  |  |  | Poster cuyas 4 esquinas estan restringidas en movimiento, y se somete a fuerzas que provocan su deformación en la direccion perpendicular al plano | 1400 |
| 12 |  |  |  | Poster con movimiento restringido por un prisma rectangular en los extremos izquierdo y derecho, siendo sometido a vientos de alta frecuencia | 1200 |

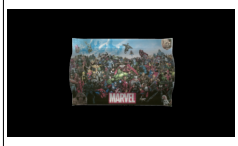
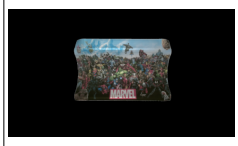
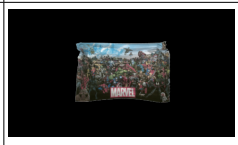
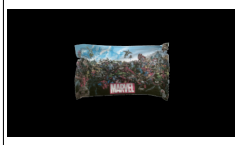
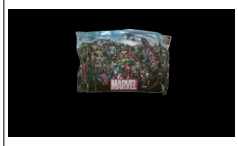
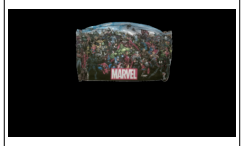
| Secuencias | Imagenes | | | Descripcion | Frames |
|---|---|---|---|---|---|
| 13 |  |  |  | Poster cuyo movimiento esta restringido por un prisma rectangular superior y que es sometido a fuerzas similares a las del viento provocando su ondeo | 1200 |
| 14 |  |  |  | Bandera cuyo movimiento esta restringido por extremo izquierdo y derecho y sometido a fuerzas que imitan su ondeo al viento | 1200 |
| 15 |  |  |  | Poster cuyo movimiento esta restringido por extremo izquierdo y derecho y sometido a fuerzas que estiran de la parte inferior central provocando oclusiones | 600 |
| 16 |  |  |  | Poster cuyo movimiento esta restringido por extremo izquierdo y derecho y sometido a fuerzas similares a las del viento en direcciones variables | 350 |
| 17 |  |  |  | Poster cuyo movimiento esta restringido por extremo izquierdo y derecho y sometido a fuerzas similares a las del viento en la dirección perpendicular al plano | 600 |
| 18 |  |  |  | Poster cuyo movimiento esta restringido por extremo izquierdo y derecho y sometido a fuerzas similares a las del viento en la dirección paralela al plano | 250 |
| 19 |  |  |  | Poster cuyo movimiento esta restringido por extremo izquierdo y derecho y una esfera y una superficie toroidal lo presionan provocando estiramiento | 400 |
| 20 |  |  |  | Poster cayendo sobre una superficie cilíndrica con una restriccion de movimiento en el extremo izquierdo creando deformaciones que se ajustan a la misma y forman oclusiones | 400 |
| 21 |  |  |  | Poster cayendo sobre una superficie toroidal con una restriccion de movimiento en el extremo izquierdo creando deformaciones que se ajustan a la misma | 300 |

Tabla 5.1: Secuencias empleadas para el entrenamiento de los algoritmos

Una vez explicadas cada una de las secuencias, queda comentar un último detalle acerca del fondo utilizado para la escena. Dado que se busca que la CNN pueda segmentar el objeto de interés del fondo, las imágenes sintéticas emplean fondos naturales aleatorios obtenidos de [85], resultando en figuras como las de 5.5:


Figura 5.5: Ejemplos de imágenes con fondos naturales

#### 5.1.1.2 Bases de datos real

A continuación se comenta como se creó la base de datos real de deformaciones. Las imágenes obtenidos del simulador Blender [16] se ajustan a la realidad en gran medida, pero no cuentan con las imperfecciones que se pueden dar en la realidad, tales como tolerancias en la deformación, imperfecciones, etc. Es, por tanto, necesario realizar un rentrenamiento sobre una base de datos real, que contenga datos directamente obtenidos desde el sensor, en este caso la cámara RGB de [6]. La base de datos presentada a continuación se ha creado mediante un póster de tamaño similar al simulado en blender e impreso con la temperatura de color más cercana a la plantilla ideal, de tal manera que se acerque lo máximo posible a las condiciones ideales.

La base de datos real consta de una secuencia de 4000 frames, en los cuales se realizan deformaciones que se parecen a simple vista a las entrenadas sintéticamente. Ae incluirán oclusiones y deformaciones a diferentes distancias y con diferentes incidencias de luz para probar la robustez ante los cambios de la misma. Unas imágenes ejemplo de esta base de datos se muestran en la tabla 5.2.

Tabla 5.2: Imágenes de la base de datos real

Tal como se observa anteriormente, la base de datos posee diferentes tipos de deformaciones en entorno real. Para realizar el rentrenamiento de dicha base de datos se emplearán una vez más las imágenes de

color reales cuyo fondo sera sustituido por fondos aleatorios de nuevo, quedando de la manera siguiente en la figura 5.6.


Figura 5.6: blender

### 5.1.2 Métricas de calidad

A continuación se definirán las métricas de calidad empleadas para evaluar los resultados del trabajo de fin de máster aquí presentado.

En este caso se empleará una única métrica, que estará representada por la raiz del error cuadrático medio o RMSE. Esto es debido a que este error nos permitirá representar directamente el error en unidades que se relacionarán directamente con medidas reales. La RMSE se expresa en la siguiente ecuación 5.1.

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n} \|y_{real} - y_{predicha}\|^2}, \tag{5.1}$$

donde $n$ representa el número de puntos mientras que $y_{real}$ y $y_{predicha}$ son el valor de GT y la predicción de la CNN respectivamente. La norma $\|x\|$ representa la norma $L_2$ del vector $x$.

