Reconstrucción de Videos y Filtro de Cámara Lenta mediante Interpolación Polinomial

Silvio Vileriño *(svilerino@gmail.com)*, Sacha Kantor *(sacha.kantor+exactas@gmail.com)*
Juan Grandoso *(juan.grandoso@gmail.com)* y Nahuel Lascano *(laski.nahuel@gmail.com)*

Resumen

La elaboración de órdenes es uno de los problemas más frecuentes que se da en nuestra sociedad. Virtualmente todos los ámbitos conocidos (deportes, laboral, lúdico, salarial, etc) enfrentan este problema, que será más o menos complejo según el contexto. A lo largo de este trabajo estudiamos este problema en el ámbito de las páginas web: ¿cuándo una página debería estar por encima de otra?

Palabras Clave

PageRank, Autovectores, Matriz Estocástica, Motores de Búsqueda

Índice

1. Introducción Teórica

En el siguiente trabajo se aborda el desafío de aumentar algorítmicamente la cantidad de frames de un video de manera que el resultado se asemeje al video original. En otras palabras, el objetivo es, a partir de un video,

generar otro con mayor cantidad de frames, de modo tal que coincidan en aquellos frames presentes en el video original y los frames generados se *ajusten* a estos, apuntando idealmente a que el ojo humano no perciba el agregado artificial.

En esta introducción presentaremos una lista no exhaustiva de las diversas situaciones que motivan la resolución de dicho problema y daremos un marco teórico a los métodos propuestos para su solución.

1.1. La motivación

1.1.1. Compresión de video

El crecimiento exponencial de internet ha dado lugar, entre otras cosas[**TP2**], a la mejora de la infraestructura utilizada, lo que en particular repercutió en un aumento generalizado de las velocidades de conexión y del ancho de banda de las mismas. Esto promovió su utilización para compartir contenido cada vez más pesado, en particular videos de todo tipo, desde caseros a profesionales. Además, de la mano con el avance de las tecnologías de captura de video, la resolución de los mismos aumenta cada vez más.

Sin embargo, la inmensa cantidad de usuarios impone un límite al ancho de banda que se le puede dedicar a cada uno, sobretodo para sitios populares como YouTube que sirven miles de videos en cada instante determinado, e impone la necesidad de criterios para reducir la cantidad de paquetes que se le transfieren a cada usuario.

Un abordaje común y ampliamente difundido es la compresión de videos, con o sin pérdida de calidad. En términos generales, consiste en que el servidor envíe una versión comprimida del video (posiblemente precomputada de antemano) y que el usuario use su propio poder de cómputo para descomprimirlo y visualizarlo. De este modo se reduce la cantidad de tráfico en la red a costa de un trabajo mayor de CPU de servidores y usuarios, que en general resulta menos costoso.

Los resultados de este trabajo pueden utilizarse como método de compresión con pérdida. Visto de ese modo, una versión *comprimida* de un video es un nuevo video con un subconjunto de los frames del original. El mecanismo de compresión, entonces, resulta muy sencillo de implementar. Para realizar la descompresión se precisa, entonces, generar los frames faltantes a partir de los recibidos. El objetivo de este trabajo es el estudio de distintos métodos para resolver ese problema.

Pero la compresión de videos no es la motivación principal de los métodos estudiados. Para dicho problema existen variados algoritmos que, sin eliminar cuadros completos, representan cada uno en función de los cambios respecto de los anteriores, obteniendo resultados más fieles (dado que no eliminan por completo la información de ningún cuadro) sin un tamaño mucho mayor[**wiki˙data˙compression˙video**].

1.1.2. Reproducción en cámara lenta

Otra motivación posible y más generalizada resulta de analizar las tecnologías de captura de video. Desde su invención, el principio básico se mantuvo intacto: capturar varias imágenes por segundo y reproducirlas en orden para dar al ojo humano la sensación de movimiento. Un video, entonces, no es más que una secuencia de imagenes (en adelante *frames*) reproducidas a una frecuencia determinada, en general mayor a 12 por segundo (el máximo que el sistema visual humano puede percibir como imágenes separadas[**wiki˙framerate**]). En la época del cine mudo las películas se filmaban con cámaras manuales, lo cual permitía alterar la cantidad de frames por segundo (en adelante *frame-rate*) según la velocidad que se le quisiera dar a la escena: a mayor frame-rate la escena se percibe más lenta y viceversa. Pero al añadirles sonido fue necesario estandarizar el frame-rate, pues el oído humano es mucho más sensible a cambios de frecuencia que el ojo[**wiki˙framerate**]. Desde entonces el estándar ha sido filmar y reproducir a (aproximadamente) 24 cuadros por segundo, tanto películas como demás videos, lo cual se mantuvo prácticamente intacto hasta 2012 con la llegada del Cine en Alta Frecuencia (*HFR* por sus siglas en inglés) de la mano de Peter Jackson en *The Hobbit: An Unexpected Journey*.

A lo largo de la historia y cada vez con mayor frecuencia se han utilizado Cámaras de Alta Velocidad (*HSC*) para generar videos que, reproducidos a 24 *fps*[a] permitan percibir cosas que una cámara normal e incluso el ojo humano no percibiría. Los usos de los mismos son muy variados, y van desde la biomecánica[b] hasta los eventos deportivos[c], pasando incluso por meras curiosidades[d]. Sin embargo los videos resultantes son sumamente pesados

a. frames por segundo, unidad estándar del frame-rate
b. https://www.youtube.com/watch?v=VSzpM8vEAFA
c. https://www.youtube.com/watch?v=O0lCJfFtjCQ
d. https://www.youtube.com/watch?v=tw3q4_jZv8M

por unidad de tiempo, haciéndolos complicados de almacenar, transportar y distribuir. Además, el equipo necesario para realizar las capturas suele ser mucho más costoso que el equipamiento normal. O quizás simplemente no se cuenta con una versión en alta velocidad de un video ya filmado.

Dicho en líneas más generales, es posible que se desee reproducir en cámara lenta un video del cual, por el motivo que fuere, solo se tiene una versión con frame-rate estándar. Una solución es generar computacionalmente los frames faltantes, aprovechando la información existente para crear frames que se acerquen lo más posible a los que hubiese producido una HFC. Lo cual nos lleva nuevamente al objeto de estudio de este trabajo.

Esta es, en particular, la motivación sobre la que más hincapié haremos en el resto del trabajo, más allá de que los mismos métodos pueden ser usados para atacar cualquiera de los problemas que describimos en esta sección.

1.1.3. Suavización de video y Morphing

También es posible que lo que se busque sea generar frames nuevos a partir de un video ya existente pero no con la intención de verlo en cámara lenta sino para que el resultado final sea más "suave" o agradable a la vista. Es un proceso común en los videos animados dibujados a mano[e], dado que el trabajo extra necesario para dibujar cada frame individualmente difícilmente sea apreciado por el espectador final. Hoy en día se utiliza también en animaciones por computadora, por ejemplo para realizar una trancisión fluida entre dos expresiones de una cara o entre dos estados posibles de un cuerpo 3D.

Un objetivo similar es generar una trancisión entre dos fotos o videos que no necesariamente forman parte de umna misma captura, pero que se quiere integrar en un único y, en lo posible, fluido video. Los usos más comunes del *morphing*, como se le llama, consisten en transformar la cara de una persona en la de otra[f].

1.2. Marco teórico

Nos interesa, entonces, transformar videos computacionalmente para que se perciban más lentamente, que el resultado final se vea "fluido" y se acerque lo más posible a lo que se habría capturado usando una Cámara de Alta Velocidad. Lo primero que hace falta es modelar los videos de un modo que nos permita manejarlos usando lenguajes de programación conocidos, sin incluir herramientas de edición complejas que exceden al alcance de este trabajo. Usamos entonces que un video, en su forma "original" (sin compresión) es un conjunto ordenado de imágenes, cada una de la cual representa un frame. En consecuencia, el problema consiste en generar nuevas imágenes "intermedias" para que, al reproducir el video con el mismo frame-rate, se perciba más lento. A esto último lo conoceremos como el efecto de *slowmotion*.

Cada imagen, a su vez, se puede modelar como una matriz de píxeles. Para simplificar el análisis, consideraremos todas las imágenes (y por lo tanto los videos) únicamente en escala de grises[a]. De este modo, un píxel se puede representar con entero entre 0 y 255 inclusive (un byte) que denota la cantidad de luz que hay en ese píxel particular (siendo 0 el negro absoluto y 255 el blanco absoluto).

La generación de estas nuevas imágenes intermedias puede hacerse de diversas maneras. Una de ellas, aplicada en este trabajo, implica generarlas píxel por píxel, utilizando la información que nos brindan los píxeles correspondientes de las imágenes cercanas en el tiempo. Más formalmente, para cada píxel p de cada frame f a generar tomamos como información los píxeles que están en la misma posición que p en las imágenes cercanas a f en el tiempo.

Se nos presenta el problema de cómo utilizar esa información para generar un píxel que resulte en un video final con las propiedades deseadas. En este trabajo estudiamos diferentes métodos y los comparamos estableciendo métricas cualitativas y cuantitativas. Pero para introducir el detalle de los métodos hace falta hacer algunas definiciones previas.

1.2.1. Interpolación

Dado un conjunto de puntos en $\mathbb{R}^2 (x_0, y_0), \ldots, (x_n, y_n)$ con $x_i \neq x_j$ si $i \neq j$, decimos que una función $f : \mathbb{R}^2 \to \mathbb{R}^2$ interpola dichos puntos si $f(x_i) = y_i$ para todo $i = 0, \ldots, n$. En particular, si nos restringimos a considerar polinomios, se puede demostrar que dados $n + 1$ puntos como los descritos existe un único polinomio $P \in \mathbb{R}[x]$

e. De hecho el primer video animado de la historia, Fantasmagorie de 1908, tiene solo la mitad de sus cuadros realmente dibujados.
f. Como se puede ver en este excelente ejemplo: https://www.youtube.com/watch?v=3ZHtL7CirJA
a. Numerical representation: https://en.wikipedia.org/wiki/Grayscale

de grado menor o igual que n tal que los interpola[[**wiki˙lagrange˙polynomial**]]. A este polinomio se lo conoce como *Polinomio Interpolador de Lagrange*, y su fórmula esta dada de la siguiente manera:

$$P(x) = \sum_{k=0}^{n} \left(y_k \prod_{i \neq k}^{n} \frac{x - x_i}{x_k - x_i} \right)$$

Sin embargo, los polinomios interpolantes tienen la desventaja de que cuando el n es grande *oscilan* demasiado. Es por esto que en general se utiliza la técnica de *Interpolación segmentada*, que consiste en desarrollar la función interpolante f de a partes tomando subconjuntos de puntos consecutivos, generando el poliniomio interpolante de cada subconjunto y luego "conectando" cada uno de estos en el orden adecuado.

Existen diversos tipos de Interpolación segmentada, entre ellos la lineal, cuadrática y cúbica, nombrados según el grado de cada polinomio interpolante. Para la aplicación del efecto de *slowmotion*, se utilizará la lineal y cúbica. A continuación se dará una breve explicación de estos dos métodos de interpolación.

1.2.2. Lineal

El caso más sencillo es la **Interpolación segmentada lineal**: construimos $S_0, \ldots, S_{n-1}$ polinomios tal que S_i interpola x_i y x_{i+1} y definimos

$$f(x) = \begin{cases} S_0(x) & \text{si } x \in [x_0, x_1] \\ \vdots & \\ S_{n-1}(x) & \text{si } x \in [x_{n-1}, x_n] \end{cases}$$

como la función interpolante final. Notar que, por ser cada S_i polinomio interpolante de dos puntos, cada S_i es de grado a lo sumo 1, con lo cual es simplemente una recta.

La interpolación segmetada lineal, no obstante, posee el problema de no resultar "suave" geométricamente, es decir, no es derivable en los puntos considerados. Esto, sumado a problemas diferentes que trae la utilización de polinomios cuadráticos, nos motiva a usar polinomios cúbicos, dando lugar a la técnica conocida como *splines*.

1.2.3. Splines (Interpolación cúbica)

Al igual que en la Interpolación lineal, se procede a construir $S_0, \ldots, S_{n-1}$ polinomios donde cada uno de ellos interpola un segmento equiespaciado sobre x_i y x_{i+1} ($i = 0, \ldots, n$). La diferencia es que dichos polinomios son de grado 3. Por conveniencia, vamos a considerar que son de la forma

$$S_i(x) = a_i + b_i(x - x_i) + c_i(x - x_i)^2 + d_i(x - x_i)^3$$

Se desea obtener la tupla de coeficientes (a_i, b_i, c_i, d_i) para $i = 0, \ldots, n$. Para ello se utiliza una serie de condiciones que, de cumplirse, mejorarían el problema descrito de la interpolación lineal. Enumeramos estos requisitos, primero, desde una perspectiva más conceptual y no tan formal:

1. Los polinomios S_i deben pasar por los conjuntos de puntos (x_i, y_i) que se establecieron como datos iniciales. Es decir, asegurar que la función resultante sea continua e interpole a los puntos originales.
2. Se debe mantener también la continuidad en las derivadas primeras y segundas. En otras palabras, por cada par S_i, S_{i+1} con $i = 0, \ldots, n-2$ se debe cumplir la igualdad de la derivada primera y segunda de dichos polinomios en el punto en que se conectan (x_{i+1}).
3. Para la última restricción se consideran dos opciones:
   a) La derivada segunda en los bordes de la función debe ser nula cuando se evalua en el x_0 y x_n, respectivamente. Se la define como *Spline natural*.
   b) La derivada primera en los bordes coinciden con el de la función original que se está aproximando en el punto x_0 y $x + n$, respectivamete. Esto implica que se necesita de cierta información extra sobre la función a interpolar. Por esta razón, se la conoce como *Spline sujeto a la función interpolada*.

Formalmente, todo esto se expresa como[a]

1. $S_i(x_i) = y_i$ para todo $i = 0, \ldots, n-1$, y $S_{n-1}(x_n) = y_n$
2. $S_i(x_{i+1}) = S_{i+1}(x_{i+1})$ para todo $i = 0, \ldots, n-2$
3. $S_i'(x_{i+1}) = S_{i+1}'(x_{i+1})$ para todo $i = 0, \ldots, n-2$
4. $S_i''(x_{i+1}) = S_{i+1}''(x_{i+1})$ para todo $i = 0, \ldots, n-2$
5. Se cumple que
   a) $S''(x_0) = S''(x_n) = 0$ (spline natural)
   b) $S'(x_0) = f'(x_0)$ y $S'(x_n) = f'(x_n)$ (spline sujeto)

Y definimos entonces al *spline* como el conjunto de todas las funciones cúbicas S_i:

$$f(x) = \begin{cases} S_0(x) & \text{si } x \in [x_0, x_1] \\ \vdots & \\ S_{n-1}(x) & \text{si } x \in [x_{n-1}, x_n] \end{cases}$$

A pesar de su mejora en ciertos aspectos, como la suavidad del *spline*, la cantidad de cálculos a realizar es mayor, lo cual nos lleva a un problema en el ámbito computacional. Estudiemos entonces cómo encontrar los coeficientes.

