UNIVERSIDAD DE CHILE
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN

USO DE TÉCNICAS PARA LA INTERPOLACIÓN DE MOVIMIENTO EN LA CREACIÓN DE ANIMACIONES 2D

MEMORIA PARA OPTAR AL TÍTULO DE INGENIERO CIVIL EN COMPUTACIÓN

MATÍAS BENJAMÍN SOTO ZÚÑIGA

PROFESOR GUÍA:
IVÁN SIPIRÁN MENDOZA

MIEMBROS DE LA COMISIÓN:
AIDAN HOGAN
MARÍA CECILIA RIVARA ZÚÑIGA.

SANTIAGO DE CHILE
2022

RESUMEN

La motivación para esta investigación vino dada por el proceso estándar de animación. Una de las partes más relevantes del mismo es la intermediación, que corresponde a la creación de fotogramas de transición entre dos poses claves de un dibujo. Inspirado por los avances realizados en el aprendizaje profundo se explora la posibilidad de usar esta tecnología para poder asistir en el procedimiento antes mencionado.

El trabajo realizado consistió en la creación de la red BLURIFE y el programa prototipo FIAI. Ambos fueron enfocados en verificar si la interpolación de movimiento realizada con aprendizaje profundo puede ser usada como herramienta para generar animaciones, y si no, inferir que características faltan.

La red fue ideada como respuesta a los artefactos visuales que se presentan al tratar de aplicar una técnica de interpolación en un video animado. Su estructura consiste en dos módulos enfocados a distintas tareas, con el primero siendo el encargado de generar los fotogramas intermedios mientras que el otro corrige errores que se presenten en las imágenes generadas.

Por otro lado, FIAI corresponde a un programa ligero ideado como un acercamiento de las tecnologías de interpolación para los animadores. Este permite generar fotogramas intermedios de acuerdo con las especificaciones que se le entregue, como sería la aplicación en secciones de imágenes.

Los resultados obtenidos muestran que BLURIFE no cumplió las expectativas de resolver los problemas presentes en RIFE y FIAI presenta utilidad limitada para los animadores. A pesar de esto, se logra determinar que el movimiento generado mediante la red es adecuado, dejando como trabajo a mejorar la confección de los fotogramas y la integración de la tecnología a programas de animación más generales.

A todos los que me han acompañado,
por la paciencia y el apoyo.

Agradecimientos

A mi familia, por ayudarme cuando más lo necesitaba.

A mis amigos, por alegrarme los días.

A mi profesores, por enseñarme todo lo que sé.

A mis gatos, por la tranquilidad que me otorgaban.

Tabla de Contenido

Índice de Tablas

Índice de Ilustraciones

Capítulo 1

Introducción

Dentro del ámbito de la televisión, uno de los pilares de la entretención para niños y adolescentes son las series animadas. Para producirlas, las dos técnicas más usadas corresponden a animación 3D y 2D, con esta última referida a veces como animación tradicional. Entre ambas la más popular es la 2D debido a su costo de producción más bajo, aunque presenta problemas y desafíos que no están tan presentes en la otra, como es la elaboración de un movimiento fluido.

Una animación es comúnmente definida como una serie de imágenes presentadas secuencialmente tal que permitan crear una ilusión de movimiento. Para poder lograr esto las imágenes (también llamadas fotogramas o *frames*) deben ser mostradas a alta velocidad y ser consistentes entre sí, tal que el cerebro complete la secuencia de forma inconsciente [1]. Si bien esta varía dependiendo de lo que se quiere representar, dentro de la industria del cine el estándar es de 24 fotogramas por segundo [2], por lo que un video animado de 20 minutos (duración usual para un capítulo de serie animada) debe contener aproximadamente 28800 imágenes distintas.

Es evidente que esta cantidad es demasiado alta, por lo que animadores han desarrollado técnicas para aliviar la carga. Entre estas se encuentra la división de dibujo en componentes y modificar la imagen cada cierto número de fotogramas [3]. A pesar de esto, la suma de imágenes aún sigue siendo exorbitante, por lo que existen programas para agilizar una de las partes más importantes al momento de crear una animación: la intermediación.

La intermediación (o *inbetweening*) es un proceso particularmente útil para animación realizada pose a pose y sus variantes por lo que su uso es substancial. Esta consiste en la elaboración de imágenes intermedias entre poses importantes de un movimiento (frecuentemente llamados *inbetweens*) [4], por lo que su calidad afecta directamente a la del proyecto completo, la cual puede verse afectada por diversos motivos, desde cortes en tiempos de producción o capital.

Es debido a esto que múltiples herramientas se han desarrollado para poder aumentar la velocidad del proceso al dar asistencia a la producción del dibujo. Es bajo este contexto entonces que se propone el siguiente trabajo de memoria, el cual corresponde a la red neuronal BLURIFE y su

programa asociado FIAI: la construcción de fotogramas intermedios sin intervención del animador.

1.1. Justificación

La idea de reducir el trabajo relacionado con la intermediación ha sido explorado de múltiples formas. Versiones primitivas de software que servían para apoyar en el proceso restringían los dibujos a figuras simples, las cuales podían ser animadas con fórmulas matemáticas. A medida que transcurrió el tiempo comenzaron a aparecer aplicaciones que permitían generar la intermediación de cualquier dibujo al añadir cierta configuración a las imágenes clave [5]. Para poder efectuar esto, los programas interpolan la información entregada por el usuario haciendo uso de alguna técnica ya existente, aunque los resultados obtenidos son insuficientes cuando los dibujos aumentaban en complejidad [6]. Debido a esto, los programas empezaron a incluir herramientas para asistir el método tradicional.

Desde lectura de líneas y su correspondencia entre fotogramas, la simulación de equipamiento físico para el proceso [7] y el coloreo inteligente de los dibujos [8], la producción de imágenes volvió a ser una labor intensamente humana. Esto conllevó a un incremento en la precariedad de las condiciones laborales dentro de la industria de animación, en particular para los animadores de inbetweens cuando la demanda por esta aumentó. Debido a lo anterior, la necesidad de establecer un método de asistencia que directamente reduzca la cantidad de trabajo a realizar se torna indispensable.

A fecha actual, métodos para producir fotogramas de forma determinística no han sido explorados a causa de la dificultad del problema, aunque sí mediante aprendizaje profundo, los cuales si bien dan resultados prometedores, aún poseen fallas. Errores visuales, movimientos inconsistentes y líneas borrosas suelen plagar estas técnicas [9], debido a que son adaptaciones de herramientas para videos reales, los que no siguen los principios de animación utilizados al crear una caricatura animada [4].

Finalmente, el trabajo está pensado como una asistencia a animadores, por lo que su alcance no abarca la producción de una animación sin inbetweens hechos manualmente, sino que se limita a la creación de fotogramas que cumpla los estándares dispuestos por animadores.

1.2. Conceptos claves

Este proyecto abarca temas relacionados con el aprendizaje de máquinas y visión de computador, por lo que a continuación se proveen la definición de aquellos relevantes para el trabajo.

Aprendizaje profundo (también conocido como *deep learning*) es una rama del aprendizaje de máquinas que busca resolver un problema mediante un método computacional que aprende como representar su entrada y como usar esta para formular la solución. Su calificación de "profundo" proviene del hecho que la representación de los datos es construida a través de múltiples iteraciones, con cada una aumentando su abstracción [10].

Un concepto relevante son las neuronas artificiales, una unidad de cálculo que busca simular una neurona real. Esta unidad corresponde al resultado de una función de activación S aplicada sobre una entrada $X = x_1, \ldots, x_n$, un peso $W = w_1, \ldots, w_n$ y un sesgo b de la siguiente forma:

$$output = S(x_1 w_1 + \cdots + x_n w_n + b) \tag{1.1}$$

Una red neuronal es un conjunto de neuronas artificiales conectadas entre sí que se envían señales, permitiendo procesar una entrada para obtener valores de salida. Estas redes son uno de los métodos de aprendizaje profundo más relevante ya que el proceso de aprendizaje se traduce solo a la adaptación de las neuronas. Las primeras creadas correspondían a las denominadas *feedforward*, en donde las neuronas eran organizadas en conjuntos denominados capas las cuales transmiten sus señales a aquellas que aún no habían procesado la representación. Estas características permitían abordar múltiples problemas, pero presentaba carencias que fueron abordadas por posteriores variaciones.

Una de estas variaciones son las redes neuronales recurrentes, las que se caracterizan por la presencia de al menos un ciclo entre las capas. Este permite que el procesamiento de una entrada afecte el resultado obtenido por las siguientes, lo cual induce un comportamiento dinámico respecto al tiempo o frente a múltiples entradas.

Otra variación son las redes neuronales residuales [11], las cuales se caracterizan por ser capaz de utilizar la salida de una capa antigua en una nueva que no sea necesariamente la siguiente, permitiendo así tener redes una mayor profundidad al reducir el desvanecimiento de gradiente, conllevando a una mejor solución.

1.3. Objetivos

1.3.1. Objetivo general

El objetivo general de este trabajo de memoria consiste en verificar el uso de técnicas de interpolación de movimiento basadas en aprendizaje profundo como herramientas para el proceso de intermediación. Esto comprende la adaptación de estas técnicas y el desarrollo de una nueva para producir fotogramas intermedios de animaciones 2D con alta calidad visual, que se ajuste a las necesidades de los animadores.

1.3.2. Objetivos específicos

- Crear una red enfocada a la interpolación de movimiento que logre evitar los problemas presentes al aplicar técnicas similares en animación 2D mediante el uso de métodos de procesamiento de imágenes enfocados a reducir el desenfoque.
- Adaptar las técnicas de interpolación de movimiento existentes para que tengan un uso más flexible, permitiendo que el animador determine aquella que más se adecue a su situación.
- Diseñar e implementar un programa amigable, intuitivo y altamente interactivo que haga uso del trabajo realizado para proveer a animadores resultados que satisfagan sus necesidades.

1.4. Metodología

1. Estudio y preparación

   En esta etapa se realizó la búsqueda de técnicas de interpolación que tuvieran buenos resultados al ser aplicados en animaciones 2D, además de buscar librerías y herramientas que fueran necesarios para el desarrollo del proyecto. También se determinó la estructura general del programa.

