Interpolación temporal para imágenes de satélite

Enrique Fernández Morales

Resumen– En los últimos años, las redes neuronales se han convertido en una herramienta muy versátil y eficaz en la resolución problemas, en especial los relacionados con imágenes. Es por ello que en este proyecto pretendemos experimentar con esta tecnología, implementando modelos capaces de realizar motion interpolation en imágenes de satélites de más de 3 canales. Hemos utilizado cuatro modelos: dos ya desarrollados, DAIN y Super-SlowMo, y dos desarrollados por nosotros, una U-Net y un método de interpolación lineal. Los resultados de todos los modelos utilizados, aun siendo algunos menos precisos que otros, han sido muy satisfactorios.

Palabras clave– Interpolación de movimiento, Aprendizaje profundo, Aprendizaje automático, Imágenes de satélite, Red neuronal, Banda espectral

Abstract– In recent years, neural networks have become a very versatile and effective tool for solving problems, especially those related to images. That is why in this project we intend to experiment with this technology, implementing models capable of performing motion interpolation in satellite images of more than 3 channels. We have used four models; two already developed, DAIN and Super-SlowMo, and two developed by us, a U-Net and a linear interpolation method. The results of all the models used, although some were less accurate than others, were very satisfactory.

Keywords– Motion interpolation, Deep learning, Machine learning, Remote sensing, Neural network, Spectral band

1 Introducción

Actualmente muchos proyectos están comenzando a sustituir sus algoritmos clásicos por algoritmos de deep learning [3], esto se debe a que, en la mayoría de casos, son mucho más eficientes que los métodos tradicionales.

La interpolación de fotogramas [1] es una de las tareas más complejas del campo de visión por ordenador. El objetivo de la interpolación de fotogramas es aumentar el número de frames de una secuencia de vídeo para hacerla más atractiva visualmente. Para llevar a cabo esta tarea, las redes neuronales convolucionales son ideales, por lo que actualmente existen muchos trabajos relacionados con el procesamiento de imágenes o vídeos que utilizan estos algoritmos. La mayoría de estas redes están limitadas al trabajar con imágenes de 3 canales.

Uno de los problemas que nos encontramos en estos métodos de deep learning para realizar motion interpolation es que las redes están entrenadas con vídeos donde el framerate es pequeño (milisegundos). Esto significa que la diferencia no es tan grande entre un frame y el siguiente, por lo que la interpolación será más precisa. En el caso de aplicarlo a imágenes por satélite esto no siempre se cumple, ya que el framerate es, en muchos casos, de días. Existen casos excepcionales de satélites que hacen capturas más rápidamente. Como por ejemplo el MeteoSat [6], aunque la resolución de las imágenes es menor y el rango de tiempo sigue siendo muy grande (15 min) en comparación con vídeos convencionales.

En este proyecto proponemos un estudio de algoritmos de motion interpolation aplicado a imágenes satélite con diferentes redes neuronales. Para ello, hemos adaptado dos proyectos de interpolación de frames ya existentes, DAIN (Depth-Aware Video Frame Interpolation) [7] y Super-SlowMo [15], y desarrollado otras dos soluciones en base a una U-Net [16] y un algoritmo de interpolación simple.

• E-mail de contacte: enrique.fernandezmor@e-campus.uab.cat
• Menció realitzada: Computació
• Treball tutoritzat per: Felipe Lumbreras Ruiz (Ciències de la Computació)
• Curs 2020/21

2 OBJETIVOS

El objetivo principal de este proyecto es aumentar el framerate utilizando motion interpolation aplicado a imágenes de satélite con más de 3 canales. En la siguiente lista hemos definido una serie de objetivos generales, los cuales se deben cumplir para poder completar el objetivo principal:

- Investigar los diferentes datasets de remote sensing, sobre todo datasets de imagen por satélite.
- Repasar la materia de deep learning y redes convolucionales.
    - Hacer pruebas con redes neuronales para aprender más sobre su funcionamiento.
    - Encontrar proyectos relacionados con deep learning y aprender formas óptimas de usarlas.
- Buscar información sobre motion interpolation e investigar proyectos relacionados y diferentes algoritmos para llevarlo a cabo.
- Trabajar con una red neuronal que haga motion interpolation con deep learning.
    - Hacer pruebas con la red usando imágenes con 3 canales.
    - Implementar una modificación en el algoritmo con el objetivo de trabajar con imágenes con más de 3 canales.
- Testear y validar la solución.
    - Aumentar la precisión de la solución entrenando desde cero la solución o haciendo fine-tuning [4].

3 ESTADO DEL ARTE

Como ha sido explicado en la introducción, hemos seleccionado varios modelos para realizar motion interpolation con deep learning. Estos algoritmos generalmente se utilizan para interpolar vídeos genéricos y están limitados a frames RGB. Por esta razón, estos algoritmos no están pensados para ser usados con imágenes de satélite, ya que estas suelen estar compuestas por más de 3 canales. Aun así, existen proyectos que están centrados en motion interpolation [1] de imágenes satélite (remote sensing) [3]. Tenemos por ejemplo proyectos que utilizan esta técnica para estimar el movimiento de los glaciares [10], [11]. Aunque estos proyectos no utilizan técnicas de deep learning. Además, hemos encontrado trabajos como el siguiente tutorial técnico [17] donde se explica y se trabaja con deep learning y remote sensing. Es por esto que consideramos que este proyecto llena un vacío y puede resultar útil para las personas que necesiten hacer motion interpolation de imágenes de satélite. Varios de los modelos que hemos seleccionado ya están desarrollados y utilizan deep learning. Estos son DAIN y Super-SloMo.

El proyecto DAIN (Depth-Aware Video Frame Interpolation) [7] cumple con el propósito de aumentar los fps con motion interpolation en cualquier vídeo usando deep learning. Los desarrolladores proponen un modelo de interpolación de fotogramas con reconocimiento de profundidad (DAIN), con la intención de detectar explícitamente las oclusiones explorando la señal de profundidad. Desarrollan una capa que proyecta el flujo óptico teniendo en cuenta la profundidad de los supuestos objetos y entorno del frame, para sintetizar flujos intermedios que muestran qué objetos están más cercas y cuáles están más lejos. Su método logra un rendimiento excepcional en conjuntos de datos específicos para motion interpolation, como por ejemplo el de Middlebury [8].

Super-SlowMo [15] es un algoritmo que propone interpolar frames utilizando una U-Net [16] en su estructura. Esta propuesta, a diferencia de la mayoría de modelos de interpolación de frames, se centra en una interpolación multi-frame, es decir, está centrada en predecir varios frames entre dos frames consecutivos. Utilizaremos este modelo para hacer un estudio de diferentes modelos de interpolación. Esta red utiliza un vídeo de entrada con 3 canales. La solución divide el vídeo en los diferentes frames que lo componen y entonces se utilizan para interpolar. En DAIN el vídeo se convertía a frames antes de ejecutar el algoritmo, pero en este modelo convierte el vídeo en frames dentro del mismo algoritmo.

También implementamos un modelo U-Net modificado para poder llevar a cabo el objetivo. En este modelo lo que hacemos es pasarle al input los dos frames concatenados, es decir, si cada frame tiene 11 canales el input tendrá 22 canales, ambos frames concatenados. Para finalizar, también utilizaremos un algoritmo de interpolación simple. Esta última utilizará el promedio de los dos frames interpolados para calcular el resultado.

4 METODOLOGÍA

En primer lugar, hemos estudiado la viabilidad del proyecto investigando sobre el actual estado del arte, buscando y estudiando proyectos que pudieran sernos útiles. Una vez investigado y comprobado si es viable, procedemos a definir los objetivos y con la ayuda de un diagrama de Gantt lo realizaremos de una forma más clara y ágil.