Recordemos que estamos considerando polinomios de la forma

$$S_i(x) = a_i + b_i(x - x_i) + c_i(x - x_i)^2 + d_i(x - x_i)^3$$

Observemos que $S_i(x_i) = a_i$. Usando la condición 1, tenemos

$$a_i = y_i \text{ para todo } i = 0, \cdots, n-1$$
$$a_{n-1} + b_{n-1}(x_n - x_{n-1}) + c_{n-1}(x_n - x_{n-1})^2 + d_{n-1}(x_n - x_{n-1})^3 = y_n$$

Combinando las condiciones 1 y 2 obtenemos $S_i(x_{i+1}) = y_{i+1} = a_{i+1}$. Entonces

$$a_i + b_i(x_{i+1} - x_i) + c_i(x_{i+1} - x_i)^2 + d_i(x_{i+1} - x_i)^3 = a_{i+1} \text{ para todo } i = 0, \cdots, n-2$$

Observar que $S_i'(x) = b_i + 2c_i(x - x_i) + 3d_i(x - x_i)^2$. Como $S_{i+1}'(x_{i+1}) = b_{i+1}$ entonces la condición 3 equivale a

$$b_i + 2c_i(x_{i+1} - x_i) + 3d_i(x_{i+1} - x_i)^2 = b_{i+1} \text{ para todo } i = 0, \cdots, n-2$$

Observar que $S_i''(x) = 2c_i + 6d_i(x - x_i)$. Como $S_{i+1}''(x_{i+1}) = 2c_{i+1}$ entonces la condición 4 equivale a

$$2c_i + 6d_i(x_{i+1} - x_i) = 2c_{i+1} \text{ para todo } i = 0, \cdots, n-2$$

Finalmente, si el spline es natural (la versión que utilizaremos en este trabajo, por desconocer la derivada de la función que genera los puntos), la condición 5 equivale a

$$2c_0 = 0$$
$$2c_{n-1} + 6d_{n-1}(x_n - x_{n-1}) = 0$$

Definamos $a_n = y_n$, $c_n = 0$ y $h_i = x_{i+1} - x_i$. Entonces las ecuaciones son

1. $a_i = y_i$ para todo $i = 0, \ldots, n$
2. $a_i + b_i h_i + c_i h_i^2 + d_i h_i^3 = a_{i+1}$ para todo $i = 0, \ldots, n-1$
3. $b_i + 2c_i h_i + 3d_i h_i^2 = b_{i+1}$ para todo $i = 0, \ldots, n-2$
4. $2c_i + 6d_i h_i = 2c_{i+1}$ para todo $i = 0, \ldots, n-1$
5. $c_0 = 0$

a. El siguiente desarrollo formal fue obtenido y adaptado del **Apunte de Métodos Numéricos** de **Guido Tagliavini Ponce** accesible en https://github.com/CubaWiki/MetNum-ApunteFinal-gtagliavini/raw/master/notas.pdf

De la ecuación 4 despejamos

$$d_i = \frac{c_{i+1} - c_i}{3h_i}$$

Sustituyendo esto último y la ecuación 1 en la ecuación 2 y despejando

$$\begin{aligned} b_i &= \frac{1}{h_i}\left(y_{i+1} - y_i - c_i h_i^2 - d_i h_i^3)\right) \\ &= \frac{1}{h_i}\left(y_{i+1} - y_i - c_i h_i^2 - \frac{(c_{i+1} - c_i)h_i^3}{3h_i}\right) \\ &= \frac{y_{i+1} - y_i}{h_i} - c_i h_i - \frac{1}{3}(c_{i+1} - c_i)h_i \\ &= \frac{y_{i+1} - y_i}{h_i} - \frac{h_i}{3}\left(2c_i + c_{i+1}\right) \end{aligned}$$

Sustituyendo en la ecuación 3 y despejando

$$\begin{aligned} 0 &= b_i - b_{i+1} + 2c_i h_i + 3d_i h_i^2 \\ &= \frac{y_{i+1} - y_i}{h_i} - \frac{h_i}{3}\left(2c_i + c_{i+1}\right) - \frac{y_{i+2} - y_{i+1}}{h_{i+1}} + \frac{h_{i+1}}{3}\left(2c_{i+1} + c_{i+2}\right) \\ &+ 2c_i h_i + 3\frac{c_{i+1} - c_i}{3h_i}h_i^2 \\ &= \left[\frac{y_{i+1} - y_i}{h_i} - \frac{y_{i+2} - y_{i+1}}{h_{i+1}}\right] - \frac{2}{3}h_i c_i - \frac{1}{3}h_i c_{i+1} + \frac{2}{3}h_{i+1}c_{i+1} + \frac{1}{3}h_{i+1}c_{i+2} \\ &+ 2h_i c_i + h_i c_{i+1} - h_i c_i \\ &= \left[\frac{y_{i+1} - y_i}{h_i} - \frac{y_{i+2} - y_{i+1}}{h_{i+1}}\right] + \frac{1}{3}h_i c_i + \frac{2}{3}(h_i + h_{i+1})c_{i+1} + \frac{1}{3}h_{i+1}c_{i+2} \end{aligned}$$

Equivalentemente

$$h_i c_i + 2(h_i + h_{i+1})c_{i+1} + h_{i+1}c_{i+2} = 3\left[\frac{y_{i+2} - y_{i+1}}{h_{i+1}} - \frac{y_{i+1} - y_i}{h_i}\right]$$

Esta ecuación, que vale para $i = 0, \cdots, n-2$, contiene toda la información de las demás (pues la obtuvimos a través de sustituciones sucesivas). Juntando estas $n-1$ ecuaciones con $c_0 = 0$ y $c_n = 0$ tenemos un sistema de $n+1$ ecuaciones y $n+1$ incógnitas

$$\begin{pmatrix} 1 & 0 & 0 & 0 & & & \\ h_0 & 2(h_0 + h_1) & h_1 & 0 & & & \\ 0 & h_1 & 2(h_1 + h_2) & h_2 & & & \\ & & & \ddots & & & \\ & & & & h_{n-2} & 2(h_{n-2} + h_{n-1}) & h_{n-1} \\ & & & & 0 & 0 & 1 \end{pmatrix}$$

Esta matriz es estrictamente diagonal dominante (pues $h_i = x_{i+1} - x_i > 0$), por lo cual es inversible[**properties˙of˙diagonally˙d**]. Luego, la solución es única: existe un único spline natural para $x_0, \cdots, x_n$, la cual se puede encontrar utilizando métodos de resolución de sistemas de ecuaciones como eliminación gaussiana[**TP1**].

1.3. Error

Para establecer la correctitud de nuestros métodos, hará falta utilizar la noción de *Error Cuadrático Medio* (ECM) y *Peak to Signal Noise Ratio* (PSNR). Básicamente, dado un video de $m \times n$ (alto $\times$ ancho) con frames originales F y uno con la misma resolución con frames generados artificialmente $\bar{F}$, definimos

$$\texttt{ECM}(F, \bar{F}) = \frac{1}{mn} \sum_{i=1}^{m} \sum_{j=1}^{n} |F_{k_{ij}} - \bar{F}_{k_{ij}}|^2$$

$$\texttt{PSNR}(F, \bar{F}) = 10 \log_{10} \left(\frac{255^2}{\texttt{ECM}(F, \bar{F})} \right)$$

Como forma de intuición, notemos que el ECM, como su nombre lo indica, es un promedio de la diferencia de todos los píxeles de dos frames al cuadrado. Asimismo, el PSNR es el una medida de cuánto se acerca el error al máximo posible (255^2) medido en escala logarítmica.

2. DESARROLLO

Finalizada la enumeración de los problemas que motivan la investigación sobre el tema y la adición de una breve explicación del problema a trabajar y de la justificación teórica detrás del método de interpolación, nos dedicaremos a desarrollar con mayor precisión cada método número en cuestión. Recordemos que estos métodos intentan atacar el problema de construir, a partir de un video filmado en tiempo real, nuevos frames que den la sensación de que el video original ha sido ralentizado (efecto de *slowmotion*), o, aún mejor, filmado con una cámara de alta frecuencia.

Aclaraciones útiles para la comprensión del desarrollo

Antes de comenzar con la explicación de los métodos queremos definir ciertas frases o palabras claves que ocuparán terreno en el resto de la sección:

- *Frame original*: Cuadro que pertenece al video original, es decir, aquel que no contiene los frames agregados que conciben al efecto de cámara lenta.
- *Frame artificial*: Cuadro que se realizará a partir del procedimiento que cada método vaya planeando. Siempre tiene a cierta distancia, tanto a derecha como izquierda, frames originales.
- *Cantidad de frames a adjuntar*: Entero que indica cuántos cuadros artificiales le adicionaremos entre cada par de frames originales. Lo denotaremos con las letras fr.
- *Matriz frame*: Aunque no se hablará de este nombre en particular, es fundamental aclarar que en cualquier comentario acerca de los valores o píxeles de un frame, se lo está modelando como una matriz de $m \times n$ (resolución del video) que lleva los valores del 0 al 255 inclusive (escala de grises).
- *Píxel*: Transladamos esta palabra asociada a la imagen digital al campo del álgebra lineal. Por lo tanto, se lo considerará como un componente de la matriz frame cuyos valores estan restringido en el rango $[0; 255]$.

Para la explicación de todos los métodos consideramos dado (como parámetros del problema) el video sobre el cual trabajar (con su resolución $m \times n$) y la cantidad fr de frames a generar entre cada par de frames del original.

Además, dado $t \in \mathbb{R}$ definiremos lo siguiente para ayudarnos en las definiciones más formales:

$$
\begin{aligned}
& original(t) \text{ sii } t \text{ coincide con el instante de tiempo de un frame original} \\
original(t) \Rightarrow\ & frame(t) = \text{el frame original del instante } t \\
\neg original(t) \Rightarrow\ & anterior(t) = \max_k(original(k) \wedge k < t) \\
\neg original(t) \Rightarrow\ & siguiente(t) = \min_k(original(k) \wedge k > t) \\
\neg original(t) \Rightarrow\ & resto(t) = t - anterior(t) \\
0 \leq i < m \wedge 0 \leq j < n \Rightarrow\ & f[i, j] = \text{el valor de brillo de píxel en la posición } (i, j) \text{ para un frame } f
\end{aligned}
$$

2.1. Cuadro más cercano

Este método se basa en una idea simplista, muy sencilla de implementar, y que nos permitirá tener una base para establecer comparaciones con métodos más "inteligentes". A pesar de esto, puede resultar precisa para ciertos tipos de videos, como por ejemplo, la filmación de un objeto inmóvil.

Intuición

Como lo indica el subtítulo de la sección, la idea del método es que cada frame artificial es una copia de su frame original más cercano en términos temporales. Para cada frame a generar el método calcula cuál es dicho "vecino más cercano" y luego copia la imágen de dicho frame al frame artificial a generar.

De esta manera, habrá al menos $fr/2$ copias de cada frame original. En caso que se decida agregar una cantidad impar, se opta por fragmentar en dos partes de $fr/2$ y $fr/2 + 1$ cuadros. En la figura 1 vemos un ejemplo conciso de lo explicado. La cantidad de cuadros a agregar es de $fr = 22$, por lo que se lo divide en dos particiones de 11 frames artificiales cuya imagen corresponderá al frame original más cercano (el marrón o blanco).

Volviendo al análisis del video en su totalidad, se repite el anterior procedimiento para cada par de frames en el orden establecido por la secuencia del video. De tal forma se obtienen los frames artificiales, consiguiendo una nueva filmación con el efecto de *slowmotion* que queríamos.

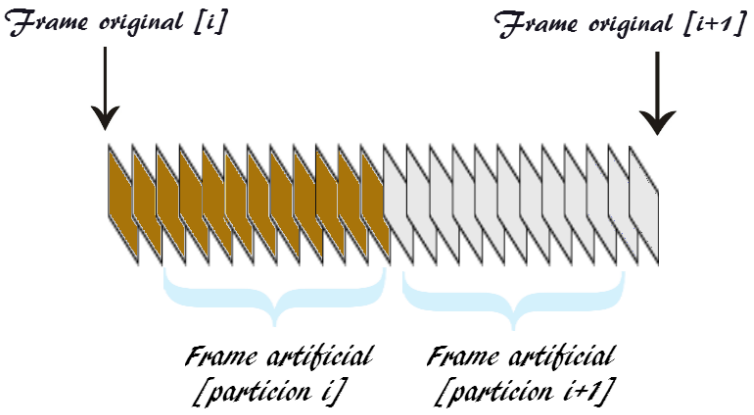


Figura 1. Ejemplo de vecino más cercano

Perspectiva matemática

¿Qué relación tiene este método con el concepto de interpolar? A pesar de ser muy sencillo, este método está usando cierta forma de interpolación. Si consideramos cada píxel (i, j) del video a lo largo del tiempo, obtenemos para cada uno una lista de pares $f_{i,j}(t) = y$ donde t es el instante de un frame original y y el valor de brillo del píxel (i, j) de dicho frame. El método propuesto interpola dichos puntos usando

$$f_{i,j}(t) = \begin{cases} frame(t)_{i,j} & \text{si } original(t) \\ frame(anterior(t))_{i,j} & \text{si } resto(t) \leq fr/2 \\ frame(siguiente(t))_{i,j} & \text{si } resto(t) > fr/2 \end{cases}$$

Es fácil ver que la función interpola efectivamente los puntos $f_{i,j}(t) = y$, aunque lo hace de modo "pobre": para cada par de frames originales tiene una discontinuidad entre ellos a distancia $fr/2$, que será percibida como un "salto" en las imágenes del video. El único caso en que esto no ocurre es cuando los frames considerados son idénticos, por lo que arriesgamos que es el caso en que el método mejor se comportará.

2.2. Interpolación lineal

Compartiendo con *vecino más cercano* la idea de trabajar píxel a píxel con cada par de frames para eventualmente obtener el video deseado con su respectivo efecto, este método puede propocionar ciertas mejoras a la hora de trabajar con videos con movimiento[a].

Intuición

El método se centra en tomar crear un interpolador lineal segmentado por cada píxel del video, cuyos puntos interpolados son los valores de brillo del píxel en los frames originales. Por ende, tendremos por cada par de frames originales $m \times n$ polinomios de grado uno, que usaremos para conocer los valores intermedios entre los dos cuadros originales y gererar así los píxeles de los frames artificiales.

En la figura 2 se puede observar el efecto producido por una interpolación de este estilo.

Perspectiva matemática

Nuevamente consideramos para cada píxel una lista de pares $f(t) = y$. Aplicando lo visto en la Introducción teórica, para cada píxel construimos $S_0, \ldots, S_{n-1}$ polinomios de grado a lo sumo 1 (con n cantidad total de frames del video) tal que S_p interpola $t_p = y_p$ y $t_{p+1} = y_{p+1}$ para cada t_p instante de un frame original. Luego, definimos la siguiente función:

a. Que, entendemos, es el caso de la mayoría de los videos, exceptuando quizás los subidos a YouTube para compartir música.