Una vez definido el RMSE es necesario comentar que su empleo derivará en dos ramificaciones de error correspondientes a las diferentes salidas de la CNN propuesta:

- **Error de profundidad:** Se expresa mediante el RMSE directamente en unidades métricas, que en este caso serán **milimetros**.
- **Error de registro:** En el caso del error de registro, el mismo se expresará al igual que el anterior mediante RMSE, pero en unidades de imagen, es decir **píxeles**.

### 5.1.3 Estrategia y metodología de experimentación

A continuación se procederá a comentar la estrategia de evaluación que se seguirá. Dicho punto es importante dado que marca la calidad de la evaluación empleada y la rigurosidad de la misma.

La evaluación que se va a llevar a cabo se realizará por separado en cada base de datos y sobre los datos de test reservados para la misma. A continuación se define en mayor profundidad las condiciones de esta evaluación:

- **Base de datos sintética**: Se emplean cinco secuencias diferentes que abarcarán datos correspondientes a diferentes casos de deformaciones y a planos rígidos en movimiento. Estos datos son diferentes a los del set de entrenamiento. Dichos datos constituirán alrededor de un diez por ciento del total de la base de datos.
- **Base de datos real**: En la base de datos real y, al igual que en la sintética, se empleará un set de test que constituirá un diez por ciento de la base de datos real. A pesar de ser mucho más pequeña que la base de datos sintética, dicho set de test incluirá bastantes tipos de deformaciones diferentes que permitirán confirmar la generalización de la CNN.

Una vez definida la forma de proceder en cuanto a los datos de test, se procederá a hablar de los algoritmos rivales con los cuales se comparará la CNN aquí propuesta, de tal manera que ofrezca comparativas ricas en datos con respecto a algoritmos del estado del arte. Los ámbitos de comparación se detallan a continuación:

1. **Registro**: En este ámbito se utilizarán dos algoritmos del estado del arte con los que se comparará la CNN aqui propuesta:
   - **Dense Optical Flow with point trajectories o OFR**: Este es un método de OF denso con el que se obtienen trayectorias de puntos a través de una secuencia de video. Dicho método es un método robusto de registro de puntos que pertenece al estado del arte y cuyas características se describen en el artículo [5].
   - **Resnet50+Fully Connected o R50F**: Este metodo emplea una CNN residual tradicional, basada en la arquitectura ResNet [22] convolucional y capas densas, tal y como se ha descrito en el capítulo anterior. Esta arquitectura realizará tanto la reconstrucción como el registro, de tal manera que se pueda probar su eficacia en ambos ámbitos y compararla con la red propuesta en este trabajo.
2. **Reconstrucción tridimensional**: En este ámbito será posible encontrar tres algoritmos diferentes con los cuales se comparará la CNN aqui propuesta:
   - **Chhatkuli17 o CH17**: Este método de reconstrucción isométrica [55] se servirá de los datos de la plantilla plana en coordenadas métricas y de sus correspondencias en coordenadas de imagen, tanto en el frame de la plantilla como en el frame de la superficie deformada. Es necesario comentar que dicho metodo empleará una malla. Este metodo requiere el registro entre la imagen deformada y sin deformar mediante una función de warp diferenciable. Se evaluará este algoritmo con los datos de registro de GT, en el caso de imagen sintética, y con los datos expedidos por el método de OF denso en el caso de datos reales.
   - **Chhatkuli17+Refinado o CH17R**: Este método es similar al anterior, a excepción de la inclusión de una etapa de refinado basada en optimización de una función de coste.
   - **Resnet50+Fully Connected o R50F**: Este metodo emplea una CNN residual tradicional, basada en la arquitectura ResNet [22] convolucional y capas densas, tal y como se ha descrito en el capítulo anterior. Esta arquitectura realizará tanto la reconstrucción como el registro, de tal manera que se pueda probar su eficacia en ambos ámbitos y compararla con la red propuesta en este trabajo.

Por último, es necesario comentar que la información de GT para reconstrucción tridimensional sera expedida directamente por la Kinect v2 [6] en el caso de imagen real. Sin embargo, no se dispone de datos de registro, dada su alta dificultad técnica. En ese caso se empleará el metodo de OF [5] explicado

previamente como un baseline para comparar el registro de la CNN clásica y la CNN aqui propuesta. Cabe destacar que la red propuesta no necesita la secuencia de video para obtener el registro mientras que el método OF requiere que la plantilla y la imagen de entrada estén conectadas por dicha secuencia.

## 5.2 Resultados experimentales

### 5.2.1 Resultados de Reconstruccion 3D Densa

A continuación se va a proceder a presentar los resultados experimentales de la reconstrucción tridimensional. En los mismos se comenzará por las tablas con los resultados de la reconstrucción en la base de datos sintética, para posteriormente comentar las tablas de los resultados de la base de datos real.

#### 5.2.1.1 Evaluacion Base de datos sintetica

A continuacion se muestra la tabla de resultados de la reconstrucción sintética en un entorno rico en características 5.3.

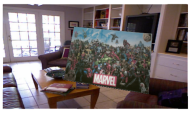
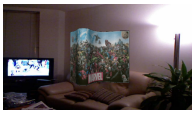
| Secuencias | Imagen | Descripción | Frames | CH17+GTR | CH17+OFR | CH17R+GTR | CH17R+OFR | R50F | CNNP |
|---|---|---|---|---|---|---|---|---|---|
| Seq1 |  | Poster retenido por un caño en el extremo derecho que sufre deformaciones por la fuerza del viento en diferentes direcciones | 1200 | 5.1098 | 8.5554 | 7.7093 | 8.6336 | 6.5566 | 1.8634 |
| Seq2 |  | Poster sin deformacion moviendose como un plano rigido | 600 | 1.6327 | 58.76 | 1.2396 | 59.72 | 12.25 | 1.6245 |
| Seq3 |  | Poster restringido en movimiento que sufre deformacion sobre todo en la zona central por causa de un objeto esferico que tira del mismo hacia fuera | 400 | 12.69 | 11.57 | 12.38 | 12.03 | 5.1403 | 1.4896 |
| Seq4 |  | Poster retenido por un caño en el extremo izquierdo que sufre deformaciones por la fuerza del viento en diferentes direcciones | 1200 | 8.1546 | 34.18 | 11.78 | 31.79 | 7.0634 | 1.5798 |

Tabla 5.3: Resultados experimentales sobre base de datos sintética, empleando como metrica RMSE(mm) en un entorno rico en características.