Cada semana, de lunes a viernes, se dedicaron como mínimo 4 horas diarias a avanzar en el proyecto, además cada martes a las 16:30 se realizaba una reunión con el tutor para discutir los avances, aclarar las dudas y definir los objetivos de esa semana.

Para llevar un seguimiento de la evolución hemos usado un método de ingeniería del software de tipo Agile, concretamente la herramienta Trello [9]. Trello es una herramienta tipo kanban que se caracteriza por plasmar el flujo de trabajo de forma visual, limitar la cantidad de trabajo en proceso, realizar un seguimiento del tiempo por tarea y una lectura sencilla con indicadores visuales. Gracias a estos puntos la identificación de los cuellos de botella puede ser más sencilla.

5 DESARROLLO

5.1. Datos utilizados

Para poder realizar las diferentes pruebas y experimentos necesitamos proveernos de imágenes de satélite. Hemos utilizado el banco de imágenes High Rate SEVIRI Level 1.5 Image Data - MSG - 0 degree [13]. Los datos de imagen de nivel 1.5 corresponden con los datos de imagen geolocalizados y radiométricamente preprocesados, listos para su posterior procesamiento. En este nivel, todos los efectos que puedan afectar a las imágenes, específicos del propio satélite, han sido eliminados en todos los canales. Se ha incluido información de control de calidad tanto radiométrica como geométrica. Además, el satélite SEVIRI hace una captura cada 15 minutos de imágenes con 12 canales (1 canal HR, 11 canales LR), como podemos observar en la tabla (1).

Channel	Bands	Centre Wavel.	Spectral Band (99% energy limits)	Dynamic Range	Operating Temp.	Detectors per channel	Sample distance at SSP
		µm	µm		°K		km
HRV	Visible &	(0.75)	broadband (peak within 0.6 - 0.9)	0 - 459 W/m^2 sr µm (scaled at centre frequency)	300	9	1
VIS0.6		0.635	0.56 - 0.71	0 - 533 W/m^2 sr µm		3	3
VIS0.8	Near	0.81	0.74 - 0.88	0 - 357 W/m^2 sr µm			
IR1.6	IR	1.64	1.50 - 1.78	0 - 75 W/m^2 sr µm			
IR3.9		3.92	3.48 - 4.36 (98% energy limits)	0 - 335 K	85-95		
IR8.7	Window	8.70	8.30 - 9.10 (98% energy limits)	0 - 300 K			
IR10.8		10.80	9.80 - 11.80 (98% energy limits)	0 - 335 K			
IR12.0		12.00	11.00 - 13.00 (98% energy limits)	0 - 335 K			
IR6.2	Water Vapour	6.25	5.35 - 7.15	0 - 300 K			
IR7.3		7.35	6.85 - 7.85 (98% energy limits)	0 - 300 K			
IR9.7	Ozone	9.66	9.38 - 9.94	0 - 310 K			
IR13.4	Carbon-dioxide	13.40	12.40 - 14.40 (96% energy limits)	0 - 300 K			

Fig. 1: Detalle de los canales espectrales de MSG SEVIRI.

Hemos elegido estas imágenes por su alto framerate comparado con otros tipos de imágenes de satélite en la que las transiciones pueden ser más caóticas.

5.2. Generación del dataset

Una de las características de este banco de datos es que es de fácil acceso, al poder seleccionar el área a descargar del mapa y el rango de tiempo deseado. Hemos seleccionado toda el área de Europa para la fecha de 07/05/2019 de 00:00 a 23:59 (96 frames, 1 frame cada 15 minutos). La composición del banco de datos es la siguiente: por cada captura hay 12 imágenes monocromáticas, cada una representa un canal. En la figura (2) podemos ver un frame de ejemplo en el canal VIS 0.8.

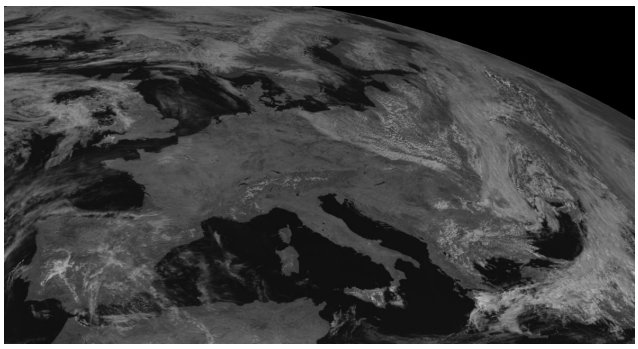

Fig. 2: Canal VIS 0.8 de la fecha 07/05/2020 a las 12:42.

Con el banco de datos seleccionado, ahora generamos el dataset. Dependiendo de la tarea para la cual el dataset estará destinado, el proceso de generarlo es distinto. Tenemos dos tipos de tarea para los datasets:

5.2.1. Dataset para testear

Este dataset lo generamos para hacer las diferentes pruebas.

1. Seleccionamos los canales que queremos utilizar en el orden deseado.
2. El algoritmo junta las diferentes imágenes monocromáticas que representan los canales seleccionados formando una imagen con el número de canales resultante. En el caso de haber seleccionado 3 canales, genera un vídeo.
3. El orden de las imágenes generadas es siempre el mismo, ordenadas cronológicamente.

5.2.2. Dataset para entrenar

Este dataset lo generamos para entrenar los diferentes modelos, utilizando data augmentation [14].

1. Elegimos el número de iteraciones, junto a los canales que queremos utilizar en el orden deseado en cada iteración. Los canales pueden ser distintos pero el número de canales debe ser igual en cada iteración.
2. El algoritmo junta las diferentes imágenes monocromáticas que representan a los canales seleccionados formando una imagen con el número de canales resultante.
3. El algoritmo recorta un área de tamaño 572×572 píxeles en una posición aleatoria, haciendo de esta manera data augmentation. Este proceso se ejecuta de 3 en 3 imágenes, teniendo como resultado 2 frames a interpolar (input) y uno central que hará las veces de ground-truth (output deseado).

5.3. Modelos utilizados

Hemos utilizado diferentes modelos para poder realizar un estudio comparando los resultados de cada uno de ellos. En todos los casos, salvo en la interpolación simple, hemos utilizado Google Colab [12]. Esta herramienta nos permite aprovechar las ventajas que proporciona la consola de Linux, sobre todo para la instalación de dependencias y la realización de los builds. También nos permite ahorrarnos los problemas que pueden conllevar las incompatibilidades entre librerías, drivers y plataformas. Los modelos utilizados son los siguientes:

5.3.1. Interpolación simple

En este tipo de motion interpolation no utilizamos redes neuronales ni algoritmos complejos, simplemente el frame interpolado se calculará como el promedio de los dos frames que se estén interpolando. Este sencillo método

nos permite usarlo como comparación base respecto de los otros modelos más complejos y completos.

5.3.2. DAIN

Como hemos visto en el apartado de estado de arte (sección 3), en el proyecto DAIN (Depth-Aware Video Frame Interpolation) [7] cumplen con el propósito de aumentar los fps con motion interpolation en cualquier vídeo usando deep learning. El funcionamiento es el siguiente: dados dos frames de entrada, DAIN primero estima los flujos ópticos y los mapas de profundidad, para utilizar la capa de depth-aware flow projection para generar flujos intermedios.

Luego adopta la capa de deformación adaptable para deformar los frames de entrada, los mapas de profundidad y los rasgos contextuales basados en los flujos y en los núcleos de interpolación con variación espacial. Finalmente, aplica una frame synthesis network para generar el frame de salida. Podemos observar esta arquitectura en la figura (A.3) del anexo.

Una de las limitaciones de la red de DAIN es que solo puede trabajar con imágenes de 3 canales.

Primero comprobamos que funciona correctamente usando varios vídeos de prueba. Los vídeos usados fueron vídeos normales de internet y vídeos de imagen satélite (usando solo 3 de sus múltiples canales). En este caso DAIN funciona correctamente.