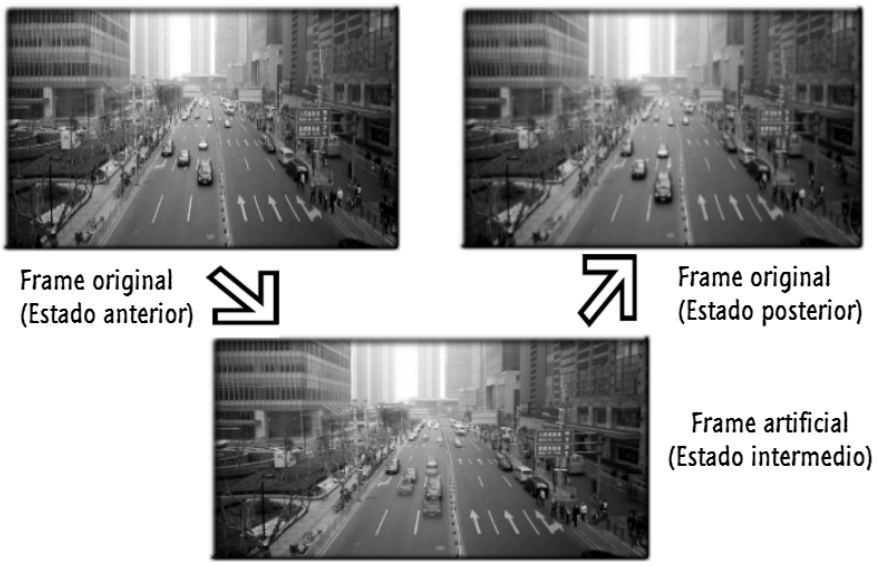


Figura 2. Muestra de interpolación lineal con un solo frame intermedio

$$f(t) = \begin{cases} S_0(t) & \text{si } t \in [t_0, t_1] \\ \vdots & \\ S_{n-1}(t) & \text{si } t \in [t_{n-1}, t_n] \end{cases}$$

Como ya dijimos, habrá una de estas $f_{i,j}$ por cada píxel (i, j), pero nos restringimos a definirla para un único para evitar la notación $_{i,j}$ en todas las ecuaciones.

Cada S_p se define como la siguiente recta:

$$S_p(t) = y_p + (y_{p+1} - y_p) * \frac{t - t_0}{t_1 - t_0}$$

Donde $y_p = frame_p[i, j]$, $y_{p+1} = frame_{p+1}[i, j]$, $t_0 = anterior(t)$ y $t_1 = siguiente(t)$. Es importante aclarar que siempre se debe evaluar esta función dentro del rango (t_0, t_1), es decir, los puntos intermedios.

Para una mayor comprensión, representaremos gráficamente el método. Supongamos que tenemos el algoritmo que instancia una función S_p, tomando la información de un par de píxeles de igual posición (por ejemplo $(1, 2)$) en $frame_0$ y $frame_1$, y la evalúa en fr puntos para generar píxeles artificiales. Asumiendo $fr = 2$, observamos en la figura 3 el resultado.

Si lo usamos considerando ahora múltiples frames ($frame_0 \dots frame_6$) generamos 6 splines $S_0, \dots, S_5$ y 12 frames artificiales como se puede ver en la figura 4.

2.3. Construcción mediante interpolación por splines (cúbica)

Continuando con la idea de conseguir los frames artificiales o intermedios mediante la creación y evaluación de polinomios para cada píxel, pasamos a utilizar polinomios de grado 3. Por lo dicho en la introducción teórica creeríamos que esto puede brindar resultados más satisfactorios en la construcción del video final, aunque habrá que confirmar esto durante la experimentación.

Intuición

Siendo éste el tercer y último método a desarrollar, podemos abstraernos de ciertos detalles que fueron previamente explicados.

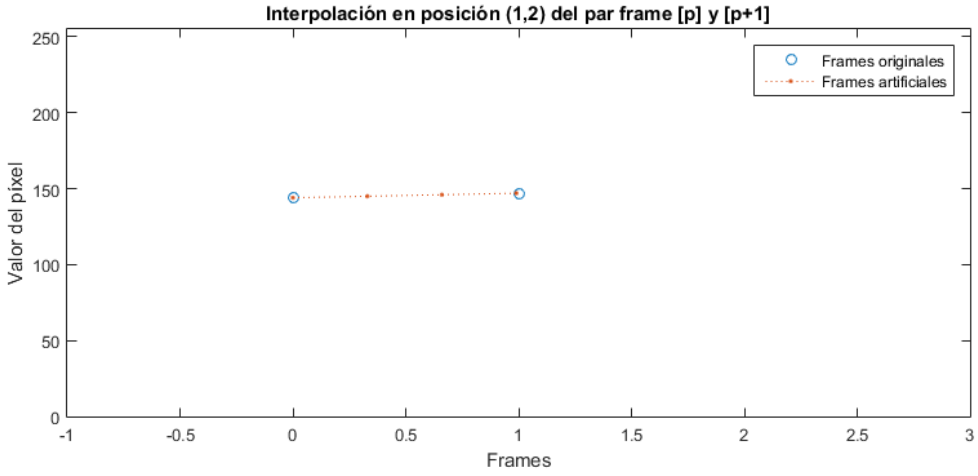


Figura 3. Resultado de interpolar usando 2 frames

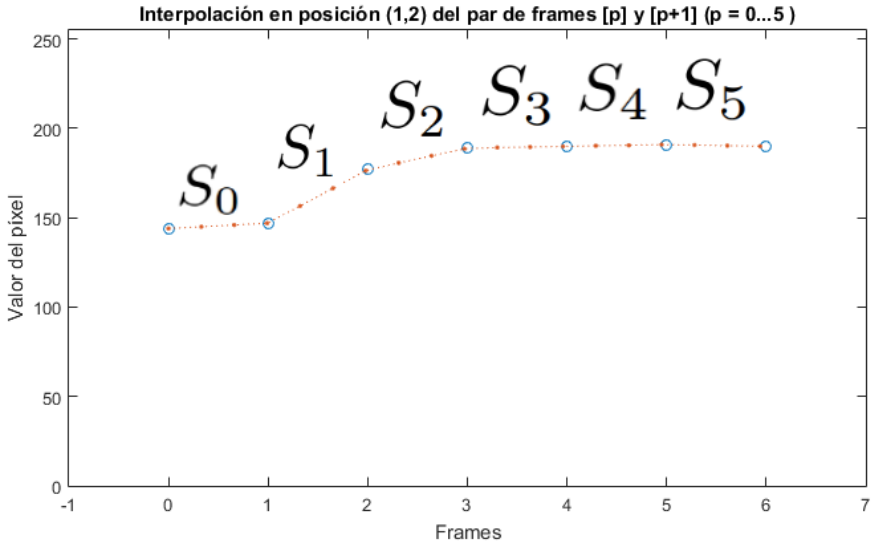


Figura 4. Resultado de interpolar usando 7 frames

Ya hemos dicho que pertenece a la familia de los interpoladores por segmentos. En principio, se podría pensar que el método no cambia su metodología con respecto al anterior. Nos limitaremos aquí a expresar en qué se diferencian.

A diferencia del método de Interpolación lineal, este método varía sus resultados dependiendo de todos los frames considerados para hacer el spline, es decir, no genera cada S_k únicamente en función del valor del píxel en los frames k y $k+1$, sino que todos los frames influyen en los coeficientes de la función S_k, y su evaluación en el punto t puede variar dependiendo de qué frames fueron considerados. Esto se debe, intuitivamente, a que el método utiliza la información de todos los frames para generar cada función $S_{i,j}$, combinándola en un único sistema de ecuaciones[a].

Una opción sencilla en su implementación es usar todos los frames del video para generar el spline. El problema es que el sistema matricial a resolver puede ser muy grande, potencialmente excediendo las capacidades de memoria de las computadoras comunes y trayendo inconvenientes no deseados al uso del método. Queda entonces la aplicación de una alternativa: crear múltiples splines por píxel, de a bloques. Es decir, tomar cierta cantidad de frames y definirla como el *tamaño de bloque*, para luego generar un spline por cada bloque.

a. ver sección 1.2.3

Perspectiva matemática

Considerando como un nuevo tamaño el tamaño de bloque (que eventualmente podría ser igual a la cantidad de frames del video, dando lugar a un único spline por píxel) generamos un spline por píxel por bloque. Los puntos a interpolar serán el tiempo t y el valor del brillo y de dicho píxel en todos los frames del bloque.

Usando la equivalencia vista en la Introducción teórica (del problema de encontrar los coeficientes de un spline con el de resolver un sistema de ecuaciones lineal) podemos generar la matriz asociada al sistema para cada spline a generar y resolver el sistema usando eliminación gaussiana normalmente, como ya hemos desarrollado en trabajos anteriores. Eso nos da como resultado los coeficientes (a_i, b_i, c_i, d_i) de cada polinomio cúbico S_k de los que componen el spline. Luego basta hacer

$$S(t) = \begin{cases} S_0(t) & \text{si } t \in [t_0, t_1] \\ \vdots & \\ S_{n-1}(t) & \text{si } t \in [t_{n-1}, t_n] \end{cases}$$

para generar cada spline S, y "conectar" entre sí los splines que corresponden a un mismo píxel. Como los splines interpolan efectivamente los puntos que coinciden con frames originales, la función $f(t)$ de cada píxel es continua incluso en los frames "límite" de cada bloque, aunque esto no sea cierto para la derivada primera y segunda. Naturalmente, al optimizar el algorimo se pierden algunas propiedades deseables.

Hace falta hacer hincapié en un detalle con el que nos encontramos a la hora de elegir cómo construir los bloques. Es importante que un *frame límite* sea tanto el último de un spline como el primero del siguiente. No hacerlo así (nuestra primera implementación), es decir, tomar un frame k como el último de un spline y el frame siguiente $k+1$ como el primero del spline siguiente lleva a que haya dos frames entre los cuales no se crean frames artificiales, perdiendo el efecto buscado entre ellos.

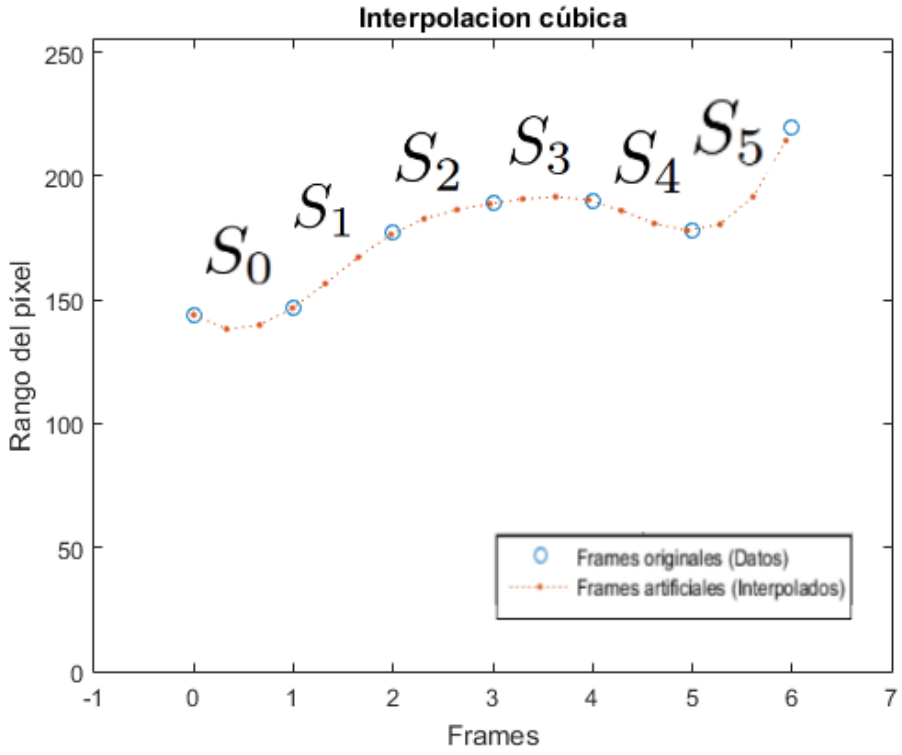


Figura 5. Resultado de interpolar usando splines

En la figura 5 graficamos cómo resulta la interpolación de un único bloque utilizando un spline. Se puede observar que la función generada resulta, al menos a simple vista, más "suave" que la lineal.

2.4. Cálculo del error de los métodos

Por último, nos interesa comparar de manera imparcial los métodos utilizados, utilizando alguna medida de cuánto se acercan a "la realidad": en nuestro caso, a lo que habría captado una cámara de alta frecuencia. Pero para hacerlo haría falta tener dos videos del mismo evento, filmados desde el mismo punto, uno con una cámara de alta frecuencia y otro con una normal. Como las cámaras ocupan un espacio, esto es físicamente imposible[**wiki˙impenetrability**]. Lo que haremos, en su lugar, para comparar los métodos, será "descartar" frames de un video (normal o no) y darle este nuevo video "recortado" como entrada a nuestros métodos de interpolación. La comparación entre el resultado y el video original nos dará una aproximación de cuán "acertados" son nuestros métodos para siumlar lo que ocurre entre dos frames de un video cualquiera.

Para realizar la comparación, usaremos las definiciones de `ECM` y `PSNR` ya dadas en la introducción teórica, comparando frame a frame entre los dos videos.

3. IMPLEMENTACIÓN

A continuación nos explayaremos sobre los detalles de la implementación del método propuesto. En las secciones previas hemos dado una introducción al problema, su modelo y justificación de por qué el mismo sirve y a su vez hemos expuesto un método que nos permite hayar la solución.

Datos al aire

$f(x) = a + b(x - x0)$

$y0 + (y1 - y0) * ((x - x0)/(x1 - x0)) \implies a = y0 \ y \ b = (y1 - y0)/(x1 - x0)$

Algoritmo 1: Pseudocódigo del algoritmo de Interpolación lineal

Entrada: Puntos del dominio conocidos x ; Puntos de imágen conocidos y, grado función gr
Salida: Vector coeficiente a, Vector coeficiente b

per $i = 1 \dots gr - 1$ **fai**
  $a_i \leftarrow y_i$;
  $b_i \leftarrow \frac{y_{i+1} - y_i}{x_{i+1} - x_i}$;
fine
devolver a,b

$f(x) = d + c(x - x0) + b(x - x0)^2 + a(x - x0)^3$

Algoritmo 2: Pseudocódigo del algoritmo de Interpolación por Splines

Entrada: Puntos del dominio conocidos x ; Puntos de imágen conocidos y, grado función gr
Salida: Vector coeficiente a, Vector coeficiente b, Vector coeficinete c, Vector coeficiente d

$MatrizSplines(0, 0) \leftarrow 1$;
$MatrizSplines(1, 0) \leftarrow 0$;
$vectorIndep_0 \leftarrow 0$;
per $i = 2 \dots gr - 1$ **fai**
  $MatrizSplines(i, i - 1) \leftarrow x_i - x_{i-1}$;
  $MatrizSplines(i, i) \leftarrow 2\ (x_{i+1} - x_{i-1})$;
  $MatrizSplines(i, i + 1) \leftarrow (x_{i+1} - x_i)$;
  $vectorIndep_i \leftarrow 3\ \frac{y_{i+1} - y_i}{x_{i+1} - x_i} - \frac{y_i - y_{i-1}}{x_i - x_{i-1}}$;
fine
$MatrizSplines(gr, gr) \leftarrow 1$;
$MatrizSplines(gr, gr - 1) \leftarrow 0$;
$vectorIndep_{gr} \leftarrow 0$;
$b \leftarrow ResolverSistemaEcuaciones(MatrizSplines, vectorIndep)$;
per $i = 1 \dots gr$ **fai**
  $a_i \leftarrow \frac{1}{3\ \frac{b_{i+1} - b_i}{x_{i+1} - x_i}}$;
  $b_i \leftarrow \frac{y_{i+1} - y_i}{x_{i+1} - x_i} - \frac{1}{3\ (2\ b_i + b_{i+1}) * (x_{i+1} - x_i)}$;
fine
$y \leftarrow d$
devolver a,b,c,d

4. EXPERIMENTACIÓN

A diferencia de trabajos previos, la experimentación de este trabajo es mucho más explorativa que basada en hipótesis a confirmar. Claramente, al comenzar la experimentación se tuvieron algunas hipótesis (detalladas a continuación), pero se comenzó esta tarea más con un objetivo de observar que ocurre con los métodos propuestos según algunas variables que fueron identificadas.