2. Aplicabilidad de redes

   Esta sección consistió en la creación de funciones y clases que permitieran usar las redes existentes de forma más controlada, permitiendo aplicarlas en secciones de videos ya sean temporales y/o espaciales.

3. Adaptación de redes

   Esta etapa abarcó el reentrenamiento de las técnicas encontradas para mejorar su rendimiento con fotogramas de animaciones 2D.

4. Creación de nueva red

   En esta parte se realizó la creación y el entrenamiento de la nueva red al extender una de las técnicas encontradas con un módulo destinado a corregir secciones borrosas de los resultados.

5. Creación de interfaz para el programa Abarca la creación de la interfaz gráfica de usuario del programa y su interacción con el manejador de videos.

6. Manejo de videos del programa

   Se crean las características necesarias para manejar el video. Estas incluyen carga de los datos de video, extracción de fotogramas específicos para mostrar el usuario y enviar a las redes, métodos para reducir el consumo de memoria RAM, entre otros.

1.4.1. Evaluación

Para evaluar la calidad de interpolación se usan medidas estándares para estas, como lo son *Peak Signal-to-Noise Ratio* [12], que mide el nivel de distorsión promedio dentro de una imagen obtenida a partir de otra, y *Structural Similarity Index* [13], que determina la similitud entre las estructuras de dos imágenes.

También se hace uso de *Learned Perceptual Image Patch Similarity* [14], una métrica basada en la distancia obtenida mediante una red neuronal entre características profundas de una imagen, en conjunto con reseña de animadores para verificar la calidad de acuerdo al ojo humano.

Para complementar estas métricas se realiza un estudio con individuos con distintos niveles de experiencia en el campo de la animación para que provean una evaluación cualitativa.Por último, se verifica la usabilidad de la solución con pruebas de usuarios realizadas con sujetos externos a la confección de este proyecto.

La evaluación busca obtener información sobre la viabilidad de las redes de interpolación de video como herramienta para el proceso de animación 2D. Esto implica determinar las características de los resultados que sirvan para tal objetivo y establecer posibles cursos de acción que las mejoren. Por consiguiente, se dará especial atención a la sección cualitativa del proyecto, basándose en que esta es aquella que permite a los animadores dar retroalimentación efectiva. Esto requiere entonces que la evaluación sea acorde a casos de uso reales o posibles para tal tecnología.

1.5. Descripción de solución

El proyecto se concentró en dos partes principales: la creación de fotogramas intermedios y el acercamiento de estas como herramientas.

La primera parte consiste en el desarrollo de una red nueva llamada BLURIFE enfocada a la creación automática de inbetweens que busca circunventar los problemas presentados por otros métodos o técnicas. Esta está compuesta de dos secciones: un predictor y limpiador. El primero toma dos fotogramas consecutivos y produce uno que corresponda a una transición entre ambos. El

segundo se encarga de corregir, en lo mejor que pueda, los errores visuales que se presenten en el resultado arrojado por el predictor. Para poder verificar cómo comparaba la red creada con respecto a aquellas que son comúnmente utilizadas se realizó un ajuste de parámetros de estas con un set de datos de animación 2D.

La segunda parte consistió en el programa FIAI, que transforma estas redes en herramientas para animadores al otorgar funcionalidades relacionadas con la edición de video como lo sería la eliminación de fotogramas y la creación de inbetweens. El software fue realizado con programación orientada a objetos y se divide en dos componentes principales: GUI y Procesamiento. El primero se ocupa de recibir entradas del usuario y envía mensajes al procesador. El segundo se encarga de manejar el video en sus distintas etapas, como el orden de fotogramas, los estados de las redes, reducir el uso de memoria y generar las imágenes intermedias de acuerdo con las instrucciones dadas, entre otros.

1.5.1. Resultados

Los resultados del trabajo indican que BLURIFE no cumplió con las expectativas dadas en la sección anterior, demostrando un rendimiento menor con respecto a otros métodos de interpolación existentes, pero aun así comparables. Por otro lado, FIAI demostró ser insuficiente como herramienta para la creación de inbetweens por sí misma, aunque podría ser relevante como característica tipo plug-in dentro de otro software de animación. Se muestra también que el movimiento interpolado por las redes es apropiado para los propósitos de los animadores, implicando que la presencia de artefactos visuales es el principal impedimento para que la tecnología sea usada como instrumento de animación.

Tanto BLURIFE como FIAI se encuentran disponibles en github

Capítulo 2

Estado del arte

2.1. Interpolación de video

En términos de técnicas para interpolar movimiento, estas aún están en proceso de desarrollo sin ningún estándar definido. Actualmente se pueden identificar 4 corrientes de interpolación:

- Fases
- Flujo óptico
- Aprendizaje profundo puro
- Híbridos

Dentro de estos el más popular es el método híbrido, como RIFE o DAIN [15, 16], aunque múltiples versiones de este comparten el problema de que son entrenados en su mayoría con videos reales, como lo son Middlebury [17], Vimeo90k [18] y UCF101 [19], por lo que no dan los mejores resultados al usarlos con animaciones 2D.

2.1.1. Interpolación basada en fases

Los métodos basados en fase consideran que cada imagen es una combinación lineal de ondas, permitiendo realizar extrapolación a partir de ellas [20]. En este tipo de técnica, el fotograma es descompuesto a través de la transformada de Fourier para obtener su representación mediante ondas, las cuales se mueven a lo largo del tiempo. Tal movimiento puede ser expresado como un cambio de fase, el cual al ser modificado produce la interpolación.

2.1.2. Interpolación basada en flujo óptico

Estos métodos [21] asumen que el movimiento presente dentro de un video es suave, continuo y que todo píxel de cada fotograma tiene su equivalente en otros. Esta suposición se ve reflejada en su definición:

$$(u,v,d) \text{ tal que } I(x,y,t) = I(x+u, y+v, t+d) \tag{2.1}$$

En esta ecuación el flujo óptico viene dado por (u,v,d), donde $I(x,y,t)$ representan al píxel (x,y) en el tiempo t. Una vez obtenido los valores, estos métodos usan un factor sobre él para generar la imagen interpolada. Por lo general estas técnicas buscan optimizar el flujo al desarrollar nuevas formas para calcularlo, ya sea mediante diferenciales, correspondencia de regiones, energía o fases [22].

2.1.3. Interpolación basada en aprendizaje profundo puro

Los métodos de aprendizaje profundo puros buscan en cambio generar las imágenes interpoladas de forma directa. Este tipo de método comenzó a tomar relevancia el 2017, donde se produjo uno en que se le entregaban dos fotogramas consecutivos a una red neuronal la cual retornaba un par de kernels (matrices) los cuales al convolucionarlos con las imágenes utilizadas producían el interpolado [23]. A partir de esta idea surgieron modificaciones a la red, agregando capas o alterando la forma con que se produce la imagen interpolada [24].

2.1.4. Interpolación por métodos híbridos

Estos métodos por lo general hacen uso del aprendizaje profundo y flujo óptico para producir las imágenes interpoladas. Un ejemplo de esto se encuentra DAIN [15], el cual consiste en múltiples redes neuronales diseñadas para extraer distintos aspectos de dos imágenes, como son las características, el flujo óptico, estimación de kernels y profundidades. Una vez finalizada la extracción, se utilizan estos datos en una red que sintetiza el fotograma interpolado a partir de ellos y las imágenes entregadas.

Dentro de los modelos híbridos resaltan RIFE [16] y SoftSplat [25]. El primero extrae el flujo óptico utilizando el método Coarse-to-fine [26] y genera una máscara de fusión tal que, tras haber realizado la deformación reversa [27] con el flujo obtenido, se crea la imagen con esta, refinando detalles con una variación de UNet [28]. Este curso de acción permite que sea rápido al momento de interpolar. El segundo en cambio utiliza una versión refinada de PWC-Net [29] para el cálculo del flujo, mientras que desarrollan un método de deformación especializado en conjunto con una red para la síntesis del fotograma intermedio. Cabe notar que este método es ligero en términos de parámetros.

2.2. Perfilación de video o *Deblurring*

La perfilación por video busca utilizar los fotogramas aledaños para producir una imagen clara. El método más común para lograr esto consiste en extraer características de las imágenes para luego procesarlas y efectuar la perfilación.

Ejemplo de este método es EDVR [30], el cual hace uso de una técnica grueso a fino para la alineación de características entre fotogramas, además de guiar el proceso de perfilador en las últimas capas haciendo uso de información extraída de las otras imágenes.

Por otro lado, también existen métodos para propagar la perfilación a través del tiempo, como lo hace ABIN [31]. En este caso se corrige un fotograma central el cual es utilizado posteriormente para perfilar los cuadros aledaños, otorgando consistencia.

Una característica común de estos métodos es el uso de bloques residuales [11] para procesar las características relevantes de las imágenes a costo de omitir aquellas menos evidentes [32].

Un problema que poseen estos enfoques es su necesidad de utilizar múltiples fotogramas consecutivos (en los dos casos ya mencionados se usan 5). Hsu et al. [33] buscó evitar esto al guardar información sobre usos anteriores para proveer un método de perfilación de video que reciba como entrada un solo fotograma.

2.3. Interpolación asistida por computadora

En el ámbito de intermediación los programas usados actualmente asisten el proceso al permitir una comparación directa del dibujo intermedio con los fotogramas clave [34]. Otro tipo de software usado permite producir dibujos interpolados, aunque requiere que las poses claves sean definidas en términos de vértices para poder predecir el movimiento a partir de estos [35] lo cual puede resultar en pérdida de tiempo.

Por otro lado, la creación de dibujos intermedios mediante computadores es un área que se ha apoyado en el concepto de un dibujo como colección de líneas. Bajo esta suposición se han desarrollado sistemas para detectar como estas se deforman a lo largo del tiempo. Dentro de los primeros creados se encuentra el propuesto por Kort [36], el cual usaba métodos deterministas para encontrar la relación entre líneas de forma automática. Posteriores iteraciones de este concepto mejoraron los métodos para encontrar la relación y se enfocaron de dar más control al animador al permitirle definir rutas predeterminadas [37].