Después, procedimos a probar si el modelo entrenaba sin problemas desde Google Colab. El dataset utilizado para esta prueba está compuesto únicamente por 2 imágenes, de esta forma podemos agilizar esta pequeña prueba. Los resultados fueron prometedores. No solo el entrenamiento funciona bien en Google Colab, sino que podemos elegir si entrenar desde cero o hacer fine-tuning.

5.3.3. Super-SloMo

Como hemos visto en el apartado de estado de arte (sección 3), esta solución se compone de dos partes. La primera parte con la computación del flujo y la segunda parte con la interpolación temporal del flujo. En la primera parte, se le pasa dos imágenes a una U-Net modificada para estimar el flujo óptico bidireccional, que luego se fusiona linealmente para aproximar el flujo óptico intermedio requerido para deformar las imágenes de entrada. Esta aproximación funciona bien en regiones lisas, pero mal alrededor de los límites del movimiento. Por lo tanto, luego utiliza otra U-Net modificada para refinar las aproximaciones de flujo y predecir los mapas de visibilidad suave. Aplicando los mapas de visibilidad a las imágenes deformadas antes de la fusión, se consigue reducir los fallos visuales. Además, este algoritmo es capaz de general tantas interpolaciones temporales como se desee, y no solo una interpolación por cada dos imágenes, dando así la posibilidad de generar más frames que en otros modelos. Podemos observar esta arquitectura en la figura (3).

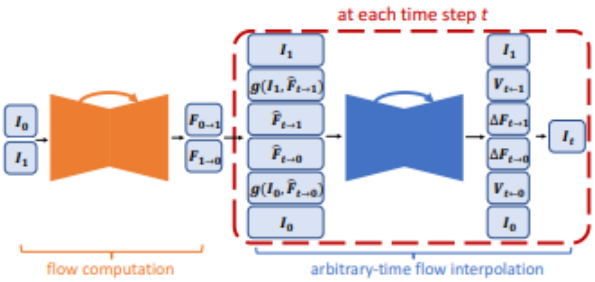

Fig. 3: Arquitectura del modelo Super-SloMo.

Al igual que con DAIN, primero comprobamos que funciona correctamente usando varios vídeos de prueba. En este caso, Super-SloMo funciona correctamente. También comparte la limitación de la red de DAIN, ya que Super-SlowMo solo puede trabajar con imágenes de 3 canales.

5.3.4. U-Net

Con un modelo U-Net [16] implementado desde cero podríamos, a diferencia de DAIN o Super-SlowMo, hacer motion interpolation de frames con más de 3 canales de forma directa. Es decir, utilizar directamente frames con 9 canales tanto para testear como para entrenar el modelo. Nos centramos con 9 canales para poder comparar fácilmente con resultados acumulados de 3 canales.

Nuestra U-Net es una réplica del paper original en PyTorch, aunque con algunos cambios. En nuestra versión de la red, el input está compuesto por los dos frames concatenados, por lo que, si utilizáramos dos frames RGB con las dimensiones $572\times572\times3$, el input sería $572\times572\times6$. La salida también esta modificada, ahora devolverá una imagen más pequeña, pero con los mismos canales de los frames individuales originales. Aunque U-Net es una arquitectura flexible y muy ligera, también tiene sus inconvenientes. El frame resultante acaba teniendo un tamaño menor. El caso del ejemplo anterior, que debería devolver una dimensión de $572\times572\times3$, acaba siendo una dimensión de $388\times388\times3$, por los sucesivos paddings.

U-Net está dividida en dos partes, el encoder y el decoder. La primera parte consiste en la aplicación repetida de dos convoluciones de 3×3, cada una seguida de una activación de tipo ReLU y una operación de max pooling de 2×2 con stride 2 para la reducción de la muestra. En cada paso duplicamos el número de canales. Hacemos esto 5 veces, con una excepción. La quinta vez no ejecutamos la operación de max pooling. Ahora entramos en la siguiente parte, el decoder. Cada bloque en este paso consiste en un aumento de la muestra del mapa de características seguido de una convolución 2×2 (“up-convolution”) que reduce a la mitad el número de canales de características, una concatenación con el correspondiente mapa de características recortado del camino del encoder, y dos convoluciones 3×3, cada una seguida de una ReLU. En la última capa se devuelve una imagen $388\times388\times9$. Podemos observar esta arquitectura en la figura (4).

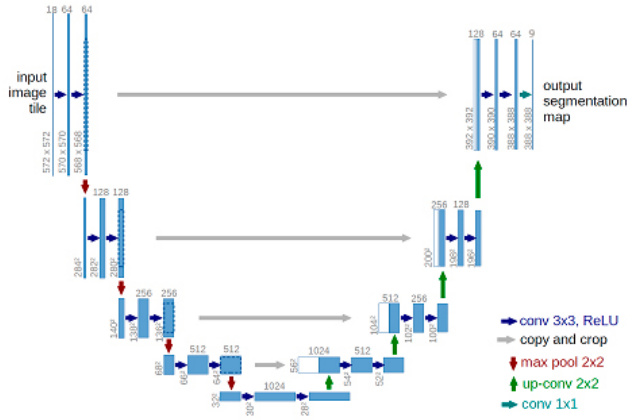

Fig. 4: Arquitectura del modelo U-Net.

Al igual que con el resto de modelos, primero comprobamos que funciona correctamente usando varios vídeos de prueba. En este caso, esta U-Net es completamente funcional, incluso es posible entrenarla y hacerle fine-tunning, y cambiando las capas inicial y final adaptamos el número de canales que necesitamos. A diferencia de los otros modelos, esta solución no tiene la limitación de los canales.

6 Experimentación y resultados

En este apartado explicamos qué métrica hemos seleccionado para evaluar las diferentes pruebas. A continuación, exponemos y explicamos los diferentes resultados obtenidos en las distintas pruebas realizadas.

6.1. Métrica seleccionada

Como métrica para medir la precisión hemos seleccionado el MSE. Además, hemos desarrollado un algoritmo que te muestra tanto el MSE como el SSIM entre una imagen original y una generada. Finalmente, se muestra de forma gráfica las zonas de discrepancia, en la cual ambas imágenes no son iguales. Podemos ver un ejemplo en la figura (5).

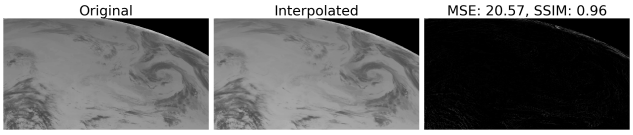

Fig. 5: Ejemplo en una interpolación de DAIN.

6.2. Experimentación de distancia temporal

El objetivo de este experimento es estudiar cómo reaccionan los distintos modelos en dos pruebas en igualdad de condiciones. Alterando únicamente la distancia temporal entre muestras. La primera prueba utiliza frames consecutivos (separación entre captura de frames de 30 minutos) y la segunda prueba utiliza frames más separados (separación entre captura de frames de 120 minutos).

Los modelos DAIN y Super-SlowMo están limitados a 3 canales, por lo que, para estar en igualdad de condiciones, en este experimento utilizaremos imágenes con 3 canales.

6.2.1. Preparación

El dataset generado para realizar las pruebas contiene combinaciones de los diferentes canales disponibles repartidos en las diferentes imágenes. Todas las combinaciones están compuestas por 3 canales, con una dimensión de $572 \times 572 \times 3$. Elegiremos los canales IR 3.9, IR 13.4 y VIS 0.6 para el experimento en ese orden. En el apartado (5.1) se encuentra la información específica de estos canales.

Como ya se ha comentado en el apartado (5.2.2), el dataset generado para entrenar contiene combinaciones de los diferentes canales disponibles repartidos en las diferentes imágenes, otorgando a las redes más flexibilidad a la hora de tratar con diferentes combinaciones de canales. Todas las combinaciones están compuestas por 3 canales, con una dimensión de $572 \times 572 \times 3$. En la figura (6) podemos observar un ejemplo de datos usados en el entrenamiento.