A su vez, el dominio de este trabajo es más subjetivo, si se quiere, que los trabajos previos. Es decir, al interpolar cuadros (o *frames*) de un video, nos interesa como es la percepción de la audiencia del mismo. ¿Se nota la interpolación?¿Da una sensación anormal?¿No se nota, quizás?. Básicamente: el resultado obtenido, ¿es bueno?. Claramente esto es muy subjetivo y dependiente, entre otras cosas, de la persona que emite el juicio. Así pues, se tuvo que buscar alguna forma de comparar los distintos métodos de la forma más objetiva posible. Pero tampoco se quizo dejar de lado este análisis "subjetivo", que al fin y al cabo, es el más importante.

A lo largo de esta sección comenzaremos primero enunciando las hipótesis que se pudieron enunciar antes de comenzar, la explicación de las variables con las que se experimentó (derivadas a partir de las hipótesis), para luego pasar a la experimentación misma (con un mínimo análisis de correctitud), dónde se analizan los aspectos cuántitativos y cualitativos (y como se a explicado, algunos de estos últimos de manera subjetiva). Finalmente, realizamos una análisis breve de los *artifacts* frutos de los videos interpolados y terminamos esta sección indicando ciertos trabajos a futuros que podrían ser de interés.

4.1. Hipótesis

EN esta parte del trabajo se explican algunas de las hipótesis que pudieron ser formuladas al pensar en el problema. Como ya se ha dicho, la naturaleza de este trabajo ha sido más exploratoria, pero aún así se comenzó el mismo con algunas conjeturas y nociones de lo que debía ocurrir.

1. **Hipótesis** saraza
   Motivación saraza

4.2. Variables Identificadas para la Experimentación

A partir de las hipótesis enunciadas, se determinaron ciertas características y parámetros que posiblemente afecten a la calidad de las interpolaciones aplicadas a los videos. Estas son:

Frames Interpolados
Método de Interpolación
Tamaño de Bloque (splines)
Resolución del Video
Duración del Video
Tipo Movimiento grabado por la Cámara
Tipo de Movimiento de la Cámara

Estas son algunas de las variables que se podrían en tener en cuenta (en particular las que se tuvieron en cuenta en este trabajo), pero no son las únicas. Por dar un ejemplo, una variable con la que no se trabajó fue con los videos que cambián de cámara, donde se pasá de un tipo de imágen a otra de formá rotunda en frames contiguos (podría haber algún tipo de transición también, que sería otro caso a estudiar). Fue meramente por cuestiones de tiempo que se decidió limitar el enfoque de este trabajo a las variables/parámetros enunciados.

4.3. Metodología de Experimentación

DURANTE la experimentación, y dada la naturaleza de la misma, se siguió una misma metodología para recabar la información necesaria para ser luego analizada.

En primer lugar se decidió trabajar con imágenes en escala de grises o, coloquialmente llamado *blanco y negro*. Esto es así ya que dentro del alcance de nuestro trabajo, estamos estudiando (principalmente) la correctitud de los frames interpolados. Si se tuvieran en cuenta los colores, se tendría la complicación del error agregado de interpolar los canales de colores por separado, lo cual escapa a los objetivos del trabajo (además de tener la limitante del tiempo para realizar este trabajo.). Resumiendo, se tomó esta decisión para simplificar y enfocar más el análisis a realizar.

La forma en la que se presentarán los resultados más adelante nada tiene que ver con el orden en el que se realizó la experimentación.

De la sección anterior se puede observar que existen dos variables que nos indican (o limitan) los tipos de videos con los que trabajaremos. Estas dos variables no son otra que las diferentes combinación del tipo de movimiento filmado y de la cámara que realiza la grabación. Así pues, se terminó teniendo cuatro combinaciónes posibles: *cámara fija-imágen fija*, *cámara fija-imágen móvil*, *cámara móvil-imágen fija* y *cámara móvio-imágen móvil*[a].

Para cada una de estas categorías de videos, se efectuó la interpolación con los 3 métodos expuestos y todas sus posibles combinaciones de parámetros (cantidad de frames a interpolar entre frame y frame, y en el caso de spline el tamaño de bloque.). Para luego poder hacer un análisis objetivo de la calidad de los frames interpolados resultantes, lo que se hizo fue remover de los videos originales una cantidad calculada[b] de frames antes de interpolar el mismo. De esta manera se puede comparar los frames interpolados resultantes con los frames originales (los que fueron removidos previos al procesamiento), que no son otra cosa que el resultado que uno quisiera obtener de la interpolación.

Luego se realizan los análisis de los métodos independientemente de los demás métodos, es decir que analizamos los resultados obtenidos de cada método respecto de sus parámetros de entrada.

Continuamos realizando un análisis de método versus método, para los mismos parámetros (en el caso de splines, al tener 2 parámetros de entrada[c] se tuvieron que utilizar resultados obtenidos del análisis del método realizado previamente).

Por último, antes de terminar enunciando las conclusiones obtenidas para el tipo de video estudiado, se generaron unos videos que realizan la comparativa frame a frame de la diferencia entre el frame del video original y su contraparte interpolada para luego ser estudiados (visualizado en forma de *heatmap*). La idea de esto es la de encontrar en que zonas de los frames, respecto de lo que ocurre en el video, genera problemas para los métodos estudiados.

Las métricas utilizadas para los análisis de los métodos no son otras que las sugeridas por la cátedra: el *Error Cuadrático Medio* [**mse**] y el *Peak Signal to Noise Ratio* [**psnr**]. El primero es una medida de error entre un valor (en nuestro caso, un frame) estimado y lo que es estimado (el frame original). Mientras que el *PSNR* es un ratio que toma en cuenta el *ECM* y el valor máximo siendo estimado[d]. Justamente el PSNR es utilizado para cuantificar la calidad de la reconstrucción de una señal (o en nuestro caso, de un frame), un alto valor del mismo del mismo indica mayor calidad de la reconstrucción (aunque se debe tener cierto cuidado en los casos donde existe compresión de datos involucrada, caso que no aplica a este trabajo). También se usaron los valores medios, desvío estándard, máximo y mínimo del ECM para realizar comparaciones para el mismo método y sus diferentes parámetros, como así también entre los distintos métodos.

Vale la pena aclarar que en las conclusiones generales es donde también se realiza un análisis más subjetivo, basado en la percepción de los autores de este trabajo. Claramente esta no es una muestra lo suficientemente alta de las posibles audiencias de los videos, pero alcanza dado el objetivo didáctico del trabajo.

Para concluir con esta sección, agregamos a su vez que se realizaron otros experimentos más puntuales para analizar los aspectos de correctitud de los métodos (mínimamente) y del tiempo de cómputo, como así también una comparativa de como los métodos se ven afectados según la variable más importante de todas: el video.

Los videos originales utilizados para la experimentación pueden obtenerse en https://drive.google.com/open?id=0B0RfkWV-4-XqMlhfa0Z2WUMtRTg

a. Existen matices en estas combinaciones. Por ejemplo, *la suavidad* de los movimientos, que podrían variar en intensidad yendo de suaves a bruscos. Nuevamente, por cuestiones de tiempo se decidió no incursionar en estos aspectos.

b. En base a la cantidad de frames a interpolar.

c. Cantidad de bloques a interpolar y tamaño de bloque.

d. Oriundo del análisis de señales, donde indica la relación entre la potencia máxima de una señal y la potencia del ruido que la afecta.

4.4. Validación de la Implementación

En este apartado, nos centramos a corroborar que los pasos de la implementación se dieron acorde a lo que cada método visto propone. Para ello, creamos dos instancias en la cual se podrán visualizar con facilidad los cambios generados al aplicar el efecto de slowmotion. De hecho, uno de ellos se basará de reproducir una sola imágen durante todo el video. El restante se basará en observar el cambio entre los dos extremos que proporciona la escala de grises, es decir, de blanco a negro.

4.4.1. Blanco-Negro

Nos situamos primero en el caso donde dicho video se comprende de dos tipos de frames, que es repetido una cierta cantidad de veces en un determinado tiempo. La particularidad de dichos frames, es que si se contempla éstos como matriz, se confirma la igualdad en cada posición del cuadro. Debido a que estamos trabajando sobre escala de grises, decidimos clasificar al tipo A como un frame con todos sus píxeles en negro (equivalente a 0) y el tipo B con píxeles en blanco (equivalente a 255).

En primer lugar, se seleccionó una imágen de esta característica ya que nos podemos abstraer del procedimiento sobre el frame en conjunto. De esa manera, nos podemos enfocar en analizar cualquier posición (i, j). ¿Qué utilidad nos brinda esto último? El hecho de poder evaluar en detalle, el comportamiento del método aferrado a nuestra implementación.

En segundo lugar, haber escogido dos valores que representan los extremos en la escala de representación, nos trae una mejor intuición del resultado esperado. Con esto nos adelantamos a decir, que el video alentizado intentará reflejar lo que sucede cuando se translada del tipo A al B mediante los frames interpolados.

Por último, aclaramos que el video solo contendrá un cambio del frame de tipo A al B, y ese mismo será definitorio.

Vecino más cercano:

Sabemos que su idea revoca en crear los cuadros intermedios copiando de un extremo u otro, tal como se explicó durante el desarrollo. Por ende, el resultado que éste dará al aplicarlo sobre el video, será bastante trivial. De hecho, lo único que se podrá apreciar es la mayor duración del video resultante con respecto al original.

Pasamos a los resultados, que por la poca complejidad algorítmica que este método implica, se validará en brevedad a lo que se buscaba.

Resultado:

Comprobamos que al decifrar los frames intermedios como matrices, éstos se identificaron con su extremo más cercano. Si al parámetro de cantidad de frames a adherir se le asignaba un número impar, luego en teoría el valor del extremo derecho (en este caso, el blanco) tendría mayor presencia. Sin embargo, como la diferencia es de un solo frame adicional, no se notó nada en la práctica debido a la cantidad de frames por segundo que corre el reproductor de video.

En consecuencia, recolectamos la totalidad de frames del video en $slowmotion$ para comprobar si la implementación acertó con la distribución de cuadros intermedios.

Interpolación lineal:

Nos inclinamos a una perspectiva que abarca mayor seriedad, ya que su comprobación se justificará del lado matemático. Si los uńicos dos puntos a trazar son el $(x0, y0) = (0, 0)$ y $(x1, y1) = (1, 255)$, luego la función a considerar tendrá la forma:

$f(x) = y0 + (y1 - y0) * \frac{x-x0}{x1-x0} = 255 * \frac{x}{255} = x$

Con esto último, ya se puede considerar que tampoco traerá dificultad al momento de analizar correctitud.

Resultado:

Fuimos elevando el parámetro de entrada, agregando más frames de por medio. De esta manera, el método se encargaba de evaluar cada $f(x) = x$, con $x \in \{\frac{1}{fr}; \frac{2}{fr} \ldots; \frac{fr}{fr}\}$. Como dicha función es creciente, notamos como efectivamente el avanze del video reflejaba el esclarecimiento del mismo, tomando valores de grises más suaves hasta llegar al cero.

En este caso, mostramos un caso donde se le adicionó 10 frames, y luego comprobando al obtener la totalidad de imágenes, se verificó que los valores de los píxeles intermedios, coincidían con la función lineal en dicho punto.

(Foto)

Splines

Seguimos con la misma metodología que con interpolación lineal, definiendo la función en cuestión:

$f(x) = a(x - x0)^3 + b(x - x0)^2 + c(x - x0) + d$

Pero, ¿Cómo hallamos los coeficientes del polinomio? Podríamos por un lado, realizar las cuentas a papel y de ahí seguir con el procedimiento de verificar con respecto a la implementación. Sin embargo, eso sería bastante engorroso de realizar, por lo que optamos por usar un software inteligente cuyo nombre es $MatLab$. Usando la función $interpo1$ podemos obtener el valor de cualquier punto intermedio, en particular los que se evaluaron en nuestro programa.

No obstante, si bien al notar que efectivamente la función $f(x)$ en este caso es también una lineal idéntica a la anterior, hay que tener en cuenta una herramienta clave en $Splines$: la construcción por bloques. Pero como decidimos que cada bloque comparta su primer y último frame (excepto los extremos, que comparten alguno de los 2), luego no existe el caso de que se divida de forma tal que quede todo negro de un lado y blanco en lo que sigue.

Resultado:

Ideamos una instancia con parámetro de adición de frames igual a 10, y dividido en 3 bloques. Como lo que importa aquí es validación y no performance, optamos por un valor menor. Dicho esto, mediante $MatLab$ visualizamos cada Spline generado por bloque y comparando con la implementación, no se encontró ninguna objeción a lo ya comentado.

(Falta unos gráficos, pero honestamente no estoy seguro de si esto es lo que se quiere)

4.4.2. Cámara Fija - Imágen Fija

Equivalente a pensar que una imágen está siendo reproducida durante un lapso de tiempo, que difiere al hecho de que una cámara esté quieta, enfocando a un objeto que puede ser sofocado por la luz. Y está claro que el mero hecho de querer alentizar un video de tal característica, solo servirá para aumentar la duración de la misma. Por lo conceptualmente visto, no hay dudas de que si la implementación se realizó de forma correcta, no existiría ningún cambio en el video.

Usamos metodologías idénticas para los tres métodos. Esto es, extraer cada cuadro artificial y corroborar que coincide con la foto utilizada.

Resultado:

Como en cada píxel (i, j) el valor se mantuvo constante, así lo fue con las funciones interpoladoras. De esta manera, cualquier punto que era evaluado por más frames que se le quisiese acomodar, solo producía un video más largo.