Este desarrollo eventualmente conllevó al uso de aprendizaje profundo para asistir en la creación de dibujos. Li et al. propuso una red capaz de generar cuadros intermedios complejos haciendo uso de los dibujos asociados y un boceto del cuadro final [38]. Por otro lado, Zhao et al. creó el set de datos ATD-12K en conjunto con proponer AnimeInterp [39], el cual puede producir fotogramas interpolados en su completitud. Esto lo hace identificando objetos en las imágenes entregadas y asociarlos entre fotogramas para crear un flujo óptico más acorde para animación 2D, como se ve en la figura 2.1. A pesar de presentar mejor resultado que los métodos mencionados previamente, sufre de largos tiempo de inferencia que no justifican la calidad generada.

Figura 2.1: Interpolación dada por AnimeInterp

2.4. Herramientas para la intermediación

Por lo general las herramientas relacionadas con la intermediación de animaciones 2D vienen incluidas dentro de programas [40–44] más generales, por lo que no tienen mucho enfoque. A pesar de esto sí se pueden encontrar ciertas características comunes.

2.4.1. Tabla de luz digital

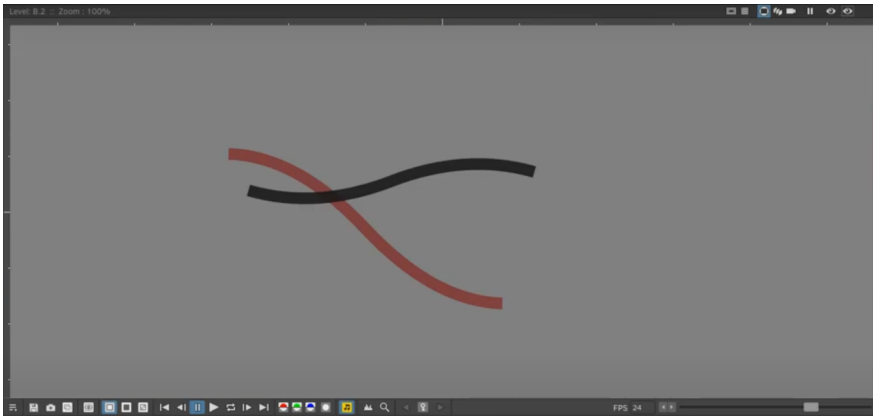


Figura 2.2: Tabla de luz digital de OpenToonz. La línea roja corresponde al fotograma anterior

La herramienta es una emulación de las tablas de luz reales, las cuales permiten vislumbrar los bocetos de animación a través de múltiples hojas para poder establecer de mejor forma los inbetweens. Dentro de los programas existentes esta funcionalidad se presenta como la silueta de un fotograma determinado que se antepone al dibujo con el que se está trabajando, tal cual se ve en la figura 2.2.

2.4.2. Interpolación de trazos

Esta opción solo es posible si el software tiene ciertos métodos para trabajar con imágenes vectoriales [45], ya sea dando opciones para que el usuario vectorice directamente la imagen ráster [45] o se haga de forma automática.

Las imágenes vectorizadas pueden ser interpoladas basándose en distintos movimientos, los cuales depende de cada programa. Por ejemplo, CACANi [41] tiene la opción de interpolar simulando inercia de los objetos. Para poder efectuar esto, se da la capacidad de crear múltiples fotogramas de acuerdo con la interpolación seleccionada.

Figura 2.3: Ejemplo de interpolación de CACANi

Como añadido a lo anterior, también se ofrece la posibilidad de determinar que sección de la interpolación será mostrada. Es decir, dado un movimiento interpolado, se puede definir que punto de este se va a plasmar en un fotograma.

2.4.3. Herramientas generales de video

Como parte de las herramientas generales de animación se incluyen:

- Explorador de fotogramas
- Creación de fotogramas en blanco
- División de dibujo en capas
- Eliminación de fotogramas
- Reproductor de video

Capítulo 3

Problema

Como se mencionó en capítulos anteriores, la problemática a solucionar es la producción de fotogramas intermedios en una animación. En particular, se busca que, dado dos secciones de fotogramas claves en formato ráster o mapa de bits, se produzca una imagen ráster que sirva como transición entre ambas, usando inteligencia artificial. La solución propuesta debe ser capaz de ser usada, con relativamente poco problema, por un animador.

3.1. Relevancia

En la industria de la animación, la etapa de intermediación abarca la mayor parte del trabajo crudo, es decir, no creativo [6]. Debido a esto, no tienen mucho enfoque en el proceso de producción, a pesar de su amplio impacto en el resultado final. Esto conlleva a que, si bien se desarrollen herramientas para este proceso, estas sean insuficientes.

Un ejemplo de esto son los métodos de interpolación de imágenes en programas existentes [41, 43]. En estos, un trazo de imagen puede ser interpolado inmediatamente después de ser dibujado, de múltiples formas para poder obtener el movimiento deseado. No obstante, tal característica es permitida por el hecho que los sistemas vectorizan el trazo antes de interpolar, y es con estos datos lo que se genera la interpolación, en vez de la imagen misma. Para imágenes y dibujos más complejos esta vectorización puede fallar o ser solicitada al usuario, con ambos siendo efectos indeseados.

Es por ello por lo que se hace necesario la creación de un sistema de interpolación que sea utilizable directamente con imágenes ráster. Si bien se han buscado formas para lograr esto de manera determinista, el desafío es altamente complejo. Para solventar esto, se puede utilizar técnicas para restaurar la tasa de fotogramas por segundo de videos, ya que suponen un problema con parámetros similares.

3.2. Requisitos y calidad

La solución a obtener debe cumplir ciertas metas que son obligatorias para resolver el problema:

- Genera una imagen dado dos secciones de fotogramas consecutivos en un video
- Se ejecuta independiente de la resolución del video
- Borra fotogramas según lo dictado por el usuario
- Muestra fotogramas solicitados por el usuario
- Crea un video según lo dictado por el usuario

Por otro lado, la calidad de los resultados debe ser considerada mediante ciertos estándares relacionados con la animación:

- Los fotogramas creados poseen líneas claras
- Los fotogramas creados interpolan correctamente las imágenes dadas
- El movimiento obtenido es consistente en transformaciones, trayectorias y dinámicas de los objetos

Capítulo 4

BLURIFE

La nueva red BLURIFE busca circunventar el problema de los artefactos visuales que ocurren al procesar un par de fotogramas con alguna red de interpolación de video. Considerando el buen rendimiento que poseen estas redes, se optó por retocar los errores que aparecieran, en vez de crear un método completamente novedoso. Para corregir esto se debe nota que las fallas que aparecen por lo general adquieren la forma de zonas borrosas, por lo que una técnica para perfilar imágenes debiera lograr lo buscado.

La estructura general de la red se ve en la figura 4.1. Esta está compuesta por dos partes: el predictor, que genera el fotograma intermedio y el limpiador, el cual busca corregir errores visuales. Una explicación más detallada de estos será explorada en las siguientes secciones.

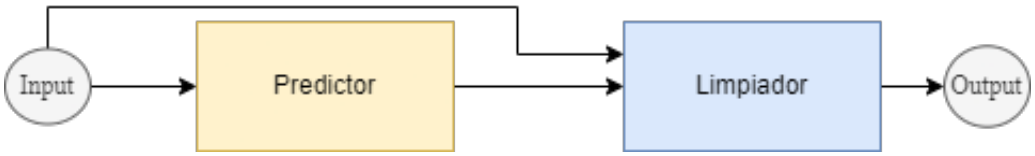


Figura 4.1: Estructura de la red

4.1. Predictor

Como predictor se usa la versión inicial de RIFE [16] debido a su alto rendimiento a pesar de su baja profundidad y tiempo de ejecución. Este genera la imagen interpolada usando el siguiente método:

$$\hat{I}_t = M \odot \hat{I}_{t\leftarrow 0} + (1 - M) \odot \hat{I}_{t\leftarrow 1} \tag{4.1}$$

$$\hat{I}_{t\leftarrow 0} = w(I_0, F_{t\rightarrow 0}),\ \hat{I}_{t\leftarrow 1} = w(I_1, F_{t\rightarrow 1}) \tag{4.2}$$

Este toma como entrada dos fotogramas RGB I_0 y I_1 para luego, usando la máscara M y los flujos ópticos entre la imagen intermedia real con las de entrada $F_{t\to 0}$ y $F_{t\to 1}$, generar la imagen interpolada $\hat{I}_t$ donde w es la deformación reversa y $\odot$ es el producto matriz de Hadamard. El resultado obtenido es posteriormente procesado por una red encoder-decoder llamada RefineNet para reducir artefactos visuales.

La máscara M y los flujos ópticos bidireccionales $F_{t\to 0}$ y $F_{t\to 1}$ son determinados utilizando un método *Coarse-to-fine* mediante IFBlocks a distintas resoluciones de I_0 e I_1. Esto consiste en definir un flujo y mapa inicial con imágenes de bajas resoluciones e iterar el proceso a resoluciones más altas, obteniendo un resultado más refinado. Esto se puede expresar como:

$$(F^i, M^i) = (F^{i-1}, M^{i-1}) + IFB^i(F^{i-1}, M^{i-1}, \hat{I}^{i-1}) \tag{4.3}$$

donde F^{i-1} y M^{i-1} representan las estimaciones actuales del flujo y la máscara respectivamente mientras que IFB^i representa el bloque IFBlock actual.

Por otro lado, RefineNet extrae las características contextuales de las imágenes las cuales son deformadas de forma reversa con los flujos obtenido con los IFBlocks para luego ser usadas como entrada en una red encoder-decoder basada en UNet en conjunto con la máscara, los flujos y las imágenes. Esto da como resultado un residuo Δ el cual se puede usar para tener una imagen más refinada mediante:

$$I_t = \max(1, \min(\hat{I}_t, 0)) \tag{4.4}$$

Esta sección del trabajo no presentó cambios relevantes a la red por lo que no será indagado más en profundidad. Para mayor información respecto a esta sección, referirse al estudio indicado [16].

4.2. Limpiador

Como limpiador se usó una variación de la red SOON [33] denominada SSOON, enfocada en reducir líneas borrosas que se presentaran en la imagen producida por el predictor. Se decidió ir por esta opción debido a su cualidad de generar una perfilación general sin requerir como entrada una gran cantidad de fotogramas. Esta red busca disminuir los artefactos visuales producidos utilizando información obtenida al procesar datos de características extraídas de fotogramas anteriores.