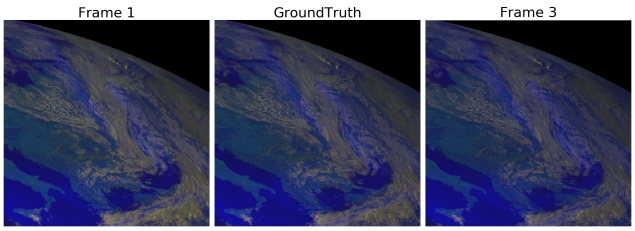

Fig. 6: Ejemplo de frames utilizados en el entrenamiento.

Para el caso de DAIN utilizaremos su versión preentrenada y su versión con fine-tunning realizado con un dataset generado por nosotros. Los valores de los parámetros utilizados en los entrenamientos son: Learning Rate: 0.0005, Weight Decay: 0.0001, Epochs: 100, Batch: 2, Número de Muestras (train+test): 286, Porcentaje train: 60.0 %. Para el caso de Super-SlowMo utilizaremos únicamente su versión preentrenada. Para el caso de la U-Net, utilizaremos la versión entrenada desde cero con un dataset generado por nosotros y con los valores de los parámetros de entrenamiento siguientes: Learning Rate: 0.0001, Weight Decay: 0.0001, Epochs: 5, Batch: 3, Número de Muestras (train+test): 286, Porcentaje train: 60.0 %. Para el caso de la interpolación simple no es necesario ninguna preparación, ya que no es necesario ningún entrenamiento.

6.2.2. Procedimiento

Como indicamos en el apartado (5.2), el satélite captura un frame cada 15 minutos. Para la primera prueba con interpolaciones de frames consecutivos hemos realizado las interpolaciones con frames capturados cada 30 minutos. Por ejemplo, si interpolamos el frame 1 (minuto 0) con el frame 3 (minuto 30), el resultado lo compararemos con el groundtruth frame 2 (minuto 15). Teniendo esto en cuenta, realizaremos la media de los resultados utilizando el banco de datos de SEVIRI anteriormente mencionado (5.2), generado con imágenes de fecha 07/05/20 de las 00:00 a

las 23:59 (96 frames).

Para la segunda prueba con observaciones más separadas hemos realizado las interpolaciones con frames capturados cada 120 minutos. Por ejemplo, si interpolamos el frame 1 (minuto 0) con el frame 9 (minuto 120), el resultado lo compararemos con el groundtruth frame 5 (minuto 60). Teniendo esto en cuenta, realizaremos la media de los resultados utilizando el banco de datos de SEVIRI, igual que en el caso anterior.

La comparación se tiene que realizar en igualdad de condiciones. El modelo U-Net propio tiene la limitación de generar un output más pequeño que el input. Por esa razón, a la hora de calcular el MSE del output de cada modelo, realizamos un recorte a 388×388 píxeles para igualar todos los modelos a la limitación de la U-Net.

6.2.3. Resultados

Una vez realizada las pruebas tenemos los siguientes resultados:

TABLA 1: RESULTADOS DE LA EXPERIMENTACIÓN DE DISTANCIA TEMPORAL.

	MSE distancia temporal Medias	
	Prueba 1	Prueba 2
DAIN preentrenada	3.83	106.59
DAIN fine-tuning	3.83	106.59
Super-SloMo	9.71	72.94
U-Net propia	118.55	122.44
Interpolación simple	7.55	149.23

Tenemos los resultados de la experimentación de la primera prueba y la segunda prueba en la tabla (1).
En la figura (7) tenemos un ejemplo de los resultados para cada modelo en 3 casos distintos. En la figura (A.1) del apéndice tenemos un ejemplo gráfico para la segunda prueba.

Para la primera prueba podemos observar de forma gráfica, en la figura (8), cómo los resultados de los 2 modelos de DAIN son iguales y son los más precisos. El entrenamiento fine-tuning no ha supuesto un cambio considerable. También vemos que Super-SloMo y la interpolación simple dan buenos resultados en esta prueba, aunque son menos precisos que DAIN. La U-Net propia aún tiene mucho margen de mejora, ya que las predicciones que genera tienden a ser muy borrosas, siendo este el motivo de tener un MSE tan malo.

Para la segunda prueba podemos observar de forma gráfica, en la figura (9), cómo los resultados de los 2 modelos de DAIN no son iguales como en el anterior, pero sí muy parecidos. En esta prueba, los modelos DAIN no son los más precisos, aunque ocupan la segunda posición. Aun así, los resultados para estos 2 modelos son muy pobres. El

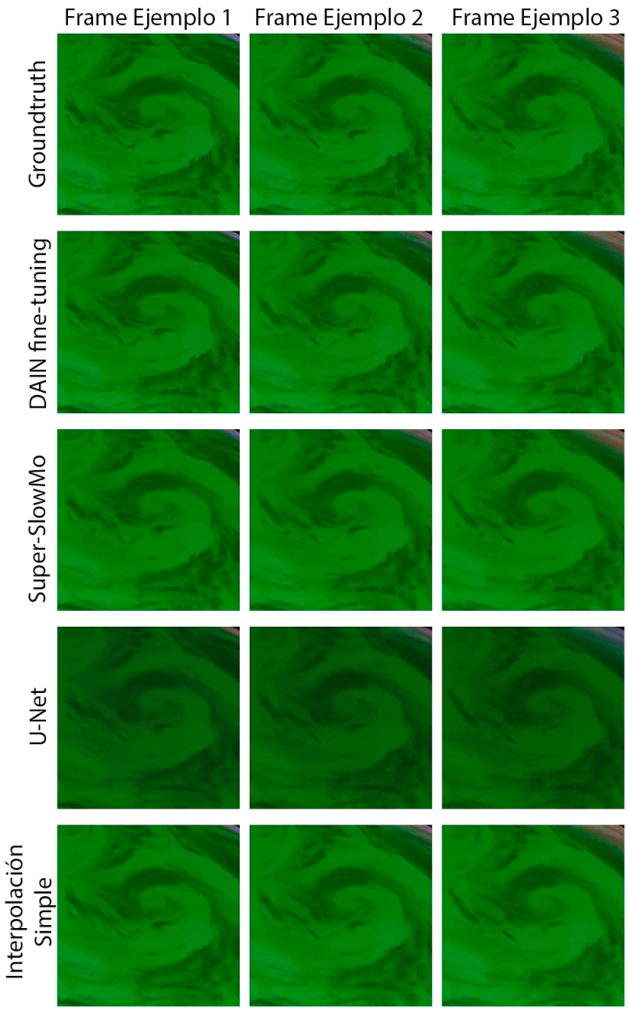

Fig. 7: Ejemplo de los resultados para cada modelo con la experimentación de distancia temporal para la primera prueba (frames consecutivos con separacion de 30 minutos).