(No esta andando el programa by the way, con out_of_range)

4.5. Tiempos de ejecución

4.6. Cámara Fija - Imágen Fija

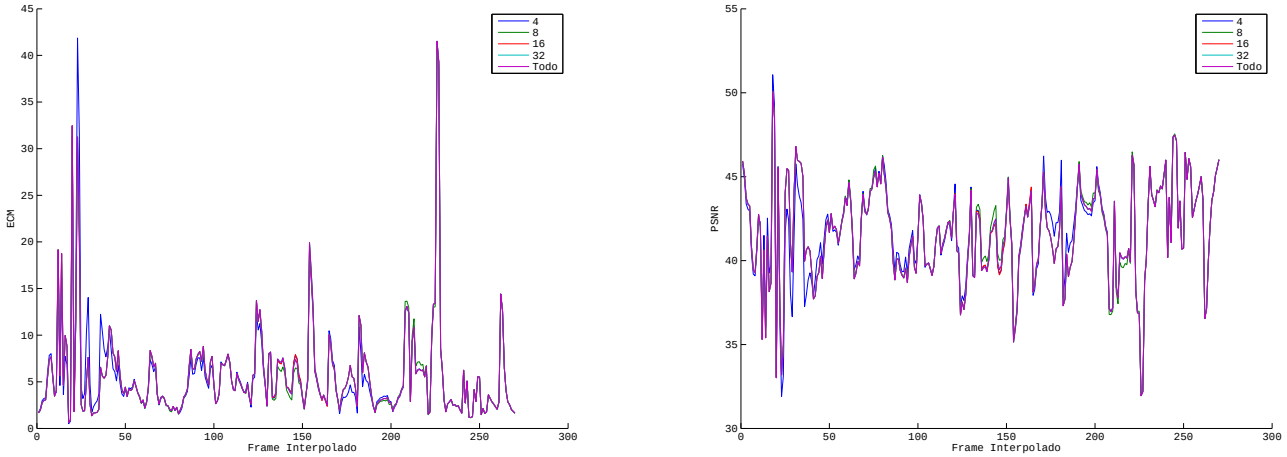
En principio, pudiera parecerle al lector, que este experimento carece de sentido. Básicamente se está evaluando un video que no tiene movimiento de ningún tipo. Lo cual es erróneo. En el caso evaluado, se tiene un video de la vista de unos edificioes y la transición del día a la noche. Esto hace que los colores, lo cual hace al observar el video en blanco y negro, que cambien las *tonalidades* o brillo de las distintas partes del video.

Así pues, un caso que en principio haría que uno diga "la interpolación debería funcionar sin error, pues que no hay movimiento" (como se mencionó, que el movimiento sea el principal factor del error de los métodos a aplicar es la principal hipótesis a comprobar) no nos queda tan claro.

Comenzamos entonces a presentar los resultados obtenidos a partir de la aplicación de los métodos explicados e implementados para este video.

4.6.1. Spline

Lo primero a tener en cuenta al experimentar con este método es que tenemos 2 variables (la tercera variable, el video, está fija): la cantidad de frames a interpolar y el tamaño de bloque. Decidimos entonces, primero "fijar" la cantidad de frames a interpolar y análizar que ocurre al variar el tamaño del bloque.



(a) ECM para 10 frames interpolados

(b) PSNR para 10 frames interpolados

Figura 6. Comparativa tamaño de bloque para 10 frames interpolados

Se puede observar en la figura 6 que las métricas utilizadas (ECM y PSNR) para los distintos tamaños de bloques son prácticamente idénticas: están solapadas casi igual para todos los frames interpolados del video. Si bien en los gráficos expuestos se pueden llegar a notar ciertas regiones de frames donde alguno de los tamños de bloque difiere (por ejemplo, en la figura 6a en el bloque de frames del 1 al 50 se ve que para el tamaño de bloque 4 tenemos 2 picos: uno donde tiene un mayor ECM que el resto y otro donde tiene un menor ECM, lo cual se ve reflejado también en la gráfica del PSNR[a]), como comportamiento general observamos que el tamaño del bloque (aplicado "a ciegas" sobre la totalidad del video) no nos asegura una mejor ni peor estimación/interpolación. Este patrón se repite para las restantes variantes en función de la cantidad de frames interpolados, motivo por el cual sólo exponemos los resultados para 10 frames interpolados a la hora de describir este comportamiento.

Habiendo analizado que el tamaño del bloque (utilizado de la manera explicada) no pareciera tener influencia en el error comentido en un sentido amplio (ya que observamos como varía el ECM y PSNR frame a frame), evaluamos algunos resultados más "estadísticos", si se quiere, del ECM a lo largo de todos los frames según cada tamaño de bloque.

a. Dónde para el pico de mayor ECM se obtiene un PSNR menor que el resto, y al revés para el pico de menor ECM. Esto tiene sentido ya que como se explicó, el PSNR es una métrica de la calidad de la interpolación/estimación del frame, y tener un mayor ECM indica menor calidad en la estimación.

Bloque	Mean	Std	Máx	Mín
4	5.6919	5.4221	41.8650	0.5080
8	5.6497	5.0825	40.8340	0.6403
16	5.6812	5.0828	41.5168	0.6407
32	5.6881	5.0950	41.5173	0.6407
Entero	5.6881	5.0950	41.5173	0.6407

(a) Valor medio, Desvío Estándar, Máximo y Mínimo

Bloque	vs 4	vs 8	vs 16	vs 32	vs Entero
4	0	11.0099	11.0156	11.0156	11.0156
8	2.9723	0	1.0059	1.0069	1.0069
16	2.8873	1.4356	0	0.5544	0.5544
32	2.9104	0.9827	0.6176	0	0
Entero	2.9104	0.9827	0.6176	0	0

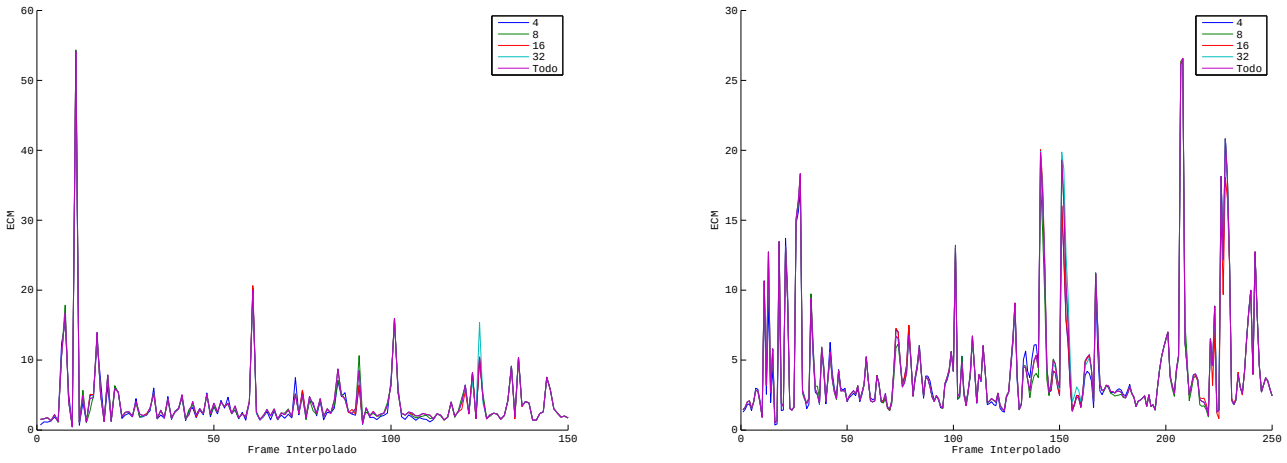
(b) Diferencia Máxima

Figura 7. Comparativa ECM según tamaño de bloque

Si observamos la tabla 7a, obsevamos resultados que validan lo observado previamente: el tamaño del bloque no pareciera influir en el error cometido. Las diferencias son mínimas, salvo por ahí el error mínimo cometido por el tamaño de bloque 4 es menor que para los demás tamaños de bloque y que su desvío estándar es un poco mayor. Aunque estas diferencias también son muy pequeñas, siendo despreciables a la hora de afirmar que el tamaño de bloque no afecta al error cometido.

Continuando, si se observa la tabla 7b, la cual nos señala la diferencia máxima de los errores cometida entre dos interpolaciones con distinto tamaño de bloque (es decir, cual fue la diferencia más amplia del ECM para un mismo frame entre dos tamaños de bloque), observamos que el tamaño de bloque 4 cometió, respecto de los demás tamaños, un error más amplio que los demás. Si volvemos a observar la figura 6a, podemos observar que hay un claro pico de error más alto para el tamaño de bloque 4 entre los primeros 50 frames. Seguramente esta diferencia observada en la tabla se debe a dicho frame, donde el tamaño de bloque 4 cometió mucho más error que los demás métodos. Descartando este valor, concluímos que el tamaño de bloque no afecta (salvo casos aislados) al error cometido por la interpolación mediante spline[a].

Hasta ahora hemos analizado lo que ocurre para diferentes tamaños de bloque. Pasamos entonces a observar que ocurre para diferentes cantidades de frames interpolados.



(a) ECM para 1 frame interpolado

(b) ECM para 5 frames interpolados

Figura 8. Comparativa según cantidad de bloques interpolados

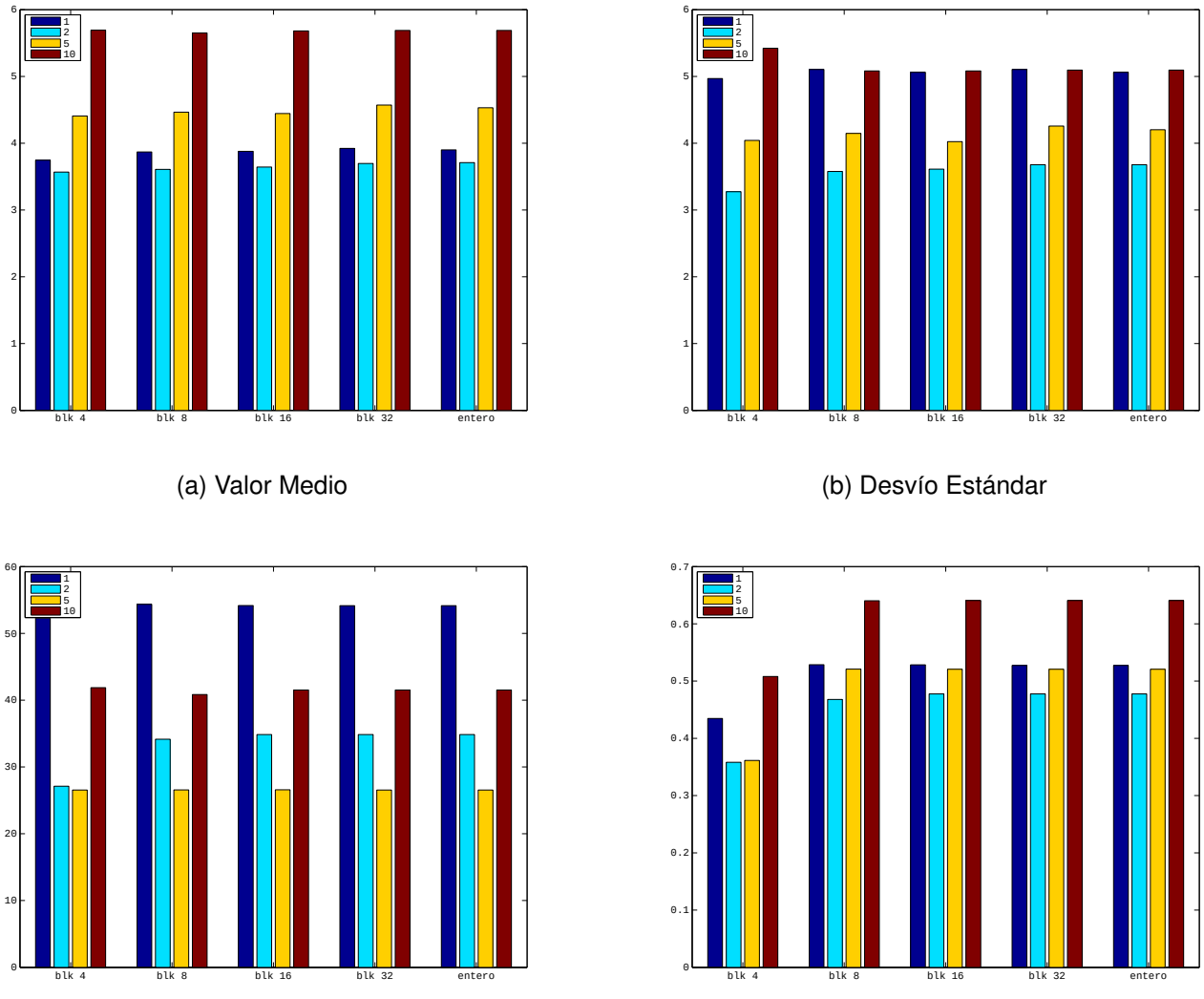
Lo primero, evidente, que salta a la vista de observar las figuras 6a, 8a y 8b, es la cantidad (y valor o intensidad) de los picos de error. Se ve que a medida que se incrementa la cantidad de frames interpolados, comienzan a acentuarse el comportamiento en ”picos” de los errores frame a frame, siendo cada vez más amplia la diferencia

a. Por cuestiones de tiempo no se pudo buscar exactamente el frame donde ocurrió esta última diferencia explicada, para observar su correlación con lo que ocurre en el video.

entre el mínimo error cometido de un ”pico” y el error máximo alcanzado en el mismo (observar con detenimiento la escala de las figuras). A su vez se observa de manera intuitiva que la media del ECM iría aumentando a medida que se interpolan cada vez más frames. Esto último tiene sentido, ya que ahora se tienen frames cada vez más distantes y se interpolan todos los frames intermedios. A mayor distancia entre los frames ”dato” que se usan para calcular el polinomio interpolador, se puede pensar que menos información se tiene y por lo tanto la calidad de la estimación se ve perjudicada.

Por otra parte, se observa un comportamiento similar, aunque de distinta intensidad, en los 3 gráficos: el ECM tiene un comportamiento oscilante, aunque no cíclico ni constante. Es decir, se ve que el ECM tienede a incrementarse hasta llegara un ”pico”, para luego comenzar a descender. Y así repetidamente (pero, como se a dicho, la ”intensidad”, o valores que alcanza -tanto mínimos como máximos locales- varía a lo largo de los frames del video). Prestando una mayor atención, se nota que el ECM tiende a ser más bajo para aquellos frames que están más cerca de alguno de los frames originales (no interpolados) del video. Nuevamente, el valor máximo del ECM alcanzado para los frames contenidos entre 2 frames originales/no interpolados varía (todavía por motivos desconocidos), pero el comportamiento de incrementarse el ECM a medida que nos alejamos de un frame original y decrementarse al acercarse es un patrón que se da en todos los casos.

Observermos los datos estádisticos del ECM para los diferentes valores de los frames interpolados[a]:



(a) Valor Medio

(b) Desvío Estándar

(c) Máximo

(d) Mínimo

Figura 9. Estádisticas ECM Según Frames Interpolados - Spline

a. Estos datos solo consideran los frames interpolados

Analizando la figura 9, lo primero que salta a la vista es que claramente hay un patrón en el ECM comparando por tamaño de bloques. La relación entre los distintos valores de frames interpolados se mantiene para cada grupo correspondiente al tamaño de bloque. Esto es consistente con los resultados vistos anteriormente en esta misma sección.