A continuación se muestra la tabla de resultados de la reconstrucción sintética en un entorno pobre en características 5.4.

| Secuencias | Imagen | Descripción | Frames | CH17+GTR | CH17+OFR | CH17R+GTR | CH17R+OFR | R50F | CNNP |
|---|---|---|---|---|---|---|---|---|---|
| Seq1 |  | Poster retenido por un caño en el extremo derecho que sufre deformaciones por la fuerza del viento en diferentes direcciones | 1200 | 5.1098 | 21.3298 | 7.7093 | 20.2110 | 6,61989 | 1,69958 |
| Seq2 |  | Poster sin deformacion moviendose como un plano rigido | 600 | 1.6327 | 16.4259 | 1.2396 | 14.5054 | 12,8265 | 1,8969 |
| Seq3 |  | Poster restringido en movimiento que sufre deformacion sobre todo en la zona central por causa de un objeto esferico que tira del mismo hacia fuera | 400 | 12.69 | 7.08 | 12.38 | 8.6062 | 5,5299 | 1,3057 |
| Seq4 |  | Poster retenido por un caño en el extremo izquierdo que sufre deformaciones por la fuerza del viento en diferentes direcciones | 1200 | 8.1546 | 17.60 | 11.78 | 18.35 | 6,9891 | 1,8506 |

Tabla 5.4: Resultados experimentales sobre base de datos sintética, empleando como metrica RMSE(mm) en un entorno pobre en características.

Tal y como se puede observar en las tablas 5.3 y 5.4, los algoritmos basados en CH17 [55], y en los casos en los que disponen de datos de registro perfectos, muestran errores que no superan el centimetro. Sin embargo, en los casos en los que el registro es suministrado por OF [5], el error de reconstrucción crece considerablemente, por encima de los dos centimetros en algunos casos. Otro hecho destacable es que el refinado isométrico empeora en la gran mayoria de los casos. Esto puede ser debido a que las deformaciones generadas por Blender no son puramente isométricas en todos los casos. En cuanto a los métodos basados en CNN, se puede observar como la red clásica tiene errores mayores o similares a los de CH17 [55], mientras que la red propuesta tiene el error mas bajo, que no supera los 2 milímetros.

#### 5.2.1.2 Evaluacion Base de datos real

A continuación se mostrarán los resultados reales de reconstrucción tridimensional. En este caso se empleará una secuencia de testeo que representará el diez por ciento de las secuencias reales. En la siguiente tabla 5.5 se mostrarán las comparativas entre los métodos clásicos y el método aqui propuesto.

| Secuencias | Imagen | Descripción | Frames | CH17+OFR | CH17R+OFR | R50F | CNNP |
|---|---|---|---|---|---|---|---|
| Seq1 |  | Poster retenido por un caño en el extremo derecho que sufre deformaciones por la fuerza del viento en diferentes direcciones | 211 | 38.1230 | 34.2527 | 17.5372 | 9.5185 |

Tabla 5.5: Resultados de reconstrucción 3D sobre la base de datos real.

En la comparativa anterior se puede observar la comparación entre los métodos del estado del arte [55] contra los métodos basados en redes convolucionales profundas; en este caso la arquitectura clásica [22] y la arquitectura aquí propuesta. Tal como se puede observar, los métodos del estado del arte rondan los 3,4 cm de error, mientras que la CNN clásica tiene 1,7 cm de error y la CNN propuesta tiene 0,94 cm de error. En este caso la CNN propuesta es mucho mejor que los métodos del estado del arte.

### 5.2.2 Resultados de Registro

A continuación se va a proceder a presentar los resultados experimentales del registro. En los mismos se comenzará por las tablas con los resultados del registro en la base de datos sintética, para posteriormente comentar las tablas de los resultados de la base de datos real.

#### 5.2.2.1 Evaluación Base de datos sintetica

A continuación se muestra la tabla de resultados de la reconstrucción sintética en un entorno rico en características 5.6.

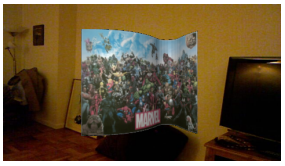
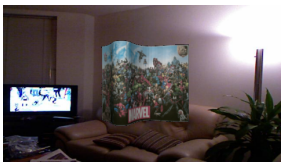
| Secuencias | Imagen | Descripción | Frames | OFR | R50F | CNNP |
|---|---|---|---|---|---|---|
| Seq1 |  | Poster retenido por un caño en el extremo derecho que sufre deformaciones por la fuerza del viento en diferentes direcciones | 1200 | 6.9774 | 5.7544 | 1.5626 |
| Seq2 |  | Poster sin deformacion moviendose como un plano rigido | 600 | 8.2314 | 7.7607 | 1.2295 |
| Seq3 |  | Poster restringido en movimiento que sufre deformacion sobre todo en la zona central por causa de un objeto esferico que tira del mismo hacia fuera | 400 | 1.9034 | 5.1290 | 1.0668 |
| Seq4 |  | Poster retenido por un caño en el extremo izquierdo que sufre deformaciones por la fuerza del viento en diferentes direcciones | 1200 | 6.548 | 5.8933 | 1.5356 |

Tabla 5.6: Resultados experimentales sobre base de datos sintetica con muchas características, empleando como metrica RMSE(px)