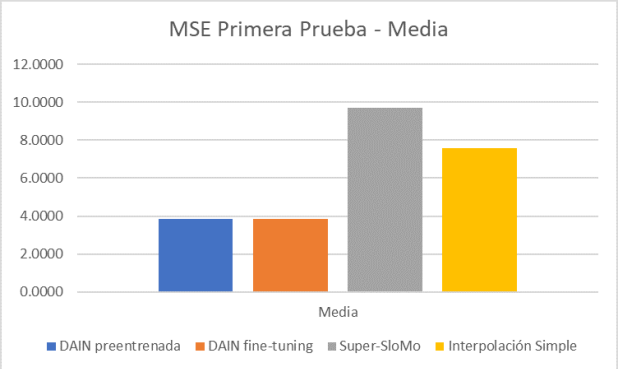

Fig. 8: Experimentación con 3 canales, media de los resultados de la tabla (1) representados de forma gráfica. El resultado de U-Net propia no está incluido, es tan alto que impide apreciar bien los resultados. Prueba de frames consecutivos (30 minutos).

modelo Super-SloMo es el que consigue mayor precisión en esta prueba con diferencia, aunque en comparación con la prueba anterior es muy impreciso. La U-Net propia en esta prueba parece tener el mismo problema que en la anterior prueba. En la interpolación simple se puede

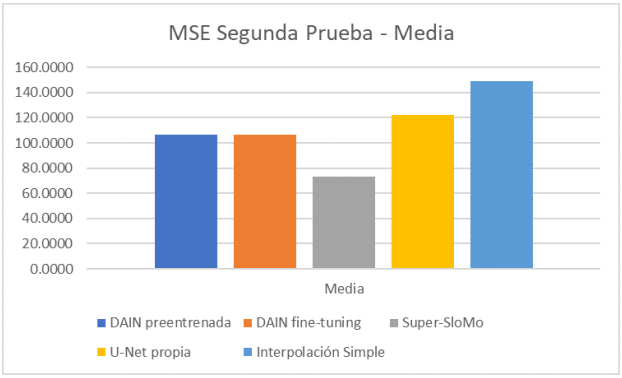

Fig. 9: Experimentación con 3 canales, media de los resultados de la tabla (1) representados de forma gráfica. Prueba con separación de 120 minutos.

comprobar cuanto más distinto son los frames interpolados, mayor será el error.

6.3. Experimentación con más canales

El objetivo de este experimento es estudiar cómo reaccionan los distintos modelos cuando interpolamos imágenes de más de 3 canales. No todos los modelos utilizados pueden hacerlo de forma directa. Los modelos DAIN y Super-SlowMo necesitan que las imágenes que interpolan sean de 3 canales. Es por eso que en este experimento utilizaremos dos métodos distintos a la hora de interpolar. El método directo (U-Net, Interpolación simple) y el método indirecto (DAIN, Super-SlowMo).

6.3.1. Preparación

Lo primero que intentamos fue juntar los 12 canales en una sola imagen. Decidimos descartar el canal HRV (high resolution) porque tenía una resolución distinta al resto de canales, dando como resultado imágenes de 11 canales.

Algunos modelos, de los utilizados en este proyecto, solo soportan imágenes de 3 canales. Para solventar esta limitación, utilizamos un múltiplo de 3 para el número de canales que utilizaremos en las imágenes. Al tener 11 canales utilizaremos 9. De esta manera, dividimos la imagen de 9 canales en 3 imágenes de 3 canales cada una. Así, por cada frame interpolamos las 3 imágenes para luego juntar las interpolaciones, generando así una imagen de 9 canales.

Los canales elegidos son y siguen el orden de: IR 1.6, IR 3.9, IR 8.7, IR 9.7, IR 10,8, IR 12.0, IR 13.4, VIS 0.6 y VIS 0.8. En el apartado (5.1) se encuentra la información espectral de estos canales.

Como anticipamos en el apartado (5.2.1), hemos realizado diferentes dataset para experimentar dependiendo del método utilizado. En el caso del método directo, el dataset generado para realizar las pruebas contiene imágenes con unas dimensiones de 572×572×9.

En el caso del método indirecto, el dataset generado para realizar las pruebas contiene imágenes con unas dimensiones de 572×572×3. Estas imágenes son, en realidad, las mismas imágenes de 9 canales del método directo. El truco está en que hemos dividido las imágenes de 9 canales en 3 imágenes de 3 canales cada una. La primera imagen contiene los canales IR 1.6, IR 3.9 y IR 8.7, la segunda imagen contiene los canales IR 9.7, IR 10,8 y IR 12.0 y la tercera imagen contiene los canales IR 13.4, VIS 0.6 y VIS 0.8. En la figura (10) tenemos un ejemplo gráfico.

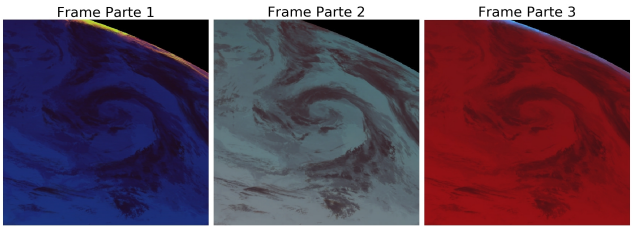

Fig. 10: Ejemplo de la composición de un frame en el método indirecto. La primera parte contiene los canales IR 1.6, IR 3.9 y IR 8.7, la segunda parte contiene los canales IR 9.7, IR 10,8 y IR 12.0 y la tercera parte contiene los canales IR 13.4, VIS 0.6 y VIS 0.8.

Como hemos visto en el apartado (5.2.2), hemos realizado diferentes dataset de entrenamiento dependiendo del método utilizado, de la misma manera que con el dataset de experimentación. En el caso del método directo, el dataset generado para entrenar contiene imágenes con unas dimensiones de 572×572×9. En el caso del método indirecto, el dataset generado para entrenar contiene imágenes con unas dimensiones de 572×572×3. Este dataset es el mismo que el que hemos utilizado en el experimento anterior (6.2).

Para el caso de DAIN utilizaremos su versión preentrenada y su versión con fine-tunning realizado con el dataset generado para el entrenamiento del método indirecto. Los valores de los parámetros utilizados en los entrenamientos son: Learning Rate: 0.0005, Weight Decay: 0.0001, Epochs: 100, Batch: 2, Número de Muestras (train+test): 286, Porcentaje train: 60.0 %.

Para el caso de Super-SlowMo utilizaremos únicamente su versión preentrenada. Para el caso de la U-Net, utilizaremos la versión entrenada desde cero con los valores de los parámetros de entrenamiento utilizados siguientes: Learning Rate: 0.0001, Weight Decay: 0.0001, Epochs: 5, Batch: 3, Número de Muestras (train+test): 286, Porcentaje train: 60.0 %. El dataset utilizado es el generado para el entrenamiento del método directo. Para el caso de la interpolación simple no es necesario ninguna preparación, ya que consiste en aplicar una fórmula directa.

6.3.2. Procedimiento

Como indicamos en el apartado (5.2), el satélite captura un frame cada 15 minutos. Para tener un groundtruth fiable, utilizaremos una captura hecha por el satélite.

Para el método directo hemos realizado las interpolaciones con frames con 9 canales. Esto conlleva a que para la experimentación realizaremos 48 interpolaciones de los 96 frames totales. Los modelos que pueden ejecutar este método son U-Net y la interpolación simple. El MSE ha sido calculado de forma directa.

Para el método indirecto hemos dividido la ejecución de las pruebas en tres partes. Cada parte está centrada en 3 canales. Para ello, hemos juntado los frames de forma concatenada, es decir, hemos generado un vídeo compuesto de las tres partes. Esto significa que el número de frames que interpolamos son de 144, el triple de los 48 que realizamos en el método directo. Los primeros 48 frames contienen los canales IR 1.6, IR 3.9 y IR 8.7, los siguientes 48 frames contienen los canales IR 9.7, IR 10,8 y IR 12.0 y los últimos 48 frames contienen los canales IR 13.4, VIS 0.6 y VIS 0.8. Además, descartamos tener en consideración la interpolación que ocurre entre la unión de los 3 grupos (interpolación entre frame 48–49 y entre frame 96–97), ya que no tendría sentido.

La comparación se tiene que procurar realizar en igualdad de condiciones. Es por eso que, aunque los métodos para lograr interpolar a más de 3 canales sean distintos, realizamos el cálculo del MSE igual, independientemente del método utilizado. Como ocurría en el experimento anterior, el modelo U-Net tiene la limitación de generar un output más pequeño que el input. Por esa razón, a la hora de calcular el MSE del output de cada modelo, realizamos un crop a 388×388 para igualar todos los modelos a la limitación de la U-Net propia.