Los cambios realizados a esta red estuvieron enfocados en mejorar el rendimiento de la red para sucesiones de fotogramas más cortas, a la vez que reducir el consumo de memoria para poder ser usado con el predictor.

4.2.1. Estructura

La red propuesta posee una estructura tipo encoder-decoder recurrente y residual, la cual viene dada en la figura 4.2 junto con la tabla 4.1. En la figura se tiene que cada bloque representa capas o tensores relevantes para la red:

- Los blancos corresponden a sets de convoluciones
- El azul representa los bloques residuales [11], cuyos detalles se encuentran en el apéndice
- El verde corresponden a la celda convLSTM [46]
- El amarillo denota al mapa de características decodificadas de la imagen entregada

El uso de la estructura encoder-decoder viene por el hecho que esta ha sido usada prominentemente en tareas de visión de computador debido a su capacidad de procesar datos completos de la imagen al finalizar su proceso de codificación. La recurrencia se debe a los dos métodos para traspasar información de fotogramas anteriores a los nuevos: traspaso de características y convLSTM [46], los cuáles serán profundizados posteriormente. Por último, como la imagen objetivo no disimila lo suficiente de la dada a la red, es más razonable tratar de descubrir los puntos en que se diferencian y corregir estos que generar la imagen completa nuevamente, por lo que resulta beneficioso que la red sea residual y posea bloques residuales [11], los cuáles también servirán para sobrellevar mejor la profundidad de la red completa.

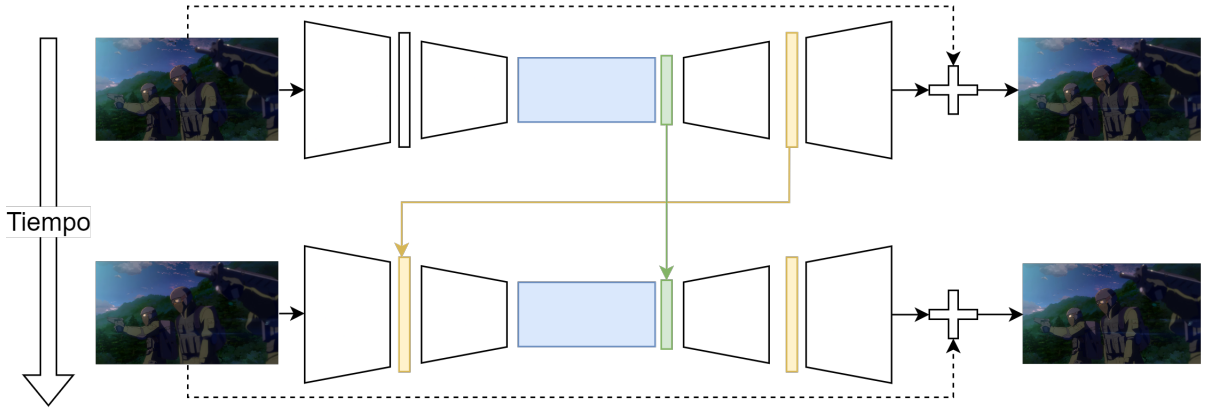


Figura 4.2: Estructura general de la red SSOON

4.2.1.1. Traspaso de características

La primera forma de recurrencia es el traspaso de características decodificadas entre entradas de la red. Considerando que la red es residual, confeccionar una red recurrente pura no obtendría resultados tan notables, pues el enfoque es encontrar diferencias. Debido a esto, es más razonable traspasar el mapa decodificado de características para entregar información relevante a la red, permitiendo obtener una mejor corrección.

Esto se realiza almacenando el resultado de la convolución previa a la capa de salida en una variable p_f. Esta se concatena con la salida de la tercera capa, para luego ser utilizado como entrada para la cuarta. Si la variable no está definida al momento de procesar la cuarta capa, esta se inicializa como un tensor de ceros con tamaño apropiado.

Tabla 4.1: Detalle de la red

Capa	Operaciones	Canales	Kernel	Stride	Padding
Conv1	ReflectionPad Conv2d, BatchNorm, ReLU	- 16	- 3	- 1	3 0
Conv2	Conv2d, BatchNorm, ReLU	32	3	1	0
Conv3	Conv2d, BatchNorm, ReLU	64	3	1	0
Conv4	Conv2d, BatchNorm, ReLU	64	3	1	1
DownConv1	Conv2d, BatchNorm, ReLU	128	3	2	1
DownConv2	Conv2d, BatchNorm, ReLU	256	3	2	1
ResBlocks	Repetir 9	-	-	-	-
ConvLSTM	ConvLSTM	256	-	-	-
UpConv1	ConvTranspose2d, BatchNorm, ReLU	128	3	2	1
UpConv2	ConvTranspose2d, BatchNorm, ReLU	64	3	2	1
OutputL	ReflectionPad Conv2d, Tanh	- 3	- 7	- 1	3 0

4.2.1.2. ConvLSTM

La segunda forma de recurrencia es la celda convLSTM [46]. Esta corresponde a una versión convolucional de una capa LSTM convencional [47]. Su uso como método para retener información temporal y espacial ha sido documentado en su estudio correspondiente y posteriores, por lo que es

usado para traspasar patrones relevantes entre procesamientos de entradas.

4.3. Implementación

El entrenamiento de la red general consistió en el siguiente proceso: se entrenan RIFE y SSOON por separados con sus set de datos respectivos, para luego ser afinados en conjunto.

4.3.1. RIFE

RIFE fue entrenado de la misma forma que fue detallada en su estudio [16], por lo que no será detallado en este informe.

En forma general, RIFE se entrena por 300 épocas utilizando una función de pérdida compuesta, utilizando ADAM con decaimiento de pesos. La tasa de aprendizaje se reduce de 3×10^{-4} a 3×10^{-5} con *cosine annealing* durante todo el proceso [48].

El set de datos usado es Vimeo90K [18], el cual es aumentado recortando los fotogramas en tamaño 224×224, invirtiendo el orden, volteando o rotando en 0, 90, 180 o 270 grados durante entrenamiento, mientras que en el testeo se usa directamente la imagen completa.

4.3.2. SSOON

SSOON se entrenó por 1000 épocas utilizando la función de pérdida *L2-norm*, utilizando ADAM como el optimizador. La tasa de aprendizaje es fija en 0.0001 durante los primeros 500 epochs, para luego decaer de forma lineal hasta 0. El tamaño del lote para entrenar es de 8.

El set de datos utilizado corresponde a GoPro [49], el cual es dividido en paquetes de 5 fotogramas consecutivos. Estos paquetes son recortados en piezas de tamaño 256×256 y volteados o rotados en 0, 90, 180 o 270 grados durante entrenamiento, mientras que solo son recortados en piezas de 640×360 en testeo.

4.3.3. Afinación

La red completa es entrenada por 50 épocas utilizando la función de pérdida *L2-norm*, con ADAM como optimizador. La tasa de aprendizaje se fija en 10^{-6} por todo el entrenamiento y el tamaño del lote a entrenar es de 8.

El set de datos usado es ATD-12K [39], el cual es aumentado recortando los fotogramas en tamaño 224×224, invirtiendo el orden, volteando o rotando en 0, 90, 180 o 270 grados durante entrenamiento, mientras que solo son recortados en piezas de 672×1280 durante el testeo.

4.3.4. Ambiente de entrenamiento

Todos los entrenamientos fueron realizados con una GPU Nvidia GeForce RTX 3090 de 24GB con Pytorch como framework de aprendizaje profundo, durando aproximadamente 16 días continuos.

Capítulo 5

FIAI

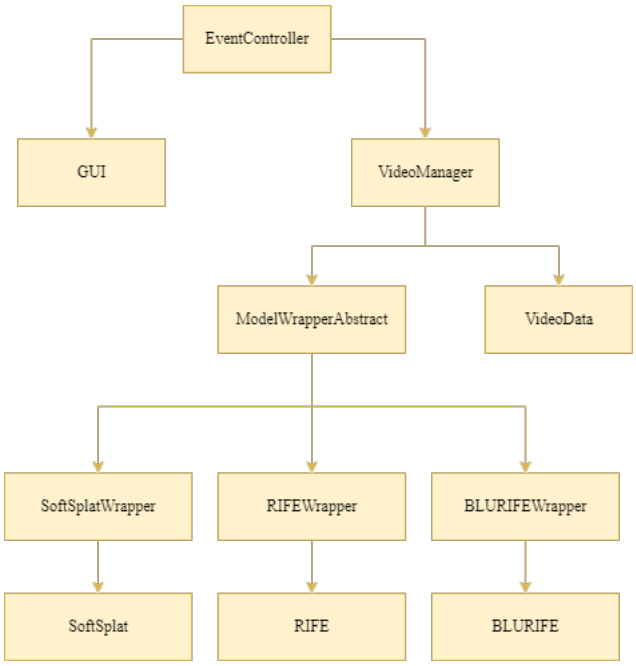


Figura 5.1: Estructura general del prototipo

FIAI (acrónimo de *Frame interpolation Aided Inbetweening*) es un programa prototipo ideado como herramienta de ejemplo sobre la utilización de redes de interpolación para la producción de fotogramas intermedios. Como tal, carece de herramientas relacionadas con edición de video que pueden ser vistas en programas dedicados a animación 2D, optando en cambio por explorar y dar opciones repecto a la creación de inbetweens.

La estructura general se ve en la figura 5.1, la cual posee dos partes importantes que son GUI y procesamiento, las cuales se comunican a través del controlador. Estas serán detalladas más adelante.

El programa fue desarrollado en Python, utilizando OpenCV para manejar los videos y PySide

para crear la interfaz de usuario. Por otro lado, Numpy y Pytorch fueron necesarios para poder realizar operaciones con los fotogramas, en conjunto con otras librerías menos relevantes.

5.1. Controlador

El controlador funciona como mediador entre el GUI y el VideoManager. Se encarga principalmente de crear los mensajes y entregarlos al objeto que corresponda. Para hacer esto, el controlador ejecuta un ciclo el cuál espera a una cola global. Esta almacena tuplas en formato (función, parámetros), que en su totalidad representan un mensaje. Estas tuplas son leídas por el ciclo y ejecutadas, pasando como parámetros adicionales el VideoManager y GUI.

Las funciones disponibles son: Abrir, guardar y cerrar un video, solicitar la interpolación de una sección del video y mostrar o borrar fotogramas.