En segundo lugar, ya concentrandonos en la cantidad de frames interpolados, vemos un resultado que no era esperado según nuestras hipótesis: observamos un valor medio de error, desvío estándar y error máximo que es notoriamente menor para la interpolacio de 2 frames (e incluso, se obtuvo la mejor aproximación de un único frame, como se observa en el gráfico del ECM mínimo). Es decir, pare este video observamos que se obtienen los mejores valores generales de error cuando se interpola de a 2 frames en lugar de a 1, como se planteó en las hipótesis. Pareciese que para este video, tener un poco menos de información mejora sustancialmente el ECM (ver la comparación para 1 frame versus 2 frames).

Siguiendo esta misma línea de análisis, observamos al interpolar 1 frame y 10 nos dan un desvío estándar alto comparado con las interpolaciones de 2 y 5 frames, aunque los valores medios del error para 10 frames son muchos más altos que para el resto de los valores. Esto último si confirma, parcialmente, la hipótesis de que a mayor cantidad de frames interpolados, mayor será el error (o peor la aproximación).

Para finalizar el análisis de estos datos, resulta realmente sorpresivo observar un error medio menor para la interpolación de 2 frames que para la de 1 frame. Aún así, la diferencia entre estos es mínima, y si despreciamos esta diferencia diciendo que la diferencia es tan pequeña que podemos considerar que ambas tiene el mismo ECM medio, vemos confirmada la hipótesis ya mencionada, ya que a mayor cantidad de frames interpolados, mayor error medio. Aunque vale la pena mencionar que, el desvío estándar nos indica que para una interpolación de 1 frame en este tipo de videos, tenemos una gran varianza frame a frame de la calidad de la interpolación. Descontando este valor sorpresivo, observamos que esta varianza mantiene la relación inicialmente esperada, aumentando a medida que aumenta la cantidad de frames interpolados.

Por último, es interesante tratar de asociar los errores cometidos por el método con lo que está ocurriendo en el video. En la figura 4.6.1 se puede observar una captura del video comparativo de este método para tamaño de bloque 4 y cantidad de frames interpolados 1[a].

Decidimos utilizar estos parámetros ya que de los datos vistos previamente, con los mimos tenemos un valor alto (relativo a los demás parámetros) de varianza en el ECM frame a frame. De esta manera, consideramos que sería menos costoso identificar la región (o regiones) del video que más influyen en el error de la interpolación.

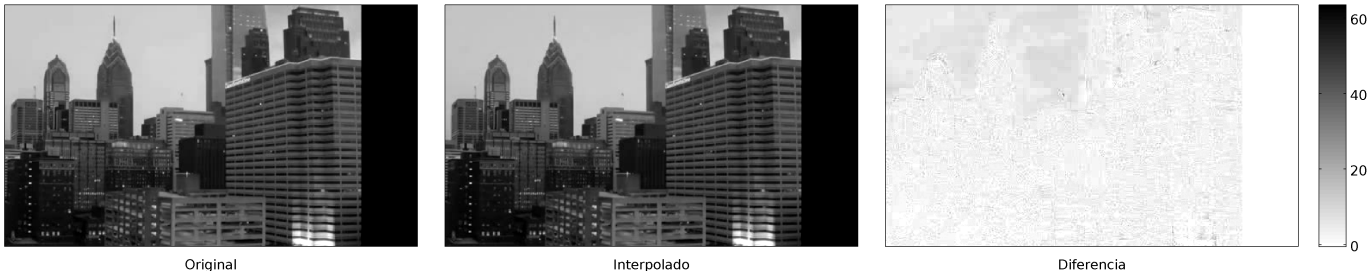


Figura 10. Región peor aproximada por la interpolación por spline

En dicho video observamos que los pixeles que hacen al ECM están distribuídos por todo el frame, asociados según interpretamos, a los cambios de iluminación[b]. La captura no es capaz de mostrar esto, pero la mostramos ya que es representativa de la zona del video que es peor aproximada por la interpolación. Se puede observar que aquellas partes más afectadas por los cambios de iluminación (el cielo y las partes altas de los edificioes) tienen un error un poco más alto que las partes bajas de los edificios (especialmente la parte inferior izquierda), sobre las cuales el efecto de la sombra de los demás edificios mitiga el efecto de los cambios de iluminación (ya que permanecen más oscuros de por sí).

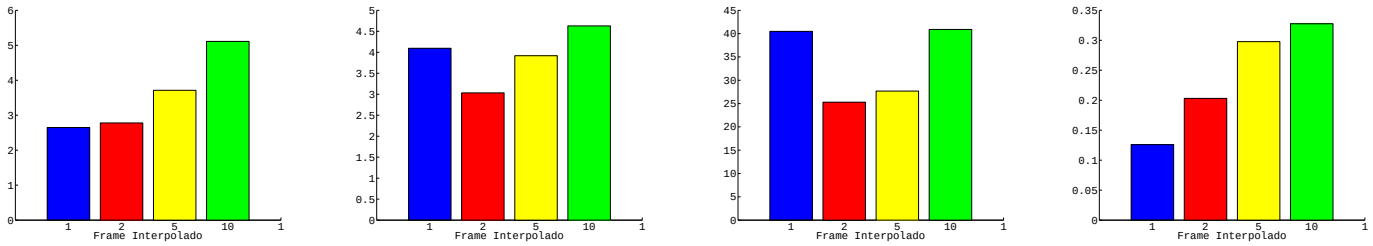
Es decir, se observa que aquellas partes del video que se ven más afectadas por cambios más ”bruscos” o diferenciados de iluminación (pasar de alto brillo a poco brillo) son aquellas zonas que más contribuyen al error de la interpolación.

a. https://drive.google.com/open?id=0B0RfkWV-4-XqNFJMT1l5NXg2VHM
b. Recordemos que el video trata de unos edificios (y parte del cielo) en una transición acelerada del día a la noche.

4.6.2. Interpolación Lineal

A la hora de analizar los resultados de un mismos método para distintos valores de frames interpolados (como en toda esta sección, la variable del video se encuentra fijada), observar frame a frame y comparar el ECM/PSNR no resulta un parámetro válido. Esto se debe a que comprar cualquiera de estas métricas tiene sentido al hacerlo sobre un mismo frame interpolado, y dado que para cada parámetro se interpolan distintos frames, la comparación carece de sentido[a].

Así pues, a la hora de analizar y comprar un mismo método y que ocurre al tener que interpolar más información/frames, recurrimos a las métricas estadísticas del ECM que fueron presentadas en la experimentación anterior. Las mismas pueden observarse en la figura 11.



(a) Valor Medio (b) Desvío Estándar (c) Máximo (d) Mínimo

Figura 11. Estádisticas ECM Según Frames Interpolados - Método Lineal

En este caso, como se puede observar en los gráficos, nuestra hipótesis sobre la calidad de la interpolación en función de la cantidad de frames a interpolar se cumple para el valor medio. A menor cantidad de frames a interpolar, menor ECM medio, aunque el error máximo cometido de todas las variantes (y el mínimo también) pertencen al caso en que se interpola de a 1 frame (es decir, el caso donde se tiene la mayor canttidad de información "original"). Nuevamente, como ocurrió con spline, pareciera que hay algún frame en particular del video que al ser interpolado de a 1 frame, genera un pico de error. De la observación del video interpolado, se ve que hay un momento al principio del mismo donde hay una baja del brillo instantánea, antes de que vuelva a su nivel normal (o similar al nivel previo antes de este evento). Esto es distinguible al ojo humano, y intuímos que lo que puede estar ocurriendo es que los frames correspondientes a esta baja de brillo son utilizados en la interpolación cuando se interpola de a 1 frame, y que no se lo usa cuando se incrementa este parámetro. De esta manera, al utilizarlo en el primer caso, se le baja el brillo a los frames interpolados, mientras que en el segundo caso está no ocurre, obteniendose el error únicamente en los pocos frames que son oscuros.

Nuevamente, se ve que el comportamiento del ECM (o PSNR, que sigue el mismo patrón) es oscilante entre números altos de error y bajos (o de aproximación, en el caso de PSNR). Esto se puede ver en la figura 12.

De nuestro análisis, entendemos que el error de interpolar entre frames cercanos será similar, salvo cuando se tiene un cambio abruto de un frame al siguiente en el video original. Así pues observamos en el gráfico (que es similar a los de mayor cantidad de frames interpolados, salvo alguna variación menor en los números) que el PSNR[b] se mantiene en valores por encima de los 45db la mayor parte del tiempo, salvo ciertos picos de error que bajan su valor por unos pocos frames. Esto estaría relacionado con los cambios de iluminación distinguibles al ojo que tiene el video.

Para finalizar con el análisis del método lineal para este video, nuevamente generamos un video que comparase la diferencia entre los frames para la interpolación lineal de 10 frames[c] (utilizando el mismo argumento que en splines, elegimos este parámetro ya que es el que mayor desvío estándar tiene de todas las variantes). Presentamos a continuación una captura del mismo, que consideramos representativa de las regiones de los frames donde se influye más en el error de interpolación.

a. Se podría tratar de encontrar frames interpolados en común, pero dicho trabajo complicaba mucho la tarea del análisis y por cuestiones de tiempo no pudo ser considerada esta opción.

b. Recordemos que esta es una métrica que a mayor valor, más fidedigna es la estimación del frame.

c.

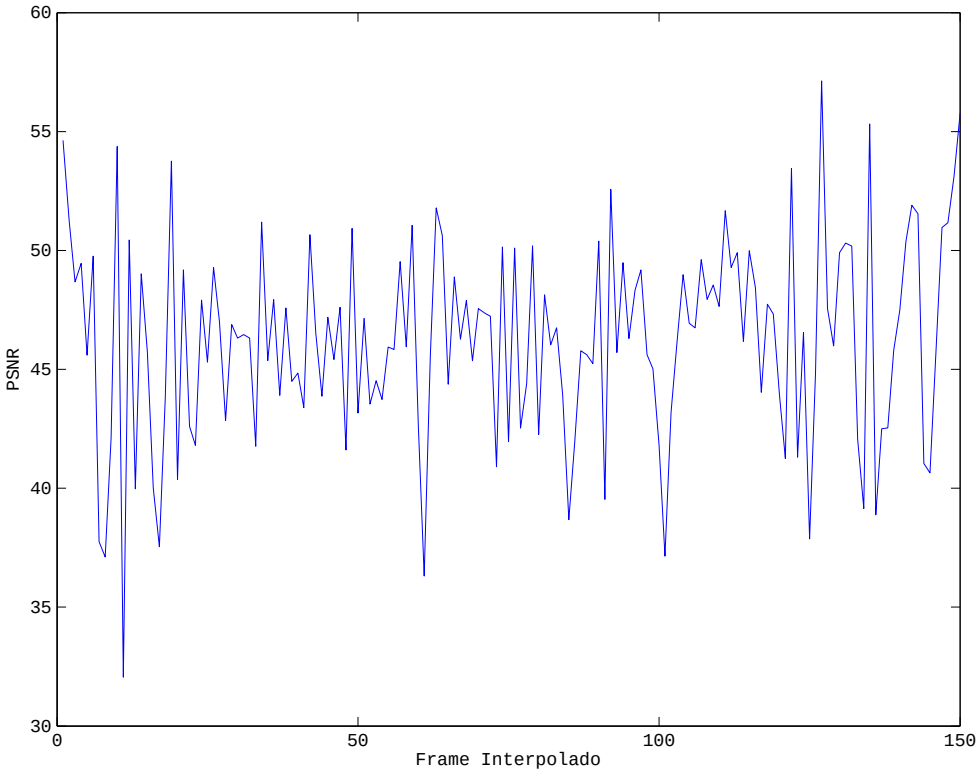


Figura 12. PSNR para 1 frame interpolado - Método Lineal

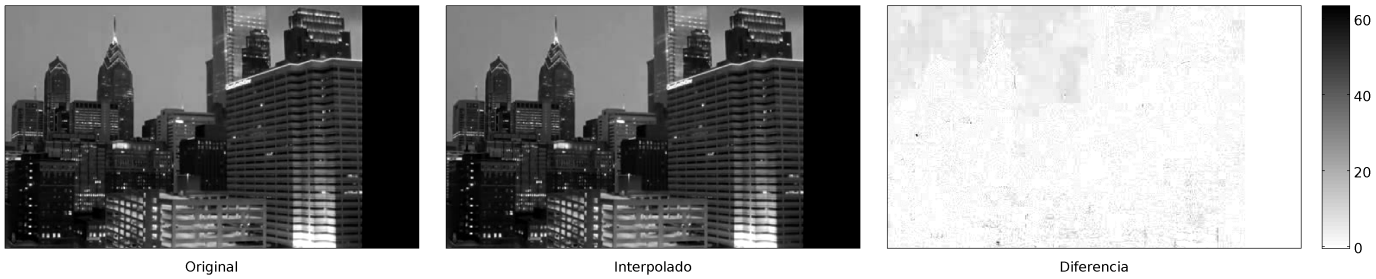
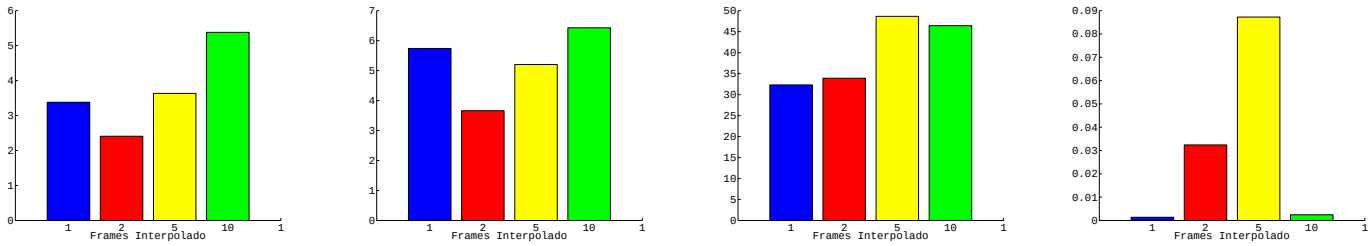


Figura 13. Región peor aproximada por la interpolación lineal

El análisis de este video comparativo, y las conclusiones, terminan siendo exactamente igual que para el caso de splines de este mismo video. Se puede observar como lo que afecta al error es la iluminación, y que las partes más afectadas por la misma (el cielo y las partes altas de los edificios durante el día, y los sectores cercanos a las fuentes de luz artificial durante la noche) son aquellas que mayor error (diferencia) introducen. No queda mucho más que agregar a lo ya dicho para el método de splines.

4.6.3. Vecino más Cercano

La metodología de experimentación pare este método no difiere respecto de los 2 métodos previos. Así pues, ahorrando explicaciones, se pasa a exponer los resultados obtenidos de la experimentación.