6.3.3. Resultados

Una vez realizada las pruebas tenemos los siguientes resultados:

TABLA 2: RESULTADOS POR PARTES (MÉTODO INDIRECTO) DE LA EXPERIMENTACIÓN CON MÁS CANALES. LA PRIMERA PARTE CONTIENE LOS CANALES IR 1.6, IR 3.9 Y IR 8.7, LA SEGUNDA PARTE CONTIENE LOS CANALES IR 9.7, IR 10,8 Y IR 12.0 Y LA TERCERA PARTE CONTIENE LOS CANALES IR 13.4, VIS 0.6 Y VIS 0.8.

	MSE más canales Media método indirecto		
	Parte 1	Parte 2	Parte 3
DAIN preentrenada	5.08	9.39	2.95
DAIN fine-tuning	5.06	9.39	2.95
Super-SloMo	12.49	15.98	9.56
U-Net propia	63.77	166.42	190.64
Interpolación simple	8.84	25.23	2.50

Tenemos los resultados de la experimentación de las tres partes en la tabla (2), junto con la tabla (3), donde se encuentra la media total para cada modelo.
En la siguiente figura (11) tenemos un ejemplo de los resultados para cada modelo de cada una de las partes. De esta forma podemos ver el resultado visualmente.

TABLA 3: RESULTADOS CON MEDIA TOTAL DE LA EXPERIMENTACIÓN CON MÁS CANALES.

	MSE más Canales Media total
DAIN preentrenada	5.81
DAIN fine-tuning	5.80
Super-SloMo	12.68
U-Net propia	140.28
Interpolación simple	12.19

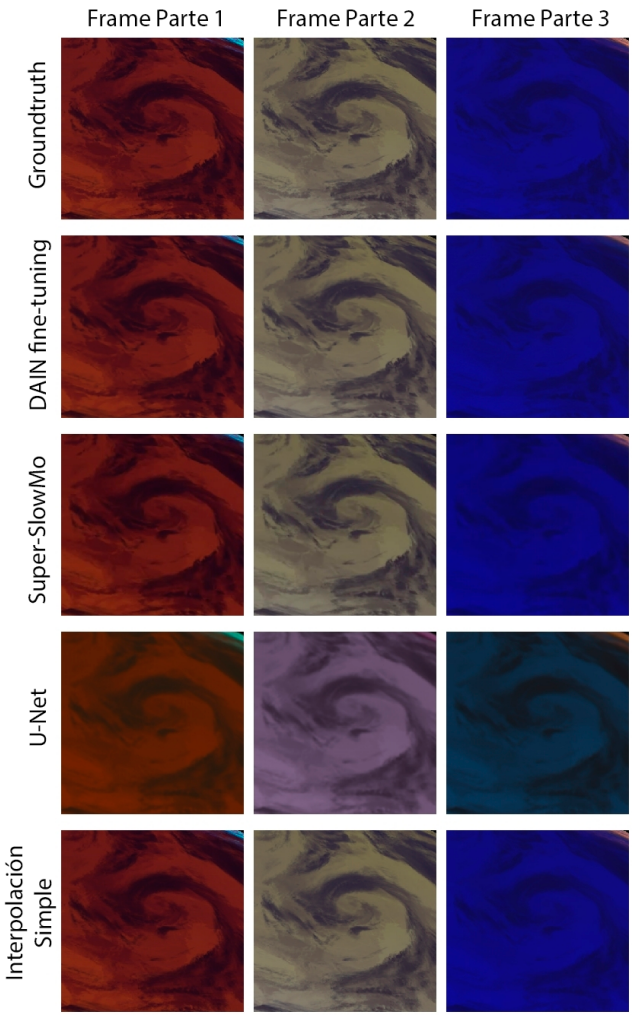

Fig. 11: Ejemplo de los resultados para cada modelo, experimentación con más canales.

Respecto al método indirecto, podemos observar en la figura (12) cómo los resultados de la parte tres son más precisos. Además, el algoritmo DAIN resulta ser más preciso que Super-SloMo. La mejora realizada por el fine-tuning es mínima.

En la figura (13) vemos el MSE promedio resultante de los distintos métodos. Los resultados indican que el mejor método para realizar interpolaciones es DAIN, además, está

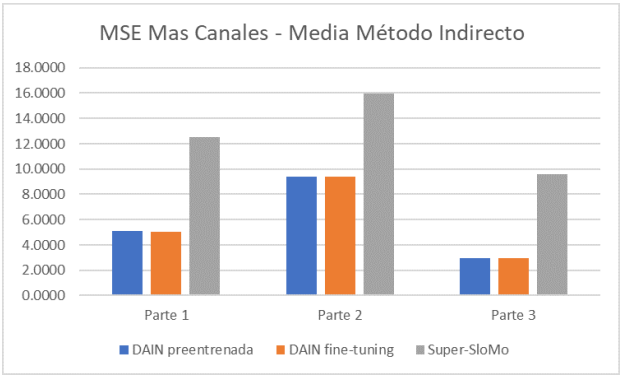

Fig. 12: Experimentación con 9 canales, Tabla (2) representados de forma gráfica. La interpolación simple y U-Net no están incluidos porque el MSE a sido calculado con el método directo.

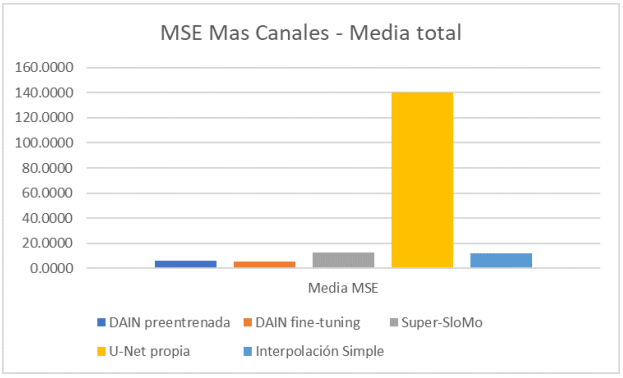

Fig. 13: Experimentación con 9 canales, Tabla (3) representados de forma gráfica.

lo suficientemente preentrenado como para que el realizarle fine-tuning no suponga una mejora notable. Seguido de cerca está Super-SloMo y la interpolación simple. Aun teniendo un resultado parecido, la interpolación simple lineal puede provocar problemas, como por ejemplo ghosting. En el caso de nuestra U-Net, el resultado sigue siendo bastante impreciso en comparación con el resto, además, sigue generando predicciones que tienden a verse borrosas. Esto puede deberse a que necesita más entrenamiento.

6.4. Interpolación con todos los canales

Hemos realizado una pequeña prueba para comprobar el funcionamiento de la U-Net que hemos desarrollado, con frames con el número máximo de canales del dataset. Hemos entrenado la red con un dataset generado por nosotros de imágenes con 11 canales y con los valores de los parámetros de entrenamiento utilizados siguientes: Learning Rate: 0.0001, Weight Decay: 0.0001, Epochs: 5, Batch: 3, Número de Muestras (train+test): 286, Porcentaje train: 60.0 %. En la figura (14) tenemos un ejemplo gráfico para el resultado y en la figura (A.2) del apéndice tenemos un ejemplo gráfico para el groundtruth.