5.2. GUI

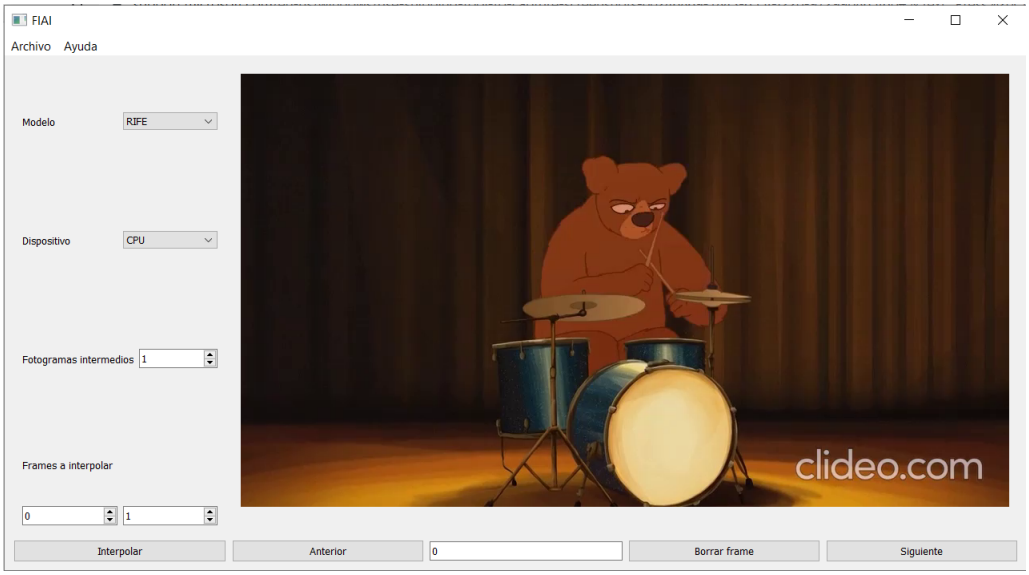


Figura 5.2: Pantalla principal de la aplicación

La interfaz de usuario usada para FIAI fue ideada para ser lo más simple posible, buscando minimizar el ruido visual y maximizar la facilidad de interacción. El resultado es lo que se ve en la figura 5.2. En esta se pueden identificar dos secciones principales: la zona de opciones, que se ubica a la izquierda, y el visor de video en la derecha.

El menú que se ve en la parte superior alberga dos pestañas. La pestaña **Archivo** muestra opciones para abrir, guardar y cerrar un video. Por otro lado, la pestaña **Ayuda** tiene la opción para mostrar un breve manual de usuario en donde se indican las acciones que se pueden realizar con la imagen dispuesta en el visor.

Esta interfaz es estrictamente visual, por lo que no realiza ninguna operación con los fotogramas directamente, aunque si guarda y procesa ciertos datos para mejorar la comunicación con el VideoManager. Estos son repartidos en los componentes de la interfaz, aunque el nombre del archivo en conjunto con el estado del manual de usuario son conocidos a nivel de ventana para determinar rápidamente si los mensajes pueden ser enviados o no.

5.2.1. Zona de opciones

En esta zona cada opción es un objeto el cual guarda los valores seleccionados en él. Estos datos solo son relevantes al momento de enviar el mensaje para interpolar por lo que no es necesario que su acceso sea inmediato.

Dentro de las opciones presentadas se encuentran:

- **Modelo**: Determina el método a utilizar para interpolar. Existen las opciones BLURIFE, RIFE y Softsplat.
- **Dispositivo**: Indica si la interpolación será realizada en memoria RAM o en GPU.
- **Fotogramas intermedios**: Indica la cantidad de imágenes a crear entre un par de fotogramas, hasta un máximo de 10.
- **Fotogramas a interpolar**: Señala el intervalo a interpolar. Esto se entiende como los pares a los cuales se les generarán inbetweens.

Esta zona también dispone del botón interpolar, el cual recopila los datos en ambas partes de la interfaz y los empaqueta en un diccionario antes de enviar la función respectiva.

5.2.2. Visor de video

En esta sección el componente más relevante es el visor de fotogramas. Este no solo se encarga de mostrar los fotogramas al usuario, si no que también da la opción para seleccionar la sección de video a interpolar. Esto se realiza mediante interacción con el ratón: un usuario puede seleccionar la sección de video que desee similar a como hacen otros programas de edición de imágenes, marcando la sección con un rectángulo celeste semi-traslúcido. Cuando se realice la solicitud para interpolar, el componente definirá los bordes con este rectángulo. Si se desea descartar la selección solo basta realizar un clic rápido, lo cuál hará desaparecer el rectángulo e indicará al componente que en caso de enviarse la solicitud de interpolación, que se use la imagen completa.

Los otros componentes en esta sección corresponden al explorador de fotogramas, permitiendo avanzar o retroceder fotograma por fotograma o indicando cuál es el deseado, y el botón para borrar fotogramas.

Como esta parte de la interfaz lidia directamente con el video, los componentes de esta poseen datos relacionados con este, como son la cantidad de fotogramas existentes y dimensiones.

5.3. Procesador

El procesador tiene como función efectuar las operaciones necesarias para realizar la interpolación y modificar el video. Como tal, se divide en múltiples partes para poder efectuar esto.

5.3.1. VideoManager

Se encarga de la comunicación entre las clases relacionadas, ya sea con el controlador o entre VideoData y los ModelWrapper. Esto implica que muchos de sus métodos se dediquen a relegar mensajes a VideoData, como sería la solicitud de fotogramas, datos del video, liberación de RAM, borrar fotogramas, guardar o abrir un archivo. El resto de sus métodos se enfocan en dar las instrucciones para poder interpolar.

La más relevante es *interpolate*, cuyo procedimiento se detalla en el código 5.1.

Código 5.1: Algoritmo para interpolar

```
input: dict
output: frame, n_frame
begin
  model, device, inbetweens, frame_start, frame_end, section ← dict
  changeModel(model)
  changeDevice(device)
  piece, frames ← extractFrames(frame_start, frame_end, section)
  liberateRam()
  interpolated_piece ← interpolate(piece)
  new_frames ← stitchPieceToFrames(interpolated_piece, frames)
  first_new_frame, pos_of_frame ← addFramesToVideo(new_frames)
  return first_new_frame, pos_of_frame
end
```

Para aclarar un poco el pseudocódigo se debe notar que cuando se extraen los fotogramas estos no solo devuelven la sección solicitada para interpolar, sino que también los fotogramas de los que fueron extraídos. De esta forma, cuando la interpolación se haya acabado, las secciones puedan ser unidas nuevamente con los fotogramas para poder obtener una imagen del mismo tamaño que el resto del video.

5.3.2. VideoData

Esta clase está destinada a manejar los datos del video como las dimensiones y cantidad de fotogramas, y ejecutar acciones relacionadas directamente con el archivo, como lo sería la adición, extracción o eliminación de fotogramas.

La principal problemática de representar un video que será modificado es que este pesa demasiado una vez decodificado por lo que guardarlo directamente en memoria RAM no es una estrategia

viable debido a que los modelos hacen uso extensivo de esta, y hacer modificaciones directamente en los videos toma un tiempo considerable. Para solventar esta problemática se estableció que los fotogramas serían guardados en disco y llamados por demanda. Para los ya existentes en los archivos esto implica decodificar solo el indicado, mientras que para los creados se cargan los archivos temporales en donde estaban guardados.

Si bien este enfoque resuelve el problema del uso de RAM, sí produjo el problema del orden de fotogramas. Este viene de que si uno agrega fotogramas en medio de la imagen, se espera que el nuevo ocupe la posición de uno que ya existía, moviendo la posición de este último y todos los que le siguen a un valor mayor. No obstante, como el video original y los fotogramas creados se guardan en disco dentro de archivos distintos, se pierde este orden.

Para resolver esto VideoData guarda un arreglo que representa el proyecto de video, en donde cada elemento consiste en una tupla compuesta de dos valores: el nombre de un archivo de video y el número de un fotograma dentro del archivo. De esta forma, si se quiere obtener un fotograma g del proyecto, solo es necesario leer aquel que es referido en la posición g del arreglo.

Para entender mejor esto considérese que O y S son archivos de video, los cuales están compuestos por fotogramas enumerados en orden del 1 al n y del 1 al m respectivamente, mientras que P es el arreglo del proyecto. Si el arreglo P está estructurado como $P = [(O,1),\ldots,(O,g-1),(S,1),\ldots,(S,m),(O,g),\ldots,(O,n)]$, entonces el fotograma g del proyecto corresponde a $P[g] = (S,1)$, es decir, el primer fotograma del archivo S, por lo que solo debe leerse aquel si es solicitado.

Aún así, esta solución igual implica un gran consumo de RAM si el archivo de video es muy grande. Para poder liberar más memoria entonces se debe considerar que esta información no es necesaria al momento de interpolar, por lo que se guarda en disco hasta que el proceso finalice, donde es cargado nuevamente a memoria principal.

5.3.3. ModelWrappers

Si bien todos los modelos cumplen la misma función e incluso reciben la misma entrada, cómo se recibe esta y cómo funcionan es diferente en ciertas ocasiones, mientras que igual en otras. Para resolver esto se decidió usar un Wrapper abstracto, el cual define las funciones comunes y provee métodos a implementar por las clases que lo hereden.

Las funciones comunes corresponden al método para mover el modelo a un dispositivo, y el constructor de la clase, pues todos los modelos siguen la lógica descrita en 5.2.

Por otro lado, la función para crear fotogramas difiere bastante entre modelos, particularmente entre RIFE y SoftSplat. RIFE (y por extensión BLURIFE) requieren que la entrada tenga una resolución estándar, por lo que se aumentan las imágenes con ceros hasta llegar a la deseada. No

así pasa con SoftSplat que puede trabajar directamente con cualquier par de imágenes.

Código 5.2: Constructor general

```
input: model, device
begin
  loadModel(model)
  sendModelToDevice(device)
  setModelInEvalMode()
end
```

Otra diferencia relevante es que BLURIFE no tiene opción para determinar qué punto del movimiento interpolado usar; siempre asume el medio, por lo que si se necesitan crear más de un dibujo entre fotogramas se debe utilizar el generado como entrada para otros. No sucede así con SoftSplat, que sí provee tal opción.