A continuación se muestra la tabla de resultados de la reconstrucción sintética en un entorno pobre en características 5.7.

| Secuencias | Imagen | Descripción | Frames | OFR | R50F | CNNP |
|---|---|---|---|---|---|---|
| Seq1 |  | Poster retenido por un caño en el extremo derecho que sufre deformaciones por la fuerza del viento en diferentes direcciones | 1200 | 7.6101 | 6.1630 | 2.1063 |
| Seq2 |  | Poster sin deformacion moviendose como un plano rigido | 600 | 2.6509 | 8.2290 | 1.8095 |
| Seq3 |  | Poster restringido en movimiento que sufre deformacion sobre todo en la zona central por causa de un objeto esferico que tira del mismo hacia fuera | 400 | 2.0850 | 6.3235 | 1.6288 |
| Seq4 |  | Poster retenido por un caño en el extremo izquierdo que sufre deformaciones por la fuerza del viento en diferentes direcciones | 1200 | 6.2140 | 6.0678 | 1.9591 |

Tabla 5.7: Resultados experimentales sobre base de datos sintetica con pocas características, empleando como metrica RMSE(px)

Tal y como se puede observar en las tablas 5.6 y 5.7, los algoritmos del estado del arte [5] tienen resultados de registro comparables o mejores que los proporcionados por la arquitectura clásica de CNN. Sin embargo, la arquitectura aquí propuesta obtiene mejores resultados que los métodos de flujo óptico en todos los caso. Es muy importante destacar de nuevo que los algoritmos de registro deformable basados en flujo óptico requieren secuencias de video y se basan en variaciones pequeñas entre los diferentes frames mientras que, en el caso de la red convolucional aquí propuesta, se realiza un registro denso con respecto a una plantilla, sin necesidad de secuencia de video. El error obtenido por la arquitectura propuesta no supera los 2 pixeles de RMSE.

#### 5.2.2.2 Evaluación Base de datos real

A continuación se mostrarán las comparaciones de registro con respecto al set de testeo real, que representa un diez por ciento de la base de datos. Es muy importante destacar que, dado que en este caso se carece de GT real para el registro, se empleará como algoritmo de baseline [5]. El baseline utilizado pierde muchos puntos a lo largo de la secuencia y, por tanto, únicamente se podrán evaluar los 100 primeros frames de testeo de los 210 frames posibles. La imagen 5.7 muestra la densidad de puntos cuyo seguimiento sobrevive hasta el frame 100 de la secuencia de test.

Figura 5.7: Imagen con puntos registrados correctamente a través de la deformación por [5]

En la tabla siguiente se observarán las comparativas entre la red CNN clásica y la aqui propuesta 5.8.

| Secuencias | Imagen | Descripción | Frames | R50F | CNNP |
|---|---|---|---|---|---|
| Seq1 |  | Poster retenido por un caño en el extremo derecho que sufre deformaciones por la fuerza del viento en diferentes direcciones | 100 | 5.0210 | 2.3201 |

Tabla 5.8: Resultados de registro sobre la base de datos real.

### 5.2.3 Temporización de algoritmos

En el apartado siguiente se aborda la temporización de cada uno de los algoritmos aquí planteados, y de aquellos con los cuales estos se van a comparar. El el gráfico de la figura 5.8 se expondrán cada una de las temporizaciones de los algoritmos en unidades de fps o frames por segundo.

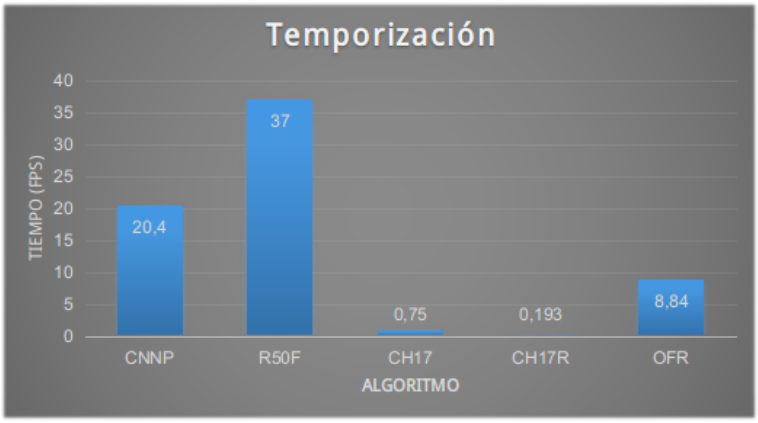



Figura 5.8: Temporizacion de los diferentes algoritmos empleados en este trabajo

En la tabla anterior se puede observar que los métodos del estado del arte no se acercan al tiempo real. Mientras tanto los métodos basados en CNN lo alcanzan o incluso lo superan. Es de destacar que R50F es más rápido a la hora de predecir que el método aquí propuesto CNNP, pero por contra sus errores tanto de registro como de reconstrucción son mucho mayores y no alcanza una buena generalización. Por lo cual se puede concluir que el método CNNP es mejor.

## 5.3 Conclusiones

Tras observar los resultados previamente mostrados se puede observar que CNNP es un método comparable con los existentes a día de hoy en el estado del arte. A continuación se realizará una pequeña comparativa en los diferentes ámbitos de comparación.

1. **Reconstrucción tridimensional:**Los resultados obtenidos por la misma igualan o superan a los de CH17 [55]. Esto puede ser debido a que la CNN no esta ligada al cien por cien a un modelo físico, como es el caso de este método. Por ello, especialmente en las deformaciones cuasisométricas de los experimentos aquí presentados, este método obtiene peores resultados pero consigue ajustarse aproximadamente. Por otro lado es de importancia destacar que CH17 [55] requiere un registro previo de los puntos de la malla a través de las imágenes. Una tarea que puede ser muy dificil en los casos reales, dado que presentan grandes casos de cambios de luz y perspectiva. En el caso de la arquitectura R50F, es posible observar que es mucho menos eficiente y no consigue generalizar las deformaciones, unicamente aproximar el plano de menor error. La arquitectura propuesta obtiene menores errores ya que se ajusta a dicha deformación en cierta medida.