(a) Valor Medio (b) Desvío Estándar (c) Máximo (d) Mínimo

Figura 14. Estádisticas ECM Según Frames Interpolados - Método Vecino Más Cercano

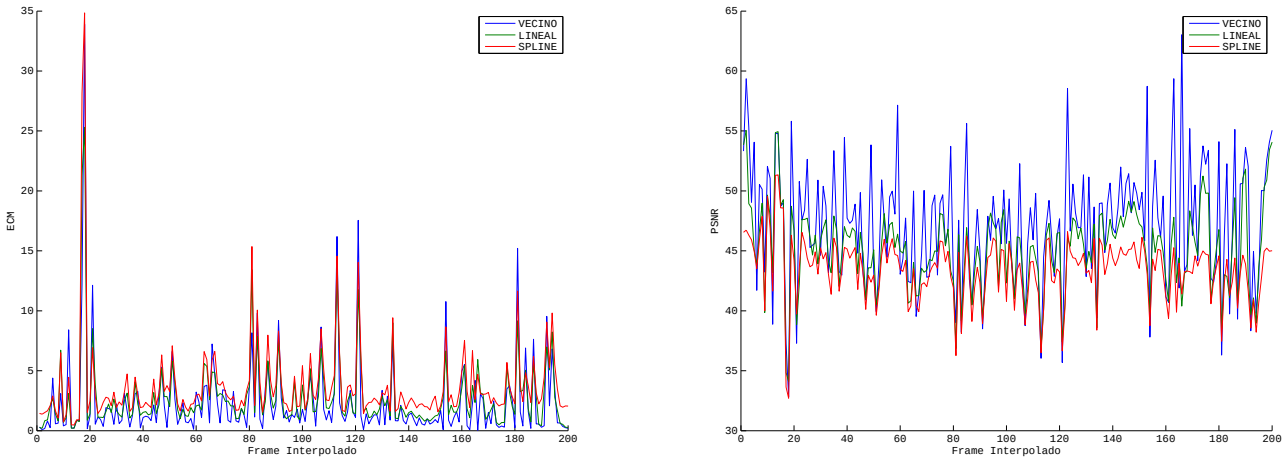
En los resultados expuestos en la figura 14 se encuentran relaciones entre las distintas variantes de frames interpolados, y con números bastantes cercanos también, que al caso de splines. La diferencia destacable radica en los valores de error mínimo alcanzado para las variantes de interpolación de 1 y 10 frames, que como se puede observar son mucho más chicos que para los parámetros restantes. Aún así, una vista detallada de los valores nominales nos muestra que a pesar de ser estos muy pequeños, ya se está teniendo errores mínimos de valor nominal muy pequeño, haciendo que esta diferencia entre distintas variantes del método sean poco importantes. Es decir, tanto como si se interpolan 1, 2, 5 o 10 frames, pareciera que siempre habrá algún frame interpolado con prácticamente 0 error de estimación.

Respecto del análisis del PSNR, y del video de comparación, los resultados observados son exactamente idénticos a los explicados en el caso del método lineal para este video.

4.6.4. Análisis entre Métodos

Luego de haber analizado cada método independientemente de los demás, basándonos únicamente en sus parámetros, pasamos a comparar cada método cualitativamente respecto de los otros, siempre para el mismo video.

En la figura 15 se puede observar el ECM y PSNR frame a frame para el caso de 2 frames interpolados de cáda uno de los métodos. En el caso del método de splines, se expuso los resultados para la interpolación del video completo (sin dividir por bloques), ya que como se vió en su respectivo análisis, el comportamiento era similar no importa el tamaño del bloque.



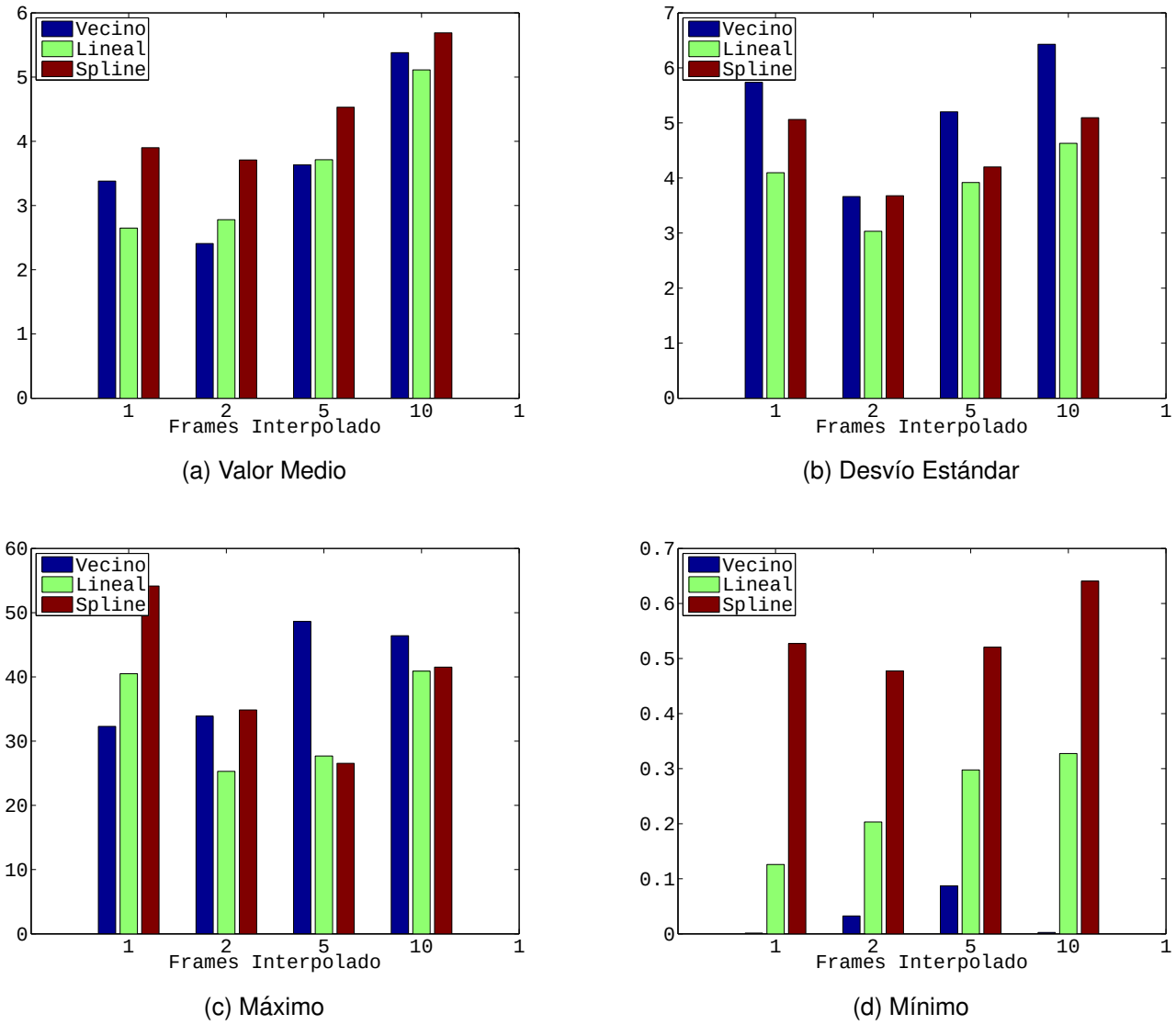
(a) ECM para 2 frames interpolados (b) PSNR para 2 frames interpolados

Figura 15. Comparativa de métodos para 2 frames interpolados

En ambos gráficos (aunque más claramente en el PSNR) se observa un patrón claro (más allá de que haya frames que sean excepciones): la fidelidad es mejor (o el error es menor) para el método del vecino más cercano, seguido por la interpolación lineal y por último el método de splines.

No se exponen estas comparativas para otras variantes de interpolación, ya que este patrón se repite en todos los casos. Además, vale la pena recordar que para la cantidad de frames interpolados igual a 2 es para el cual se obtenía el menor ECM medio para los métodos de splines y vecino, y virtualmente el menor también para la interpolación lineal (la diferencia para frames interpolados igual a 1 es mínima).

Pasemos a observar a continuación, las comparaciones para los valores estadísticos del error cuadrático medio:



(a) Valor Medio

(b) Desvío Estándar

(c) Máximo

(d) Mínimo

Figura 16. Estádisticas ECM - Comparativa de Métodos

Al observarse estos datos, aparece algo que *a priori* parecería una contradicción: el valor medio del ECM para el método lineal es menor en 2 de las variantes (cantidad de frames 1 y 10) que el método del vecino más cercano. Analizando un poco esto (pues, como afirmamos unos párrafos más arriba, para todas las variantes encontramos que el PSNR para el método del vecino más cercano seguía el patró de ser más alto que el de los restantes métodos), entendemos que el motivo de esto se debe a que el método del vecino tiene picos más amplios donde comete mucho más error que el método de interpolación lineal. Esto puede observarse, nuevamente, en la figura 15. Esto explica el porque del valor medio del error es más bajo para el método lineal, aunque en la gran mayoría de los frames la aproximación del método del vecino es mejor.

4.6.5. Conclusiones

A lo largo de la experimentación para el tipo de video *cámara-fija, imágen fija*, observamos que todos los métodos mostraron un buen comportamiento. Esta afirmación es subjetiva y basada en la opinión de los autores luego de

observar los videos resultantes. Para el ojo humano, las diferencias entre los métodos (o para el mismo método con diferentes parámetros) fueron imperceptibles.

Aún así, un análisis más objetivo basado en las métricas nos llegó a indicar que, al menos para los parámetros evaluados, el método del vecino pareciera ser mejor, incluso cuando se interpolan muchos frames (recordemos que una las hipótesis al comienzo de la experimentación era que los métodos de spline e interpolación lineal deberían ser mejores en caso de tener que estimar/reconstruir muchos frames). Si bien su ECM medio es más alto que el de otros métodos, pudimos observar que también lo es su desvío estándar. Y justamente esto último es lo que nos permitió descubrir que el motivo de su ECM medio más alto se debe a que, en determinados frames (los más alejados de los 2 frames utilizados para interpolar, de hecho) el método tiene un error más alto que el resto. Como esto último ocurre seguido, el ECM tiende a subir, pero al observar el PSNR frame a frame se pudo concluir que en realidad, para la gran mayoría de los frames interpolados, el método del vecino superaba a sus pares.

Esto último no es un resultado menor, más si se tiene en cuenta que el método del vecino es el más barato computacionalmente hablando, como ya se ha visto.

En otra línea de análisis, también se observó que al menos para este tipo de video, que el tamaño de bloque utilizado no hizo variar a la efectividad del método de spline. Esto, si bien no esperado al comenzar los experimentos, es razonable si se considera que en si mismo el video no presenta casi cambios, más allá del brillo de sus puntos debido al cambio de iluminación. Tiene sentido entonces que el resultado de splines no varíe ya que "la función interpolada"[a] cambían en función del frame de manera "suave", con lo cual no importe desde que frame a que frame se genere el spline, sus valores en los extremos de los bloques tenderían a ser similiares a lo que se les exije en el método a los *sub-splines*.

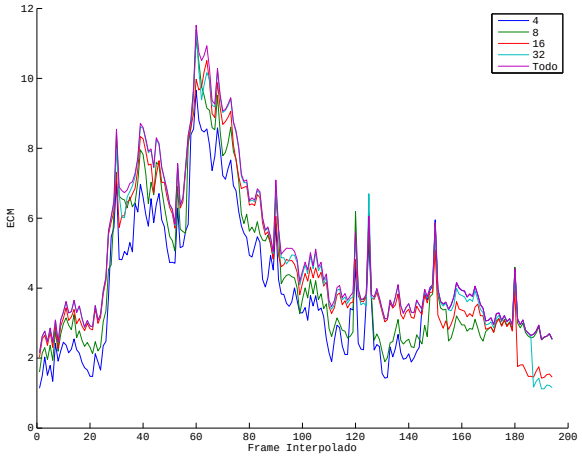
Para finalizar este experimento, vale la pena remarcar nuevamente que si bien la interpolación de todos los métodos tuvieron error, el mismo siempre fue muy pequeño nominalmente. De hecho, como se mencionó al principio de estas conclusiones, fueron imperceptibles para los autores. Si se observan los errores máximos obtenidos de todos los métodos, observamos que el error más grosero cometido fue de 60, el cual quedará en evidencia que es un valor pequeño al compararlo con los errores de los métodos en los otros tipos de videos que se exponen más adelante. Más aún, si se observar los videos comparativos de las interpolaciones, se observará que el *heatmap* de la diferencia absoluta entre el frame original y el interpolado nunca tiene regiones enteras con mucho (o incluso, más que poco) error. A lo sumo existen pequeños grupos de no más de 10 píxeles que muestran un error alto, pero nunca regiones.

a. Los pixels a lo largo de los frames.

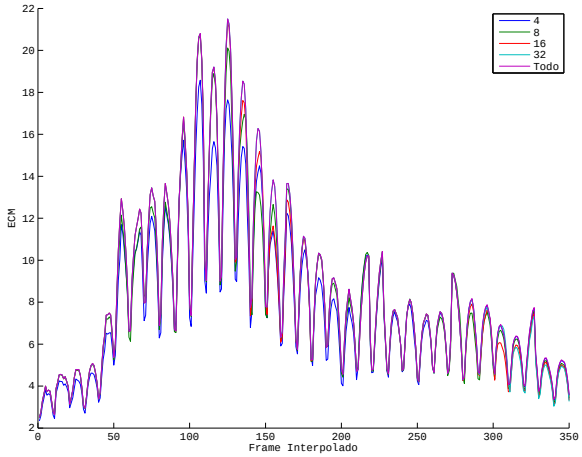
4.7. Cámara Fija - Imágen Móvil

HABIENDO ya realizado detalladamente el anásis propuesto por la metdología explicada en la sección 4.3 para el caso de *cámara fija, imágen fija* (sección 4.6), no ahondaremos tanto en detalles sobre la explicación de los datos obtenidos. Los gráficos, tablas e histogramas utilizados son los mismos para este caso y los que le siguen[a]. Simplemente expondremos los resultados y los analizaremos.

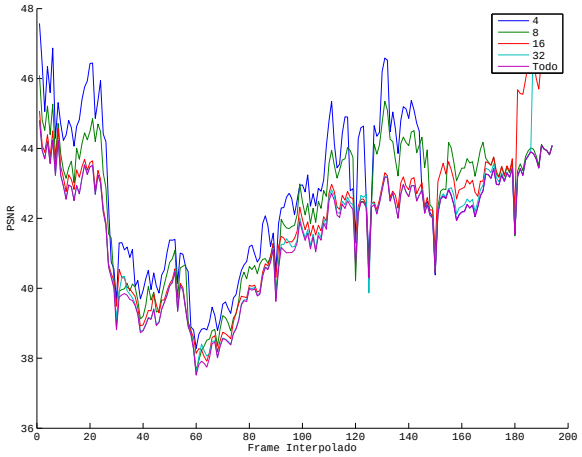
4.7.1. Spline



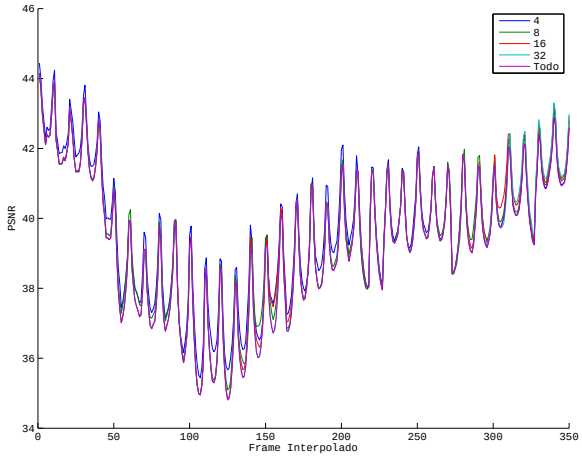
(a) ECM para 1 frames interpolados



(b) ECM para 10 frames interpolados



(c) PSNR para 1 frames interpolados



(d) PSNR para 10 frames interpolados

Figura 17. Comparativa tamaño de bloque para 1 y 10 frames interpolados

En primer lugar, al igual que para el experimento previo, se exponen sólo los gráficos comparativos de tamaño de bloque del ECM por frame para 1 y 10 frames interpolados (figura 17). No se exponen los casos para 2 y 5 frames interpolados ya que el comportamiento observado se refleja de la misma manera en estos, y para explicar el mismo alcanza con las variantes expuestas.