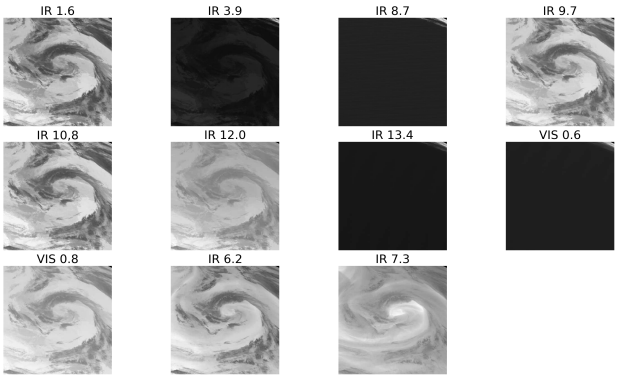

Fig. 14: Ejemplo de resultado gráfico de la interpolación con todos los canales disponibles, 11 canales. (PONER MAS GRANDE)

7 Conclusiones

En este proyecto hemos podido comprobar como diferentes modelos lograban hacer motion interpolation a imágenes de satélite de más de 3 canales. De los modelos seleccionados DAIN ha sido el más preciso. Siendo este el modelo más completo y avanzado de todos, tiene una limitación que comparte con Super-SloMo, que es el segundo modelo más completo, y se trata de que no puede hacer interpolaciones con frames de más de 3 canales de forma directa. No obstante, hemos diseñado un método para que pudiera interpolar frames de más de 3 canales de forma indirecta. Nuestro algoritmo U-Net ha sido capaz de interpolar con frames con cualquier cantidad de canales, aunque mostrando resultados no muy precisos. Creemos que con un entrenamiento mucho más masivo este problema se solucionaría. El método de la interpolación lineal, aun siendo un método simple, ha dado resultados precisos. Este método también es capaz de interpolar con frames con cualquier cantidad de canales.

Durante el desarrollo del proyecto hemos aprendido y profundizado en el ámbito del remote sensing, sobre su composición y los diferentes servicios donde conseguir imágenes tomadas por satélites. También hemos trabajado el tema de la interpolación de vídeos (motion interpolation), cómo funcionan los métodos en estado del arte y hemos trabajado la generación de modelos propios que interpolan temporalmente. Y sobre todo no solo hemos aprendido a trabajar con redes neuronales, sino que hemos aprendido a programar, entrenarlas desde cero y hacer fine-tuning.

Agradecimientos

En primer lugar, quiero agradecer a mi tutor Felipe Lumbreras el tiempo que ha invertido en este proyecto, la ayuda prestada al consultar a los expertos y todas las ideas aportadas durante las reuniones. En segundo lugar, agradecer a EUMETSAT por habernos proporcionado las imágenes satélite necesarias para realizar el proyecto. Por último, quiero dar las gracias al proyecto BOSSS TIN2017-89723-P.

REFERENCIAS

[1] Y. Lee and T. Nguyen. High frame rate Motion Compensated Frame Interpolation in High-Definition video processing. 2010 IEEE International Conference on Acoustics, Speech and Signal Processing, Dallas, TX, 2010, pp. 858–861, doi: 10.1109/ICASSP.2010.5495214.
Accedido: 16-Sep-2020.

[2] LeCun, Yann Bengio, Y. Hinton, Geoffrey. (2015). Deep Learning. Nature. 521. 436–44. 10.1038/nature14539.
Accedido: 16-Sep-2020.

[3] USGS, *"What is remote sensing and what is it used for?"*.
https://www.usgs.gov/faqs/what-remote-sensing-and-what-it-used?qt-news_science_products=0#qt-news_science_products.
Accedido: 16-Sep-2020.

[4] X. Yin, W. Chen, X. Wu and H. Yue. Fine-tuning and visualization of convolutional neural networks. 2017 12th IEEE Conference on Industrial Electronics and Applications (ICIEA), Siem Reap, 2017, pp. 1310–1315, doi: 10.1109/ICIEA.2017.8283041.
Accedido: 16-Sep-2020.

[5] ESA, *"Sentinel-2 MSI Introduction"*.
https://sentinel.esa.int/web/sentinel/user-guides/sentinel-2-msi.
Accedido: 16-Sep-2020.

[6] ESA, *"MeteoSat"*.
https://en.sat24.com/en/sp.
Accedido: 16-Sep-2020.

[7] Bao, Wenbo and Lai, Wei-Sheng and Ma, Chao and Zhang, Xiaoyun and Gao, Zhiyong and Yang, Ming-Hsuan. Depth-Aware Video Frame Interpolation. IEEE Conference on Computer Vision and Pattern Recognition. 2019
Accedido: 16-Sep-2020.

[8] D. Scharstein, H. Hirschmüller, Y. Kitajima, G. Krathwohl, N. Nesic, X. Wang, and P. Westling. High-resolution stereo datasets with subpixel-accurate ground truth. In German Conference on Pattern Recognition (GCPR 2014), Münster, Germany, September 2014.
Accedido: 06-Oct-2020.

[9] Trello, *"Trello"*.
https://trello.com/.
Accedido: 06-Oct-2020.

[10] Fang L, Ye Z, Su S, Kang J, Tong X. Glacier Surface Motion Estimation from SAR Intensity Images Based on Subpixel Gradient Correlation. Sensors. 2020; 20(16):4396.
Accedido: 06-Oct-2020.

[11] Nagler T, Rott H, Hetzenecker M, Wuite J, Potin P. The Sentinel-1 Mission: New Opportunities for Ice Sheet Observations. Remote Sensing. 2015; 7(7):9371–9389.
Accedido: 06-Oct-2020.

[12] Google, *"Google Colaboratory"*.
https://colab.research.google.com.
Accedido: 08-Nov-2020.

[13] Eumetsat, *"High Rate SEVIRI Level 1.5 Image Data - MSG - 0 degree"*.
https://navigator.eumetsat.int/product/EO:EUM:DAT:MSG:HRSEVIRI.
Accedido: 08-Nov-2020.

[14] A. Mikołajczyk and M. Grochowski. Data augmentation for improving deep learning in image classification problem. 2018 International Interdisciplinary PhD Workshop (IIPhDW), Swinoujście, 2018, pp. 117–122, doi: 10.1109/IIPHDW.2018.8388338.
Accedido: 08-Nov-2020.

[15] Jiang, Huaizu Sun, Deqing Jampani, Varan Yang, Ming-Hsuan Learned-Miller, Erik Kautz, Jan. (2018). Super SloMo: High Quality Estimation of Multiple Intermediate Frames for Video Interpolation. 9000–9008. 10.1109/CVPR.2018.00938.
Accedido: 08-Dic-2020.

[16] Ronneberger, Olaf; Fischer, Philipp; Brox, Thomas (2015). U-Net: Convolutional Networks for Biomedical Image Segmentation. Accedido: 08-Dic-2020.

[17] L. Zhang, L. Zhang and B. Du. Deep Learning for Remote Sensing Data: A Technical Tutorial on the State of the Art. in IEEE Geoscience and Remote Sensing Magazine, vol. 4, no. 2, pp. 22–40, June 2016, doi: 10.1109/MGRS.2016.2540798. Accedido: 08-Dic-2020.