2. **Registro:** En cuanto a los resultados de registro, el método propuesto no requiere una secuencia de video para hacer un registro con respecto a la plantilla. Mientras tanto, gran parte de los métodos del estado del arte requieren pequeñas variaciones entre frames para hacer el registro a través de una secuencia. A pesar de no requerir una secuencia de video, obtiene resultados de registro más precisos que los métodos basados en flujo óptico. Este es un resultado muy prometedor del método propuesto. Además, la arquitectura CNNP aqui propuesta muestra ser bastante mejor que la R50F clásica.

En definitiva, y a falta de un análisis experimental más extenso, se puede concluir que el método propuesto representa una solución competitiva con el estado del arte actual en SfT, solucionando paralelamente además el problema del registro deformable.

# Capítulo 6

# Conclusiones y líneas futuras

En este apartado se detallan y explican cada una de las conclusiones y lineas futuras sobre el trabajo de fin de master aquí realizado. Dentro de las lineas futuras se explican algunas líneas que posiblemente el día de la presentación de este trabajo se estén llevando a cabo.

## 6.1 Conclusiones

Una vez acabada la explicación y las comparativas del trabajo expuesto, es posible concluir que las redes convolucionales pueden ser una opción más en la técnica SfT. Esto abre una posible veta de investigación de cara al futuro, en el caso del contexto de este trabajo con fines de realidad aumentada. A continuación se puntualizan las posibles conclusiones más remarcables en cuanto a este trabajo.

- Reconstrucción y registro denso: Este trabajo de fin de master demuestra la posibilidad de realizar una reconstrucción 3D y a su vez la obtención del registro de dicha reconstrucción con respecto a una plantilla. Dado que dicho proceso se realiza mediante mapas densos de profundidad y seguimiento, se eliminan los problemas tradiccionales de la discretización de mallas, que resultaban en una problemática de procesado y aprovechamiento de la información espacial.
- Arquitectura Encoder-Decoder VS Encoder-Densas: A través de este proyecto se visualiza una comparación entre una estructura clásica bastante usada en el campo de las CNN y la estructura aquí propuesta. Esto se realiza así para probar que una estructura convencional no fue suficiente para realizar una reconstrucción lo suficientemente buena. En los resultados se puede observar como la estructura propuesta es mejor y más eficiente que la clásica.
- Ruptura del paradigma iterativo: En este trabajo se comparan las diferentes CNN con métodos iterativos clásicos. Pero, ¿Por que la CNN deberían ser mejor que dichos métodos?. El primer problema de dichos métodos es la reconstrucción discretizada de sólidos. Debido a la gran carga computacional que sufren tienen limitaciones de tamaño de malla de reconstrucción, lo cual se soluciona mediante dichas CNN. Esto permite que la solución sea un mapa denso sin limitación de tamaño, excepto el tamaño de la imagen de entrada. Por otro lado destacar que estas CNN funcionan en tiempo real, mientras que dichos métodos en su gran mayoría no funcionan en tiempo real, requieren un X de iteraciones bastante alto para alcanzar buenas reconstrucciones.
- Problemas de Hardware: Las CNN por contra suelen requerir bastante más hardware que cualquier otro método, aunque a día de hoy cada vez son más eficientes. De las redes propuestas aquí se propondrá una red convolucional que funcionará en tiempo real en GPU .

## 6.2 Líneas futuras

Por último se presentarán las lineas futuras propuestas para este trabajo de fin de master, dado que podrá ser ampliado y del mismo podrán partir ciertas lineas de investigación interesantes.

- Reconstrucción de sólidos elásticos: Este trabajo propone la reconstrucción de sólidos deformables pero cuasisómetricos. Por lo cual una posible linea de investigación interesante sería portar dichas reconstrucciones a sólidos elásticos deformables que se ajusten al modelo de equiárea, por ejemplo. Este es un modelo que se acerca más a los problemas reales, dado que casos como el de los organos del cuerpo humano se ajustan mejor a dicho modelo, o al menos la mayoría de ellos, como el riñon, por ejemplo.
- Reconstrucción de objetos volumétricos: Este trabajo ha sido realizado en el entorno de reconstrucción de objetos no volumétricos tales como pósters y folios, objetos cuyo modelo físico y comportamiento es más sencillo que el de otros tales como peluches, camisetas o incluso órganos.
- Empleo de funciones de coste nuevas: La función de coste empleada en este trabajo es una función genérica empleada para regresión, pero lo ideal sería emplear una función de pérdidas que incluyerá características del tipo de objeto, como restricciones físicas o características del sensor empleado como pueden ser las restricciones de proyección.
- Creación de nuevas capas propias: En este caso se han empleado capas genéricas de Keras/Tensorflow, pero sería posible crear capas propias del problema que se abarca en este trabajo. Estas posiblemente facilitarían el entrenamiento y mejorarían los resultados aquí obtenidos.