Capítulo 6

Validación

6.1. Resultados métricos

Como se mencionó en la sección 1.4.1, las medidas a usar son *Learned Perceptual Image Patch Similarity*, *Peak Signal-to-Noise Ratio* y *Structural Similarity Index*, con la definición formal de estas dos últimas siendo las siguientes:

$$PSNR = 10 \cdot \log_{10} \frac{MPV}{MSE} \tag{6.1}$$

$$MSE = \frac{1}{XYC} \sum_{y=1}^{Y} \sum_{x=1}^{X} \sum_{c=1}^{C} [F(x,y,c) - G(x,y,c)]^2 \tag{6.2}$$

$$SSIM = \frac{(2\mu_P\mu_Q + c_1)(2\sigma_{PQ} + c_2)}{(\mu_P^2\mu_Q^2 + c_1)(\sigma_P^2 + \sigma_Q^2 + c_2)} \tag{6.3}$$

Para *PSNR*, *MPV* es el valor máximo para un píxel, X es el ancho de un fotograma, Y es su altura y C es la cantidad de canales de las imágenes F y G.

Por otro lado, en *SSIM* μ_i es el promedio de los píxeles de i, σ_j^2 es la varianza de j, σ_{jk} es la covarianza entre j y k, mientras que c_1 con c_2 son constantes proporcionales al valor máximo de un píxel. P y Q son las imágenes a comparar en escala de grises. Para extender esta medida a imágenes RGB, se toma el promedio de los índices para cada canal.

Para realizar la evaluación se compara BLURIFE con los métodos de interpolación RIFE y SoftSplat en el set de datos ATD-12K. Por completitud, se afinan estos dos modelos con la estrategia mencionada para BLURIFE. Los modelos afinados serán denominados RIFE_ft y SoftSplat_ft respectivamente. Los resultados se encuentran presentes en la tabla 6.1.

Tabla 6.1: Tabla comparativa de métricas en ATD-12K

Modelo	PSNR	SSIM	LPIPS
RIFE	28.56±5.44	0.93±0.04	0.06±0.06
RIFE_ft	28.82±5.32	0.93±0.04	0.07±0.066
SoftSplat	27.83±5.32	0.92±0.04	0.06±0.06
SoftSplat_ft	28.27±4.79	0.93±0.04	0.08±0.06
BLURIFE	28.19±5.03	0.89±0.05	0.09±0.06

Se ve entonces que BLURIFE no presenta una mejora respecto a otros métodos de interpolación, con RIFE_ft siendo aquel que tuvo mejor desempeño. Esto no es apoyado por el hecho que el consumo de memoria de la red crece exponencialmente con el tamaño de la imagen, con fotogramas de 720×1280 requiriendo aproximadamente 16GB de RAM, con un tiempo de ejecución que puede superar el minuto. A pesar de esto, la red creada logra obtener resultados comparables con los métodos que definen el estado de arte de la interpolación de video.

6.2. Resultados cualitativos

(a) Fotograma 0 (b) Fotograma interpolado (c) Fotograma 1

Figura 6.1: Interpolación con BLURIFE

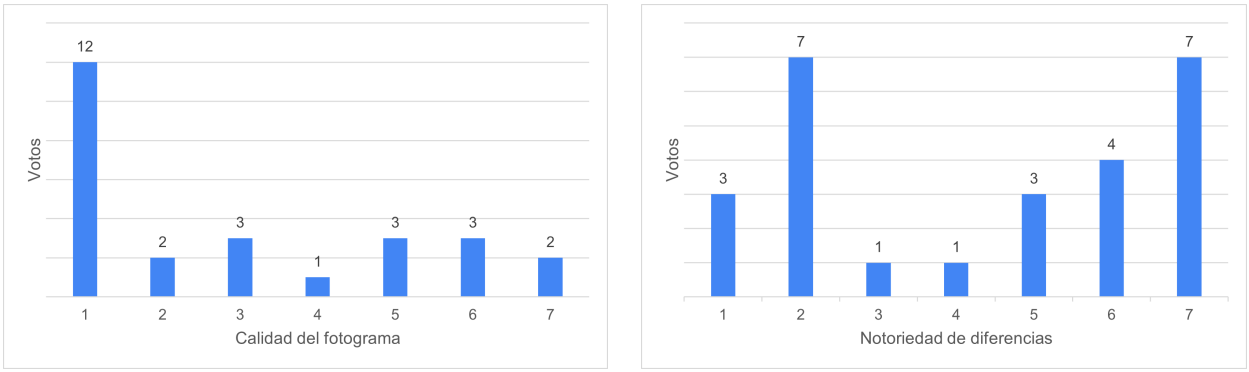
A pesar de la sección de limpiador de BLURIFE, interpolación con esta red aún produce líneas borrosas notables, las cuales se hacen más evidentes mientras mayor sea la diferencia entre fotogramas. Una característica notable es que, en ciertas ocasiones, la red logra interpolar correctamente un movimiento brusco, como se muestra en la figura 6.1, en donde el fotograma interpolado asume con alta confianza la posición del brazo basándose en la entrada, aunque esto no es consistente, como se ve en la figura 6.2

Figura 6.2: Interpolación común para BLURIFE sobre movimientos bruscos

6.2.1. Cuestionario

Se realizó un cuestionario sobre los resultados obtenidos con BLURIFE a 13 estudiantes de Cine, los cuales poseían distintos grados de expertiriencia respecto a la animación 2D. El cuestionario consistió en la presentación de 5 piezas de contenido realizadas con la red las cuales debían ser evaluadas por los participantes y deberían dar su opinión sobre lo mostrado. Tales piezas tanto como el cuestionario se encuentran en el apéndice.

La primera parte del cuestionario buscaba verificar si la calidad de un fotograma interpolado era lo suficientemente bueno como para emular un trabajo hecho manualmente, y cómo se comparaba con el fotograma original. Para lograr esto, se mostraron dos imágenes que mostraban un fotograma real y uno interpolado, sin indicar cuál de ellos era el original.



(a) (b)

Figura 6.3: Resultados parte uno del cuestionario

En las figuras 6.3 los números en los ejes están ordenados de menos a más, por lo que se tiene que las diferencias entre un fotograma hecho manualmente y uno interpolado es bastante notorio, y que la calidad de los creados es pobre.

La segunda parte del cuestionario busca responder cómo funciona la red como interpolador de movimiento, verificando si las transformaciones, movimientos y trayectorias alcanzan los estándares de calidad en animación 2D. Para establecer esto, se mostró a los participantes 3 animaciones cortas que hacían uso amplio de BLURIFE para generar fotogramas.

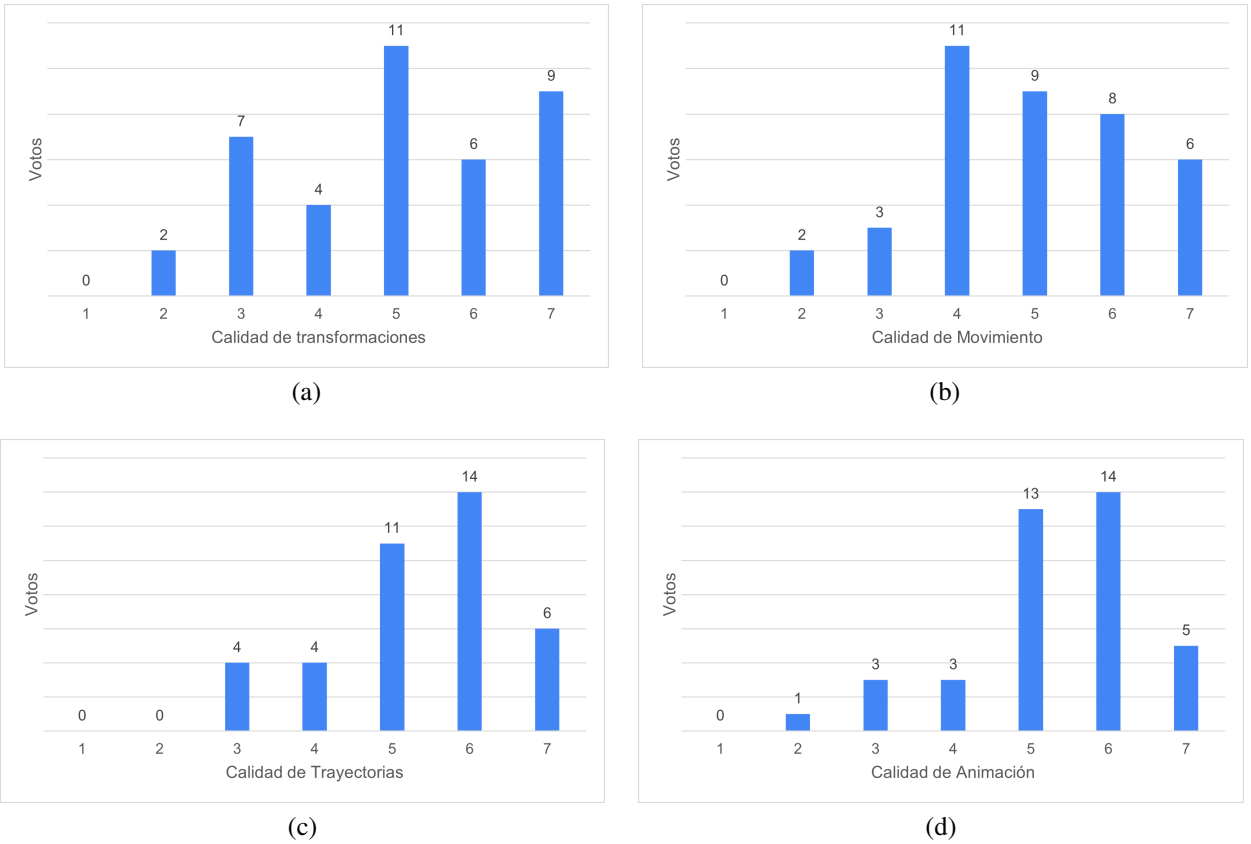


Figura 6.4: Resultados parte dos del cuestionario

Al igual que en la figura anterior, los números en los ejes de la figura 6.4 están ordenados de menos a más, por lo que la red puede producir dinámicas de objetos que son suficientes para crear una animación.