APÉNDICE

A.1. Apéndice 1

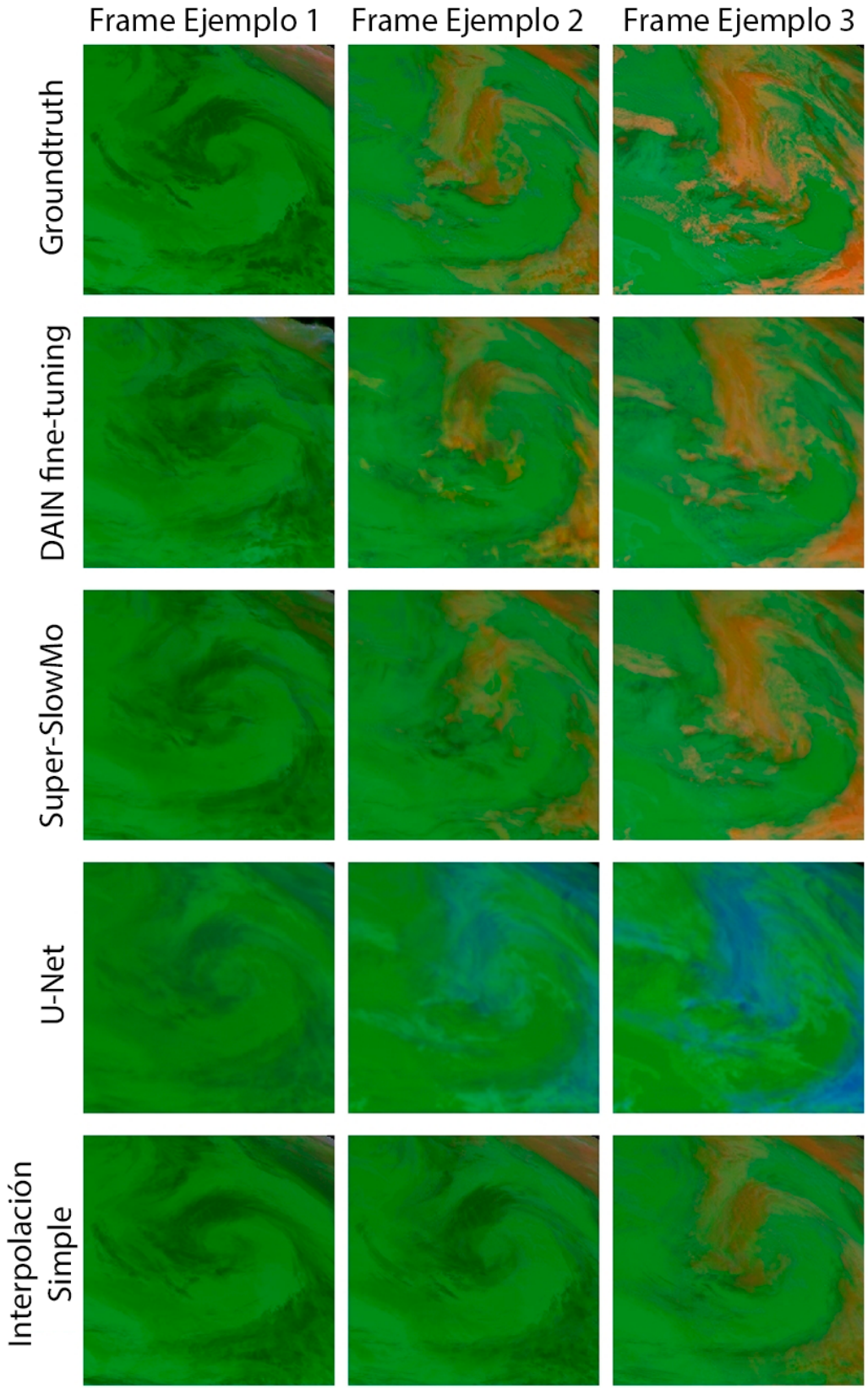

Fig. A.1: Ejemplo de los resultados para cada modelo, experimentación de distancia temporal, segunda prueba.

A.2. Apéndice 2

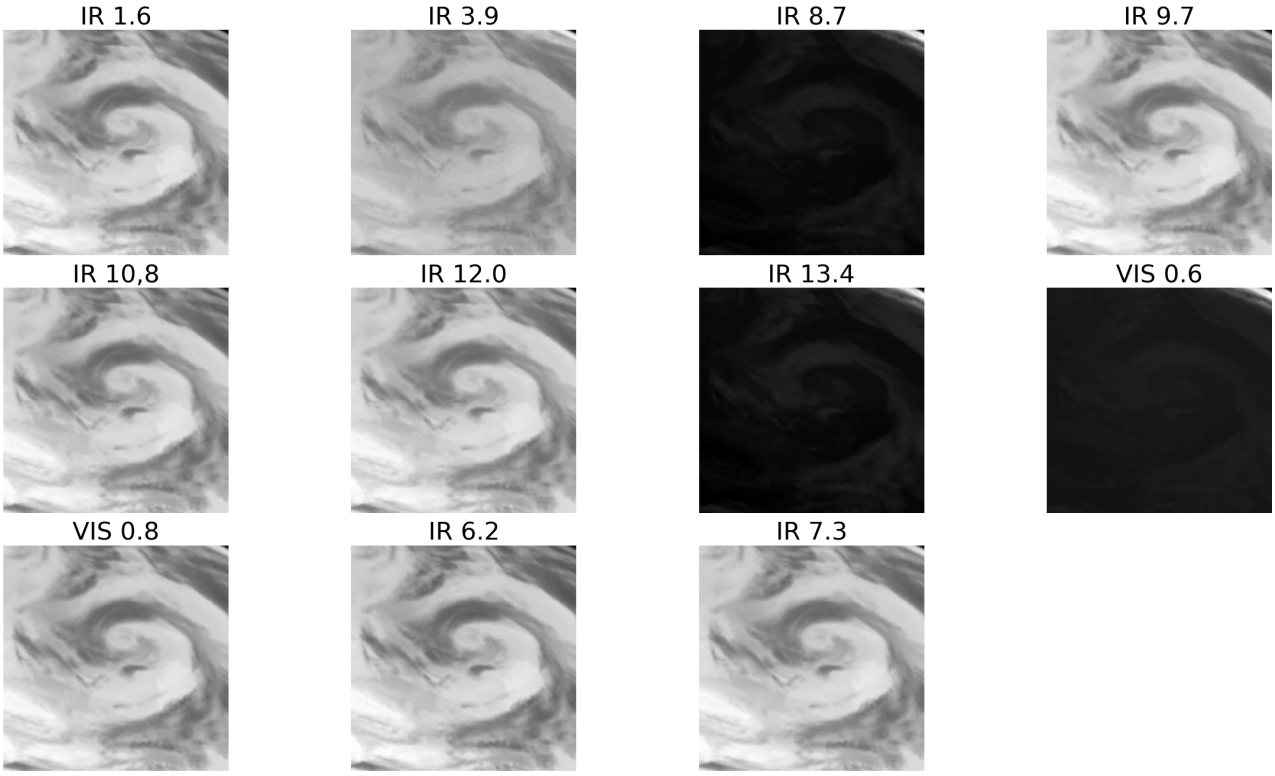

Fig. A.2: Ejemplo del groundtruth gráfico de la interpolación con todos los canales disponibles, 11 canales.

A.3. Apéndice 3

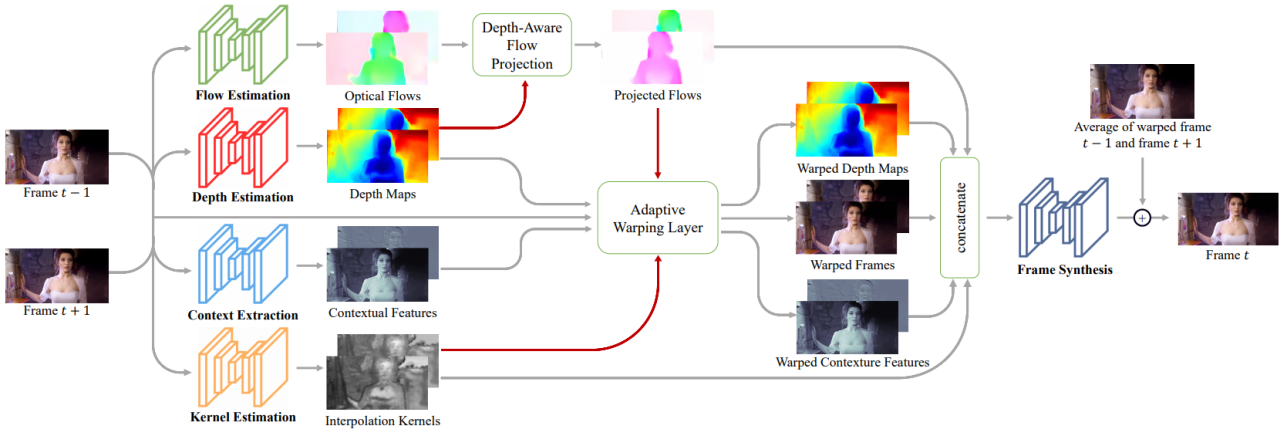

Fig. A.3: Arquitectura del modelo DAIN.