# Capítulo 7

# Presupuesto

## 7.1 Costes de equipamiento

### 7.1.1 Equipamiento hardware utilizado:

| Concepto | Cantidad | Coste unitario | Subtotal(euros) |
|---|---|---|---|
| Ordenador I7(3700Mhz)+NVIDIA GTX1080 | 1 | 2000 | 2000 |
| Camara Tof Kinect v2 | 1 | 176,71 | 176,71 |
| Coste Total | | | 2176,71 |

### 7.1.2 Recursos software utilizados:

| Concepto | Cantidad | Coste unitario | Subtotal(euros) |
|---|---|---|---|
| Ubuntu 14.04.1 LTS | 1 | 0 | 0 |
| Librerias Opencv Python 3.1.0 | 1 | 0 | 0 |
| Librerias Libfreenect 2 | 1 | 0 | 0 |
| Librerias Tensorflow 1.4.0 | 1 | 0 | 0 |
| Librerias Keras 2.1.0 | 1 | 0 | 0 |
| Matlab | 1 | 2000 | 2000 |
| Image Proccesing Toolbox for Matlab | 1 | 1000 | 1000 |
| Computer Vision Toolbox for Matlab | 1 | 1250 | 1250 |
| Image Adquisition Toolbox for Matlab | 1 | 1000 | 1000 |
| Texstudio | 1 | 0 | 0 |
| Cuda Libraries | 1 | 0 | 0 |
| Total | | | 5250 |

## 7.2 Costes Mano de obra

| Concepto | Cantidad | Coste unitario | Subtotal(euros) |
|---|---|---|---|
| Montaje y fijacion cenital de la cámara | 10 | 15 euros/hora | 150 |
| Programacion y Desarrollo de Software | 287 | 60euros/hora | 17220 |
| Mecanografiado de documentacion,manuales y tutoriales | 65 | 15 euros/hora | 975 |
| Total | | | 18345 |

## 7.3 Costes Totales

| Concepto | Subtotal |
|---|---|
| Hardware | 2176,71 |
| Software | 5250 |
| Mano de obra | 18345 |
| Total | 25771,7 |

El importe total del presupuesto asciende a la cantidad de: VEINTICINCOMIL SETECIENTOS SETENTA Y UN EUROS.

En Alcalá de Henares, a de septiembre de 2018. David Fuentes Jiménez ,Master en Ingeniería Industrial.

# Bibliografía


[1] S. Parashar, D. Pizarro, and A. Bartoli. Isometric non-rigid shape-from-motion in linear time. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4679–4687, June 2016.

[2] Nazim Haouchine, Jérémie Dequidt, Igor Peterlik, Erwan Kerrien, Marie-Odile Berger, and Stéphane Cotin. Image-guided Simulation of Heterogeneous Tissue Deformation For Augmented Reality during Hepatic Surgery. In *ISMAR - IEEE International Symposium on Mixed and Augmented Reality 2013*, Adelaide, Australia, October 2013.

[3] Bongjin Koo, Erol Ozgur, Bertrand Le Roy, Emmanuel Buc, and Adrien Bartoli. Deformable registration of a preoperative 3d liver volume to a laparoscopy image using contour and shading cues. In *Medical Image Computing and Computer Assisted Intervention - MICCAI 2017 - 20th International Conference, Quebec City, QC, Canada, September 11-13, 2017, Proceedings, Part I*, pages 326–334, 2017.

[4] Pauline Chauvet, Toby Collins, Clement Debize, Lorraine Novais-Gameiro, Bruno Pereira, Adrien Bartoli, Michel Canis, and Nicolas Bourdel. Augmented reality in a tumor resection model. *Surgical Endoscopy*, 32(3):1192–1201, Mar 2018.

[5] Narayanan Sundaram, Thomas Brox, and Kurt Keutzer. Dense point trajectories by gpu-accelerated large displacement optical flow. In *Proceedings of the 11th European Conference on Computer Vision: Part I*, ECCV'10, pages 438–451, Berlin, Heidelberg, 2010. Springer-Verlag.

[6] Microsoft. Kinect v2 product. https://www.microsoftstore.com.

[7] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, ISBN: 0521540518, second edition, 2004.

[8] Liu-Yin Qi, Rui Yu, Lourdes Agapito, Andrew W. Fitzgibbon, and Chris Russell. Better together: Joint reasoning for non-rigid 3d reconstruction with specularities and shading. *CoRR*, abs/1708.01654, 2017.

[9] Toby Collins Mathias Gallardo and Adrien Bartoli. Using shading and a 3d template to reconstruct complex surface deformations. In Edwin R. Hancock Richard C. Wilson and William A. P. Smith, editors, *Proceedings of the British Machine Vision Conference (BMVC)*, pages 38.1–38.12. BMVA Press, September 2016.

[10] Mathias Gallardo, Toby Collins, and Adrien Bartoli. Dense non-rigid structure-from-motion and shading with unknown albedos. In *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, pages 3904–3912, 2017.

[11] E. North Coleman and Ramesh Jain. Shape from shading for surfaces with texture and specularity. In *Proceedings of the 7th International Joint Conference on Artificial Intelligence - Volume 2*, IJCAI'81, pages 652–657, San Francisco, CA, USA, 1981. Morgan Kaufmann Publishers Inc.

[12] R. White and D. A. Forsyth. Combining cues: Shape from shading and texture. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 1809–1816, June 2006.

[13] L. Smith, D. Nevins, N. T. Dat, and P. Fua. Measuring the accuracy of softball impact simulations. *Sports Engineering*, 2016.

[14] T. Collins and A. Bartoli. [poster] realtime shape-from-template: System and applications. In *2015 IEEE International Symposium on Mixed and Augmented Reality*, pages 116–119, Sept 2015.

[15] L Maier-Hein, A Groch, A Bartoli, S Bodenstedt, G Boissonnat, P-L Chang, NT Clancy, DS Elson, S Haase, E Heim, J Hornegger, P Jannin, H Kenngott, T Kilgus, B Mueller-Stich, D Oladokun, S Roehl, Santos TR dos, H-P Schlemmer, A Seitel, S Speidel, M Wagner, and D Stoyanov. Comparative validation of single-shot optical techniques for laparoscopic 3-d surface reconstruction. *IEEE TRANSACTIONS ON MEDICAL IMAGING*, 33:1913–1930, 2014.

[16] Blender Online Community. *Blender - a 3D modelling and rendering package*. Blender Foundation, Blender Institute, Amsterdam.