Finalizado el cuestionario los participantes hicieron anotaciones relevantes para el trabajo. La más común es que si bien la red podía generar animaciones con alta calidad en dinámicas de objetos, cuando no lograba producir una buena interpolación y se producían artefactos, estos eran muy notables y rompía la ilusión del movimiento. Este efecto se veía exacerbado si el estilo artístico era muy distintivo, pues el contraste entre los fotogramas incrementaba.

6.3. Pruebas de usuario

Para verificar la usabilidad del prototipo se le pidió a 3 individuos no relacionados con el proyecto y con bajo conocimiento de animación que realizarán ciertas tareas con el programa. Estas fueron ideadas para que el usuario pruebe todas las características del software, las cuales incluyen:

- Abrir, guardar y cerrar archivos
- Explorar el video
- Borrar fotogramas

- Interpolar fotogramas con sus múltiples variantes

El sentimiento general que estuvo presente en las tres pruebas realizadas es la falta de claridad respecto a las capacidades del programa. Esto provocó que ciertas tareas no pudieran ser completadas, provocando frustración en los usuarios.

Otro punto que fue discutido fue la falta de retroalimentación al usuario. Múltiples veces el usuario efectuaba una acción con el programa y se preocupaba pues este se congelaba por un tiempo sin aviso de lo que estaba sucediendo.

Observaciones que también fueron mencionadas es la pobre calidad visual del programa y la falta de herramientas para edición de video, en particular el reproductor, ya que este sirve para verificar la calidad de la animación.

Por último, si bien todos los participantes tuvieron problemas con la aplicación, todos culpaban en parte su propia inexperiencia con animación, y creían que alguien más versado en el tema tendría mejores resultados.

6.4. Discusión

Respecto a objetivo *"Crear una red de interpolación para videos 2D que circunvente los problemas presentes en otras técnicas similares"*, los resultados demuestran que esto no se cumplió pues la borrosidad aun se mantiene. Considerando que BLURIFE utiliza RIFE como subred encargada de generar el fotograma intermedio, se puede descartar a esta sección como la responsable. Esto deja a SSOON como la parte que genera la reducción de calidad.

Esta reducción puede tener como causa múltiples factores. Los más relevantes son listados a continuación:

1. Modificación del traspaso de características

   El método usado por SSOON busca traspasar la mayor cantidad de información extraída del fotograma anterior, a diferencia de SOON [33] que trata de enviar información general de los últimos cinco. Para lograr esto último, se usa una capa especial la cuál prepara los datos para que puedan ser usados de forma más eficiente en la siguiente iteración. Esta capa no se encuentra en SSOON, lo que puede haber afectado el resultado obtenido.

2. Entrenamiento con GoPro

   GoPro [49] es un dataset de videos borrosos. La característica principal de estos es que tal borrosidad es uniforme en toda la secuencia. Debido a esto, SSOON busca perfilar de la misma forma considerando el entrenamiento que tuvo.

3. Profundidad

Si bien SSOON fue diseñada para ser menos detallada que su versión original, al ser unida con RIFE la profundidad de la red completa aumenta considerablemente. Esto, en conjunto con la nula existencia de conexiones entre las capas internas de ambas subredes, puede causar un peor rendimiento, como menciona Xiangyu et al [11].

Para poder determinar que causa el rendimiento obtenido sería necesario realizar experimentación adicional.

A pesar de esto, la red presentó resultados favorables relacionados con la dinámica de objetos. En particular, logra determinar con cierto éxito el movimiento o trayectoria de los objetos entre los fotogramas. Esto se puede atribuir a la afinación con ATD-12K [39], debido a que su construcción fue ideada específicamente para representar distintas formas de movimiento, permitiendo así tener un entrenamiento más robusto.

El objetivo *"Adaptar técnicas de interpolación de movimiento existentes para que tengan un uso más flexible"* fue parcialmente logrado. Si bien se consiguió cierta flexibilidad en el uso de la redes, las limitaciones que provenían de su definición resultaron críticas, pues impedían al usuario definir el tipo de interpolación a realizar como lo hacen los programas tradicionales mencionados en la sección 2.4.2. Es a esta limitación la que se le puede atribuir el rendimiento obtenido en la calidad de transformaciones y movimiento, al ser estos dos aspectos los más afectados.

Respecto al objetivo *"Diseñar e implementar un programa amigable"*, este fue incumplido, en gran parte por la falta de características presentes en otros programas de animación como CACANi[41] o OpenToonz [43].

Finalmente, el objetivo de *"Verificar el uso de técnicas de interpolación de movimiento basadas en aprendizaje profundo como herramientas para el proceso de intermediación"* fue incumplido. Como se mencionó en la discusión del primer objetivo, la presencia de artefactos borrosos se mantiene, y de acuerdo con los resultados del cuestionario, este sigue siendo el problema más relevante.

Basándose en lo discutido en los párrafos anteriores, se puede ver que el uso de un perfilador uniforme no mejora de forma directa el rendimiento de una red dedicada a la interpolación de movimiento pues aún genera líneas borrosas notables. Esto resulta relevante considerando que es la existencia de artefactos visuales lo que impide principalmente el uso de la tecnología en la producción de una animación, con otros aspectos como la calidad de trayectorias en objetos interpolados teniendo buen rendimiento, tomando en cuenta que la evaluación realizada emulaba casos reales.

Si dividimos el proceso de interpolación de video en dos partes, una encargada de generar el movimiento de los objetos, y otra a confeccionar los fotogramas estos, entonces se tiene que para poder utilizar la técnica explorada en este proyecto en animación 2D se debe mejorar la segunda parte sin afectar a la primera. Una vez logrado esto, se debe dar una modificación a los métodos de

interpolación para que den más control a los animadores sobre el movimiento a generar, para poder así tener una herramienta de intermediación que compita con las existentes.

Capítulo 7

Conclusiones

Durante el transcurso de este proyecto se creó la red para interpolación de movimiento BLURIFE y su programa asociado FIAI, ambos con la finalidad de explorar la oportunidad de acelerar el proceso de intermediación con inteligencia artificial. Los resultados obtenidos indican que si bien el trabajo puede tener cierta utilidad, este no es suficiente para cumplir la meta.

El hecho que animaciones con fotogramas interpolados lograran obtener buena calificación en términos de calidad de movimiento verifica que es posible utilizar IA para generar inbetweens de animación 2D, aunque el resto de los resultados denota que aún falta progreso por hacer. En primer lugar, la red creada no cumplió con las expectativas de resolver los problemas que existían en los otros métodos. Por otro lado, si bien la adaptación de técnicas de interpolación da flexibilidad a su uso, este sigue siendo limitado debido al rendimiento visual que tienen las redes en sí. Por último, el programa creado no logra ser lo suficientemente intuitivo para el usuario común, aunque si es posible que lo sea para alguien más experimentado; no obstante, esto requiere de un testeo más extenso.

Los resultados de BLURIFE arrojan que su rendimiento con respecto a medidas utilizadas para comparar resultados de interpolación es menor en comparación a otros métodos que se encuentran en el estado de arte. Esto se puede atribuir a múltiples factores, como lo sería el entrenamiento de la red, el traspaso de características o la profundidad, aunque para confirmar esto se necesita realizar más experimentos. Por otro lado, la red presentó un aprendizaje relacionado con las dinámicas de los objetos presentes en los fotogramas derivado de la afinación realizada con el dataset ATD-12K, aunque cabe mencionar que esto aún no es suficiente para ser usado en cualquier situación como lo demuestran los resultados vistos en el capítulo anterior.

Los resultados de FIAI demuestran que como herramienta para generar inbetweens es insuficiente ya que carece de características necesarias para la edición de video, no obstante sí abre la puerta a la posibilidad de tener una mayor utilidad si es insertado en algún programa destinado a la creación de animaciones en general.

En conclusión, si bien el trabajo efectuado no genera una utilidad relevante para los animadores, si presenta un avance con respecto a la problemática de la producción de inbetweens. Los resultados obtenidos nos permiten vislumbrar qué aspectos de esta problemática pueden ser atacados como lo serían la elaboración de líneas bien definidas y/o las opciones de trayectorias. Por ejemplo, tomando como inspiración lo realizado con BLURIFE, se podría explorar cómo un perfilador que funcione por regiones afectaría los resultados obtenidos con algún método de interpolación ya visto. También sería interesante indagar en la posibilidad de crear un sistema de interpolación que reciba además de los fotogramas una función que denote el movimiento que debería obtener, siendo este último una mejora a los sistemas de interpolación que ya están disponibles para los animadores.

Bibliografía

[1] MacGillivray, C., "How psychophysical perception of motion and image relates to animation practice," pp. 81 – 88, 2007, doi:10.1109/CGIV.2007.48.

[2] Read, P. y Meyer, M.-P., Restoration of Motion Picture Film. Elsevier Science 2014, 2014.

[3] Williams, R., The Animator's Survival Kit–Revised Edition: A Manual of Methods, Principles and Formulas for Classical, Computer, Games, Stop Motion and Internet Animators. Faber Faber, Inc., 2009.

[4] Thomas, F. y Johnston, O., The Illusion of Life: Disney Animation. Abbeville Press, 1981.

[5] Díaz García, M., Análisis de los avances digitales para el desarrollo e integración de la animación tradicional y la animación generada por ordenador en películas históricas. PhD thesis, 2011.

[6] yi Tai, P., "The aesthetics of keyframe animation: Labor, early development, and peter foldes," Animation, vol. 8, no. 2, pp. 111–129, 2013, doi:10.1177/1746847713487815.

[7] Carvalho, L., Marroquim, R., y Vital Brazil, E., "Dilight: Digital light table – inbetweening for 2d animations using guidelines," Computers Graphics, vol. 65, pp. 31–44, 2017, doi:https://doi.org/10.1016/j.cag.2017.04.001.

[8] Li, X., Zhang, B., Liao, J., y Sander, P. V., "Deep sketch-guided cartoon video synthesis," CoRR, vol. abs/2008.04149, 2020, https://arxiv.org/abs/2008.04149.

[9] Yagi, Y., "A filter based approach for inbetweening," CoRR, vol. abs/1706.03497, 2017, http://arxiv.org/abs/1706.03497.

[10] Goodfellow, I., Bengio, Y., y Courville, A., Deep Learning. MIT Press, 2016. http://www.deeplearningbook.org.

[11] He, K., Zhang, X., Ren, S., y Sun, J., "Deep residual learning for image recognition," CoRR, vol. abs/1512.03385, 2015, http://arxiv.org/abs/1512.03385.

[12] Vranjes, M., Rimac-Drlje, S., y Grgic, K., "Locally averaged psnr as a simple objective video quality metric," en 2008 50th International Symposium ELMAR, vol. 1, pp. 17–20, 2008.

[13] Wang, Z., Lu, L., y Bovik, A. C., "Video quality assessment based on structural distortion measurement," Signal Processing: Image Communication, vol. 19, no. 2, pp. 121–132, 2004, doi:https://doi.org/10.1016/S0923-5965(03)00076-6.

[14] Zhang, R., Isola, P., Efros, A., Shechtman, E., y Wang, O., "The unreasonable effectiveness

of deep features as a perceptual metric," 2018.

[15] Bao, W., Lai, W.-S., Ma, C., Zhang, X., Gao, Z., y Yang, M.-H., "Depth-aware video frame interpolation," 2019, doi:10.1109/CVPR.2019.00382.

[16] Huang, Z., Zhang, T., WenHeng, BoxinShi, y Zhou, S., "Rife: Real-time intermediate flow estimation for video frame interpolation," 2020, https://arxiv.org/abs/2011.06294.

[17] Scharstein, D. y Szeliski, R., "A taxonomy and evaluation of dense two-frame stereo correspondence algorithms," International Journal of Computer Vision - IJCV, vol. 47, pp. 7–42, 2000.

[18] Xue, T., Chen, B., Wu, J., Wei, D., y Freeman, W. T., "Video enhancement with task-oriented flow," International Journal of Computer Vision (IJCV), vol. 127, no. 8, pp. 1106–1125, 2019.

[19] Soomro, K., Zamir, A., y Shah, M., "Ucf101: A dataset of 101 human actions classes from videos in the wild," CoRR, 2012.

[20] Meyer, S., Wang, O., Zimmer, H., Grosse, M., y Sorkine-Hornung, A., "Phase-based frame interpolation for video," 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1410–1418, 2015.

[21] Zhai, J., Yu, K., Li, J., y Li, S., "A low complexity motion compensated frame interpolation method," 2005, doi:10.1109/ISCAS.2005.1465738.

[22] Barron, J., Fleet, D., y Beauchemin, S., "Performance of optical flow techniques," International Journal of Computer Vision, vol. 12, pp. 43–77, 1994, doi:10.1007/BF01420984.

[23] Liu, F., Mai, L., y Niklaus, S., "Video frame interpolation via adaptive convolution," 2017, doi:10.1109/CVPR.2017.244.

[24] Liu, F., Mai, L., y Niklaus, S., "Video frame interpolation via adaptive separable convolution," 2017, doi:10.1109/ICCV.2017.37.

[25] Niklaus, S. y Liu, F., "Softmax splatting for video frame interpolation," en IEEE Conference on Computer Vision and Pattern Recognition, 2020.

[26] Amiaz, T., Lubetzky, E., y Kiryati, N., "Coarse to over-fine optical flow estimation," Pattern Recognition, vol. 40, no. 9, pp. 2496–2503, 2007, doi:https://doi.org/10.1016/j.patcog.2006.09.011.

[27] Rueckert, D., Sonoda, L., Hayes, C., Hill, D., Leach, M., y Hawkes, D., "Nonrigid registration using free-form deformations: application to breast mr images," IEEE Transactions on Medical Imaging, vol. 18, no. 8, pp. 712–721, 1999, doi:10.1109/42.796284.

[28] Ronneberger, O., Fischer, P., y Brox, T., "U-net: Convolutional networks for biomedical image segmentation," CoRR, vol. abs/1505.04597, 2015, http://arxiv.org/abs/1505.04597.

[29] Sun, D., Yang, X., Liu, M.-Y., y Kautz, J., "PWC-Net: CNNs for optical flow using pyramid, warping, and cost volume," 2018.

[30] Wang, X., Chan, K. C. K., Yu, K., Dong, C., y Loy, C. C., "EDVR: video restoration with

enhanced deformable convolutional networks," CoRR, vol. abs/1905.02716, 2019, http://arxiv.org/abs/1905.02716.

[31] Zhang, X., Jiang, R., Wang, T., Huang, P., y Zhao, L., "Attention-based interpolation network for video deblurring," Neurocomputing, vol. 453, 2020, doi:10.1016/j.neucom.2020.04.147.

[32] Hu, Y., Chen, Y., Li, X., y Feng, J., "Dynamic feature fusion for semantic edge detection," en Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19, pp. 782–788, International Joint Conferences on Artificial Intelligence Organization, 2019, doi:10.24963/ijcai.2019/110.

[33] "Learn from the past – sequentially one-to-one video deblurring network," Journal of Visual Communication and Image Representation, vol. 78, p. 103159, 2021, doi:https://doi.org/10.1016/j.jvcir.2021.103159.

[34] Carvalho, L., Marroquim, R., y Vital Brazil, E., "Dilight," Computers Graphics, vol. 65, pp. 31–44, 2017, doi:10.1016/j.cag.2017.04.001.

[35] Burtnyk, N. y Wein, M., "Interactive skeleton techniques for enhancing motion dynamics in key frame animation," Commun. ACM, vol. 19, p. 564–569, 1976, doi:10.1145/360349.360357.

[36] Kort, A., "Computer aided inbetweening," en NPAR '02, 2002.

[37] Yang, W., "Context-aware computer aided inbetweening," IEEE Transactions on Visualization and Computer Graphics, vol. 24, no. 2, pp. 1049–1062, 2018, doi:10.1109/TVCG.2017.2657511.

[38] Li, X., Zhang, B., Liao, J., y Sander, P., "Deep sketch-guided cartoon video inbetweening," IEEE Transactions on Visualization and Computer Graphics.

[39] Li, S., Zhao, S., Yu, W., Sun, W., Metaxas, D. N., Loy, C. C., y Liu, Z., "Deep animation video interpolation in the wild," 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 6583–6591, 2021.

[40] Inc, T. B. A., "Harmony," 2005, https://www.toonboom.com/products/harmony.

[41] Limited, C. P., "Cacani," 2016, https://cacani.sg/.

[42] Foundation, K., "Krita," 2005, https://krita.org/.

[43] Dwango, "Opentoonz," 2016, https://opentoonz.github.io/e/.

[44] Tahoma2D, "Tahoma2d," 2020, https://tahoma2d.org/.

[45] Chapman, N. y Chapman, J., Digital Multimedia. Worldwide Series in Computer Science, Wiley, 2000, https://books.google.cl/books?id=Bp9QAAAAMAAJ.

[46] Shi, X., Chen, Z., Wang, H., Yeung, D., Wong, W., y Woo, W., "Convolutional LSTM network: A machine learning approach for precipitation nowcasting," CoRR, vol. abs/1506.04214, 2015, http://arxiv.org/abs/1506.04214.

[47] Hochreiter, S. y Schmidhuber, J., "Long short-term memory," Neural computation, vol. 9,

pp. 1735–80, 1997, doi:10.1162/neco.1997.9.8.1735.

[48] Loshchilov, I. y Hutter, F., "SGDR: stochastic gradient descent with restarts," CoRR, vol. abs/1608.03983, 2016, http://arxiv.org/abs/1608.03983.

[49] Park, D., Kim, J., y Chun, S. Y., "Down-scaling with learned kernels in multi-scale deep neural networks for non-uniform single image deblurring," CoRR, vol. abs/1903.10157, 2019, http://arxiv.org/abs/1903.10157.

Anexos

Anexo A. Detalle ResBlock

El ResBlock utilizado para SSOON viene dado por la figura A.1 y la tabla A.1

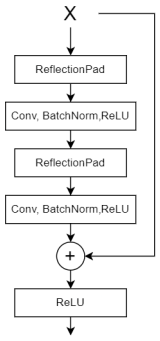


Figura A.1: ResBlock simple

Tabla A.1: Detalle de la ResBlock

Operaciones	Canales	Kernel	Stride	Padding
ReflectionPad	-	-	-	1
Conv2d, BatchNorm, ReLU	256	3	1	0
ReflectionPad	-	-	-	1
Conv2d, BatchNorm, ReLU	256	3	1	0

Anexo B. Cuestionario

El cuestionario se encuentra en el siguiente enlace de Google Forms, con las imágenes utilizadas siendo las siguientes:

Mientras que los videos usados se encuentran en la siguiente lista de reproducción: Encuesta

Anexo C. Prueba de usuario

Las pruebas de usuario consistieron en un cuestionario previo, seguido por la ejecución de las tareas, finalizando con una serie breve de preguntas.

El cuestionario previo consistía en una serie de preguntas enfocadas a determinar los conocimientos de los participantes, determinando que información necesitarían estos para poder usar el programa. Estas preguntas eran:

- ¿Qué carrera estudias o estudiaste?

- ¿Qué conceptos de animación 2D conoces?
- ¿Tiene conocimiento sobre lo que es un inbetween?
- ¿Sabe lo que es un fotograma?

Las tareas a realizar, como se menciona en la sección 6.3, se enfocan en probar las características del software utilizando un video de prueba que se encuentra en el siguiente enlace. Para cada una se midió el tiempo que usaba para completarla o omitirla, marcando si lograba efectuar lo pedido. Estas son:

- Abrir video de prueba
- Mostrar fotograma 32
- Borrar fotograma 33
- Generar 1 fotograma de la imagen completa entre 33 y 34
- Generar 1 fotograma de la imagen completa entre los fotogramas 42 y 43, y entre los fotogramas 43 con 44
- Generar 2 fotogramas de la imagen completa entre 123 y 124
- Generar 1 fotograma del brazo izquierdo del oso entre 121 y 122
- Generar 1 fotograma de la imagen completa entre los punto que desee
- Guardar el video con formato AVI
- Cerrar el programa

Por último, las preguntas finales buscaban recibir retroalimentación sobre el programa por parte de los participantes:

- ¿Qué opinas del programa?
- ¿Encontró complejo descifrar las tareas dadas?
- ¿Qué mejoras le haría usted al programa?