[17] Odos. Odos starform tof camera. http://www.odos-imaging.com.

[18] Baxler. Baxler tof camera. http://www.baslerweb.com/.

[19] Microsoft. Kinect v1 structured light camera. http://123kinect.com.

[20] Orbec. Orbec structured light camera. https://orbbec3d.com.

[21] Asus xtion pro. https://www.asus.com/es/3D-Sensor/Xtion_PRO/. [Online].

[22] Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun. Deep residual learning for image recognition. *Arxiv*, arXiv:1512.03385, December 2015.

[23] Christian Szegedy Sergey Ioffe. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *Arxiv*, arXiv:1502.03167(6), February 2015.

[24] S. Kumar, Y. Dai, and H. Li. Multi-body non-rigid structure-from-motion. In *2016 Fourth International Conference on 3D Vision (3DV)*, volume 00, pages 148–156, Oct. 2016.

[25] Akhter, Ijaz, Sheikh, Yaser, Khan, Sohaib, and Kanade, Takeo. Nonrigid structure from motion in trajectory space. In *Neural Information Processing Systems*, volume 1, pages 41–48, 2008.

[26] Agudo, A. and Moreno-Noguer. Simultaneous pose and non-rigid shape with particle dynamics. *CVPR*, (1), 2015.

[27] Bartoli, A. and Collins, T. Template-based isometric deformable 3d reconstruction with sampling-based focal length self-calibration. *CVPR*, 2013.

[28] Y., Chadebecq, F. Bartoli, A., Gerard and Collins, T. On template-based reconstruction froma single view: Analytical solutions and proofs of well-posedness for developable, isometric and conformal surfaces. *CVPR*, 2012.

[29] Bartoli, A., GÃ©rard, Y., Chadebecq, F., Collins, T. and Pizarro, D. Shape-from-template.

[30] Bartoli and OzgÃ¼r. A perspective on non-isometric shape-from-template. *Ismar*, 2016.

[31] Bartoli, A., Pizarro, D. and Collins, T. A robust analytical solution to isometric shape- from-template with focal length calibration.

[32] Brunet, F., Bartoli, A. and Hartley, R. Monocular template-based 3d surface reconstruction: Convex inextensible and nonconvex isometric methods. *Computer Vision and Image Understand- ing*, 2014.

[33] Brunet, F., Bartoli, A. and Hartley, R. Monocular template-based 3d surface reconstruction: Convex inextensible and nonconvex isometric methods. *Computer Vision and Image Understand- ing*, 2014.

[34] Chhatkuli, A., Pizarro, D. and Bartoli, A. Stable template-based isometric 3d reconstruc- tion in all imaging conditions by linear least-squares. *CVPR*, 2014.

[35] Collins, T. and Bartoli, A. Realtime shape-from-template: System and applications. *Ismar*, 2015.

[36] Gallardo, M., Collins, T. and Bartoli, A. Using shading and a 3d template to reconstruct complex surface deformations. *Bmvc*, 2016.

[37] Malti, A., Hartley, R., Bartoli, A. and Kim, J.-H. Monocular template-based 3d reconstruc- tion of extensible surfaces with local linear elasticity. *CVPR*, 2013.

[38] Ngo,Ostlund and Fua. Template-based monocular 3d shape recovery using laplacian meshes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2016.

[39] Perriollat, M., Hartley, R. and Bartoli, A. Monocular template-based reconstruction of inextensible surfaces. *BMVC*, 2008.

[40] R., Russell, C., Campbell, N. D. F. Yu and Agapito, L. Direct, dense, and deformable: Template-based non-rigid 3d reconstruction from rgb video.

[41] C. Bregler, A. Hertzmann, and H. Biermann. Recovering non-rigid 3d shape from image streams. In *Proceedings IEEE Conference on Computer Vision and Pattern Recognition. CVPR 2000 (Cat. No.PR00662)*, volume 2, pages 690–696 vol.2, June 2000.

[42] Yuchao Dai. A simple prior-free method for non-rigid structure-from-motion factorization. In *Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, CVPR '12, pages 2018–2025, Washington, DC, USA, 2012. IEEE Computer Society.

[43] Ravi Garg, Anastasios Roussos, and Lourdes Agapito. Dense variational reconstruction of non-rigid surfaces from monocular video. In *CVPR*, pages 1272–1279. IEEE Computer Society, 2013.

[44] J. Taylor, A. D. Jepson, and K. N. Kutulakos. Non-rigid structure from locally-rigid motion. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 2761–2768, June 2010.

[45] Ajad Chhatkuli, Daniel Pizarro, and Adrien Bartoli. Non-rigid shape-from-motion for isometric surfaces using infinitesimal planarity. In *Proceedings of the British Machine Vision Conference*. BMVA Press, 2014.

[46] Antonio Agudo and Francesc Moreno-Noguer. Simultaneous pose and non-rigid shape with particle dynamics. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

[47] D. T. Ngo, J. Ostlund, and P. Fua. Template-based monocular 3d shape recovery using laplacian meshes. 38(1):172–187, Jan. 2016.

[48] M. Perriollat, R. Hartley, and A.E. Bartoli. Monocular template-based reconstruction of inextensible surfaces. In *Proc. BMVC*, pages 60.1–60.10, 2008. doi:10.5244/C.22.60.

[49] Florent Brunet, Adrien Bartoli, and Richard I. Hartley. Monocular template-based 3d surface reconstruction: Convex inextensible and nonconvex isometric methods. *Computer Vision and Image Understanding*, 125:138–154, 2014.

[50] Toby Collins and Adrien Bartoli. Realtime shape-from-template: System and applications. In *ISMAR*, pages 116–119. IEEE Computer Society, 2015.

[51] Mathieu Salzmann and Pascal Fua. Linear local models for monocular reconstruction of deformable surfaces. *IEEE Trans. Pattern Anal. Mach. Intell.*, 33(5):931–944, 2011.

[52] A. Malti, A. Bartoli, and T. Collins. A pixel-based approach to template-based monocular 3d reconstruction of deformable surfaces. In *2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops)(ICCVW)*, volume 00, pages 1650–1657, Nov. 2012.

[53] Dat Tien Ngo, Sanghyuk Park, Anne Jorstad, Alberto Crivellaro, Chang Yoo, and Pascal Fua. Handling occlusions and sparse textures in a deformable surface tracking framework. *CoRR*, abs/1503.03429, 2015.

[54] Rui Yu, Chris Russell, Neill D.F. Campbell, and Lourdes Agapito. Direct, dense, and deformable: Template-based non-rigid 3d reconstruction from rgb video. 2015.

[55] Ajad Chhatkuli, Daniel Pizarro and Adrien Bartoli. Stable template-based isometric 3d reconstructionin all imaging conditions by linear least-squares. *ALCoV-ISIT*, (1), 2017.

[56] A.Pumarola,A.Agudo,I.Pozi,A.Sanfeliu,V.Lepetit,F.Moreno-Noguer. Geometry-aware network for non-rigid shape prediction from a single view. *iri.upc.edu*.

[57] Wei, Shih-En, Ramakrishna, Varun, Kanade, Takeo, and Sheikh, Yaser. Convolutional pose machines. In *CVPR*, pages 4724–4732. IEEE Computer Society, 2016.

[58] Julien Pilet, Vincent Lepetit, and Pascal Fua. Fast non-rigid surface detection, registration and realistic augmentation. *International Journal of Computer Vision*, 76(2):109–122, Feb 2008.

[59] Daniel Pizarro and Adrien Bartoli. Feature-based deformable surface detection with self-occlusion reasoning. *International Journal of Computer Vision*, 97(1):54–70, Mar 2012.

[60] David G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60:91–110, 2004.

[61] Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool. Speeded-up robust features (surf). *Comput. Vis. Image Underst.*, 110(3):346–359, June 2008.

[62] Pablo Fern$#225;ndez Alcantarilla, Adrien Bartoli, and Andrew J. Davison. Kaze features. In *Proceedings of the 12th European Conference on Computer Vision - Volume Part VI*, ECCV'12, pages 214–227, Berlin, Heidelberg, 2012. Springer-Verlag.

[63] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary R. Bradski. Orb: An efficient alternative to sift or surf. In Dimitris N. Metaxas, Long Quan, Alberto Sanfeliu, and Luc J. Van Gool, editors, *ICCV*, pages 2564–2571. IEEE Computer Society, 2011.

[64] T. Brox and J. Malik. Large displacement optical flow: Descriptor matching in variational motion estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(3):500–513, March 2011.

[65] Yoshua Benghio Xavier Glorot, Antoine Bordes. Deep sparse rectifier neural networks. *Procedings MLR*.

[66] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15:1929–1958, 2014.

[67] Leon Botou. Large-scale machine learning with stochastic gradient descent. *Compstat*, 2010.

[68] Yolan Singer John Duchi, Elad Hazam. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 11, 2011.

[69] Geoffrey Hinton. Overview of minibatch gradient descent.

[70] Matthew D. Zeiler. ADADELTA: an adaptive learning rate method. *CoRR*, abs/1212.5701, 2012.

[71] Jimmy Ba Diederik P. Kingma. Adam: A method for stochastic optimization. *Arxiv*, arXiv:1412.6980(6), December 2014.

[72] Timothy Dozat. Incorporating nesterov momentum into adam. *stanford*.

[73] Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. Cifar-10.

[74] Yann LeCun and Corinna Cortes. MNIST handwritten digit database. 2010.

[75] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. pages 1097–1105, 2012.

[76] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014.

[77] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Computer Vision and Pattern Recognition (CVPR)*, 2015.

[78] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. *CoRR*, abs/1512.00567, 2015.

[79] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alex A. Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. In *ICLR 2016 Workshop*, 2016.

[80] Francois Chollet. Xception: Deep learning with depthwise separable convolutions. *Arxiv*, arXiv:1610.02357, October 2016.

[81] Md Amirul Islam, Mrigank Rochan, Neil D. B. Bruce, and Yang Wang. Gated feedback refinement network for dense image labeling. In *Computer Vision and Pattern Recognition (CVPR)*, 2017.

[82] Guosheng Lin, Anton Milan, Chunhua Shen, and Ian D. Reid. Refinenet: Multi-path refinement networks for high-resolution semantic segmentation. *CoRR*, abs/1611.06612, 2016.

[83] David Eigen and Christian Puhrsch andRob Fergus. Depth map prediction from a single image using a multiscale deepnetwork. *CoRR*, 2014.

[84] Nitish Shirish Keskar and Richard Socher. Improving generalization performance by switching from adam to SGD. *CoRR*, abs/1712.07628, 2017.

[85] Pushmeet Kohli Nathan Silberman, Derek Hoiem and Rob Fergus. Indoor segmentation and support inference from rgbd images. In *ECCV*, 2012.

# Apéndice A

# Herramientas y recursos

Para poder un correcto funcionamiento del sistema descrito en este trabajo es necesario cumplir unos requisitos de hardware y software mínimos en los equipos que se van a utilizar

## A.1 Requisitos de Hardware

- 16 Gb de Ram DDR2 a 800Mhz o superior
- Procesador de 64 bits OctaCore 3 Ghz o superior
- Al menos 15 Gb de memoria libre en el disco duro o superior para la instalación de las librerías requeridas, Matlab y sus respectivas Toolbox

## A.2 Requisitos de Software

- Sistema operativo Ubuntu 14.04
- Pycharm Comunity
- Keras 2.1.2
- Tensorflow 1.4.0 [35]
- Python 3.5
- Matlab 2018
- Procesador de Latex