Lo que se observa inicialmente de la figura 17a respecto del tamaño de bloque, es que para este video el tamaño de bloque 4 pareciera ser significativamente mejor[b] que para los demás tamaños (más allá de algunos pocos frames

a. Cámara móvil, imágen fija y móvil.
b. Menos error.

donde esto no ocurre). Más aún, en términos generales más allá de frames excepcionales, se observa que el ECM aumenta a medida que aumenta el tamaño de bloque.

Sin embargo, esto no pareciera ser tan claro para el caso de 10 frames interpolados (figura 17b). En éste se puede distinguir que el tamaño de bloque 4 suele tener menos error que el resto de los tamaños, aunque la diferencia es notoriamente menor que para el caso de 1 frame interpolado. Y, áun más, no queda claro a simple vista que el error vaya aumentando a medida que aumenta el tamaño de bloque (solapamiento de los ECM), aunque sí podemos afirmar que la interpolación por spline para todo el video (sin dividir en bloques) es el que suele tener para la gran mayoría de los frames el ECM más alto. Este solapamiento no queda del todo claro en esta figura, principalmente dado que la escala difiere respecto de la figura 17a, pero queda ya claro al observar los gráficos de PSNR, que tienen la particularidad de normalizar respecto del valor máximo de los píxeles los valores del ECM.

Más aún, si tenemos en cuenta el ECM/PSNR para 2 y 5 frames interpolados, el patrón que se observa es que a medida que se interpolan más frames se incrementa el solapamiento, es decir, se "pegan" las funciones de ECM para los distintos tamaños de bloque. Aunque, como parte de este patrón, pareciera que a menor tamaño de bloque se sigue comientiendo menos error.

Básicamente, se observa un patrón: a menor tamaño de bloque menor error cometido, aunque esta diferencia a favor de una interpolación de menor tamaño de bloque se va diluyendo a medida que se incrementa la cantidad de frames interpolados.

Si nos detenemos en los cuadros de la figura 18, se observará que se confirmá este patrón[a]. Se puede observar aquí que para cualesquiera 2 tamaños de bloque distintos, la máxima diferencia del ECM para un mismo frame siempre es mayor para el tamaño de bloque mayor. Por ejemplo, si tomamos los tamaños 4 y 8, veremos que siempre 4 vs 8 (la diferencia máxima para la cual el ECM de 4 es mayor al de 8) es mayor a 8 vs 4.

Bloque	vs 4	vs 8	vs 16	vs 32	vs Entero
4	0	1.0553	1.3580	1.5157	0.0836
8	2.2855	0	1.3654	1.4776	0.5794
16	2.4014	1.8014	0	0.3766	0.0696
32	2.5348	1.8834	1.4168	0	0.6421
Entero	2.5429	1.9027	1.5443	1.5157	0

(a) 1 Frame Interpolado

Bloque	vs 4	vs 8	vs 16	vs 32	vs Entero
4	0	1.0587	1.4019	1.5831	0.5797
8	2.6425	0	1.1953	1.4636	0.7106
16	2.7262	1.6141	0	0.3542	0.0415
32	2.7816	1.7799	1.3980	0	0.0566
Entero	2.7875	1.7952	1.4049	1.5831	0

(b) 2 Frames Interpolados

Bloque	vs 4	vs 8	vs 16	vs 32	vs Entero
4	0	0.8709	0.6384	1.8067	0.6342
8	3.4411	0	0.6755	1.6095	0.6634
16	3.5990	3.2710	0	1.4143	0.1297
32	3.6145	3.3025	0.9938	0	0.0607
Entero	3.6162	3.3024	1.0310	1.8448	0

(c) 5 Frames Interpolados

Bloque	vs 4	vs 8	vs 16	vs 32	vs Entero
4	0	1.5891	0.8529	0.6327	0.5407
8	3.2999	0	1.5430	0.7919	0.7928
16	3.6673	2.3288	0	0.5000	0.5016
32	3.8571	3.0513	2.4365	0	0.2676
Entero	3.8574	3.0504	2.4367	0.6565	0

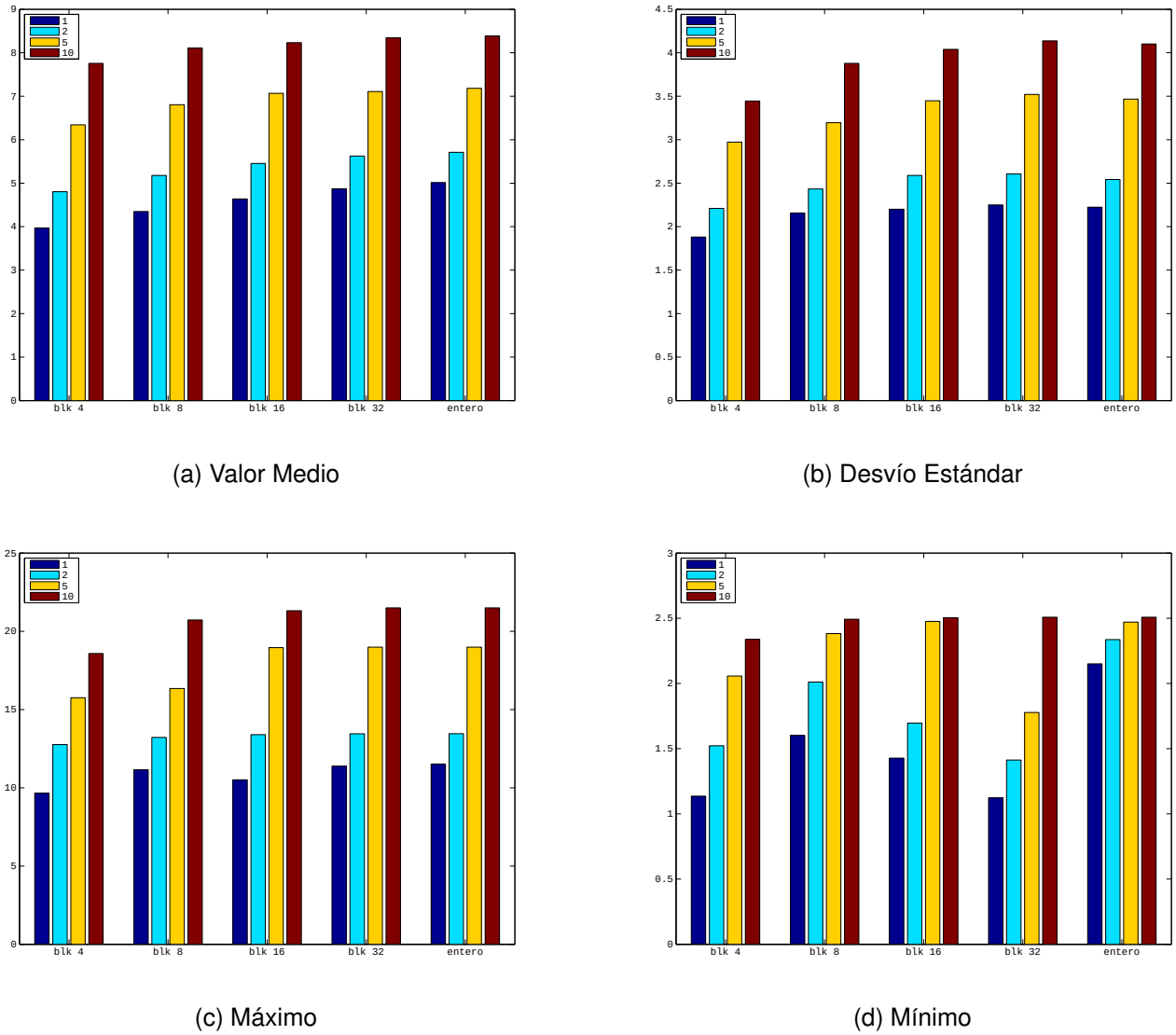
(d) 10 Frames Interpolados

Figura 18. Diferencia máxima de ECM para mismo frame según tamaño de bloque

Ahora bien, siguiendo esta línea de análisis, se observa (salvo alguna excepción) otro patrón: a medida que se interpolan más frames, las diferencias máximas del ECM incrementarse para los casos de tamaño i vs j con $i > j$, y decrementarse para $i < j$. Es decir, mientras que en el gráfico 17d se observa un mayor solapamiento del PSNR, en las tablas de la figura 18 vemos que las diferencias máximas (o picos) entre dos interpolaciones con distinto tamaño de bloque se incrementan a mayor cantidad de frames interpolados. Es decir, si bien en la media los tamaños de bloque están cada vez más cerca a mayor cantidad de frames interpolados, los errores máximos cometidos por las interpolaciones con mayor tamaño de bloque respecto de los de menor tamaño se incrementan (es decir, mayores

a. A modo de recordatorio, cada cuadro nos indica la diferencia máxima del error para un mismo frame interpolado. Básicamente, en la posición i, j, tendremos la diferencia máxima de las estimaciones sobre las cuales el tamaño de bloque i cometió un error mayor que tamaño j.

picos de error entre distintos tamaños[a]).



(a) Valor Medio

(b) Desvío Estándar

(c) Máximo

(d) Mínimo

Figura 19. Estádisticas ECM Según Frames Interpolados - Spline

En las estadísticas expuestas en la figura 19 se observa otra confirmación del patrón mencionado. Se puede ver que la media del ECM para una misma cantidad de frames interpolados (barras del mismo color) es menor a menor tamaño de bloque (aunque las diferencias no son tan significativas entre tamaños consecutivos). Y también se observa que esta media es menor a menor cantidad de frames interpolados, lo que es consistente con nuestras hipótesis iniciales.

También observamos que este patrón se mantiene en el caso del desvío estándar de los datos. Es decir, hay una varianza menor para a menor cantidad de frames interpolados, y ligeramente menor para una misma cantidad de frames interpolados pero con tamaños de bloque menores.

Otra cosa que destaca de la figura 17 es que se observa una mayor cantidad de errores (o menor PSNR) en los primeros frames (el primer cuarto de video). Analizando el video comparativo[b] de las diferencias entre frames originales e interpolados para tamaño de bloque 4 (menor ECM medio) y 10 frames interpolados (mayor varianza

a. A no confundir con los picos de error absoluto. Aquí nos referimos a los errores máximos entre dos interpolaciones de la misma cantidad de frames pero distinto tamaño se incrementa a mayor cantidad de frames interpolados.

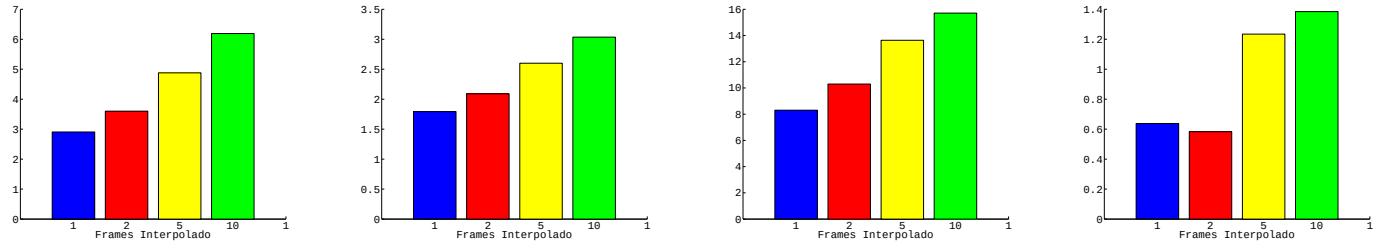
b. https://drive.google.com/open?id=0B0RfkWV-4-XqSHJ3NXBSRUo4SjQ

dentro de las variantes de cantidad de frames interpolados)[a], observamos que al comienzo del video (una jugada de un partido de fútbol desde un punto fijo) tenemos un jugador que comienza a correr en un primer plano, mientras que el resto de los jugadores en movimiento se encuentran en un plano cada vez más lejano. En la figura 4.7.1 se observa una captura del video comparador representativo de esto.

Figura 20. Región peor aproximada por la interpolación por spline

En el video comparador queda claro que los movimientos de los jugadores son los que más error aportan a la interpolación. Y no sólo eso, sino que el hecho de que haya jugadores en planos más cercanos a la cámara generan un mayor error en la interpolación, ya que dichos jugadores ”ocupan” una mayor cantidad de pixeles de los frames que aquellos que se encuentran en planos más lejanos. Este jugador que comienza en un plano más cercano, al comenzar a pasar a planos cada vez más lejanos comienza a ser representado por cada vez menos píxeles, y así entonces cada vez disminuye su injerencia en el error de la interpolación (menos pixeles interpolados con cambios debidos al desplazamiento de este jugador). Y esto queda evidenciado en las primeras figuras donde observamos que en el último cuarto de video/frames los picos del ECM son menores (o que los del PSNR son cada vez mayores).

4.7.2. Interpolación Lineal



(a) Valor Medio (b) Desvío Estándar (c) Máximo (d) Mínimo

Figura 21. Estádisticas ECM Según Frames Interpolados - Método Lineal

Observando las estádisticas del método para las distintas variantes de la cantidad de frames interpolados, se hayan resultados que son consistentes con las hipótesis iniciales enunciadas. Se observa como el ECM medio es cada vez mayor a medida que se interpolan más frames (es decir, se utilizan 2 frames para interpolar una cantidad cada vez mayor de frames entre ambos).

Además también se observa que la varianza del error de la aproximación de la interpolación aumenta conforme aumenta la cantidad de frames interpolados (lo cual es de alguna manera razonable, al interpolarse más frames, los frames interpolados más cercanos a los frames utilizados en la interpolación tendrán menor error, mientras que a mayor distancia de estos más error habrá, generando multiples valores de error distintos y distantes entre sí -y de la media-). Este último comportamiento se puede observar más claramente en la figura 22.

Aquí se puede observar como el PSNR[b] tiene un comportamiento oscilante, yendo de un valor ”alto” de PSNR a uno bajo y luego comenzando a ascender nuevamente hasta llegar a otro pico ”alto”.

a. La elección de estos parámetros para le generación del video de diferencias frame a frame se basa en los mismos argumentos que en el caso de splines del experimento previo.

b. Recordemos que el PSNR, a diferencia del ECM, nos da una noción de la calidad de la estimación del frame. Es decir, a mayor PSNR, menor ECM (aunque la relación no es lineal, ya que el PSNR normaliza la medida según el valor máximo de los píxeles).

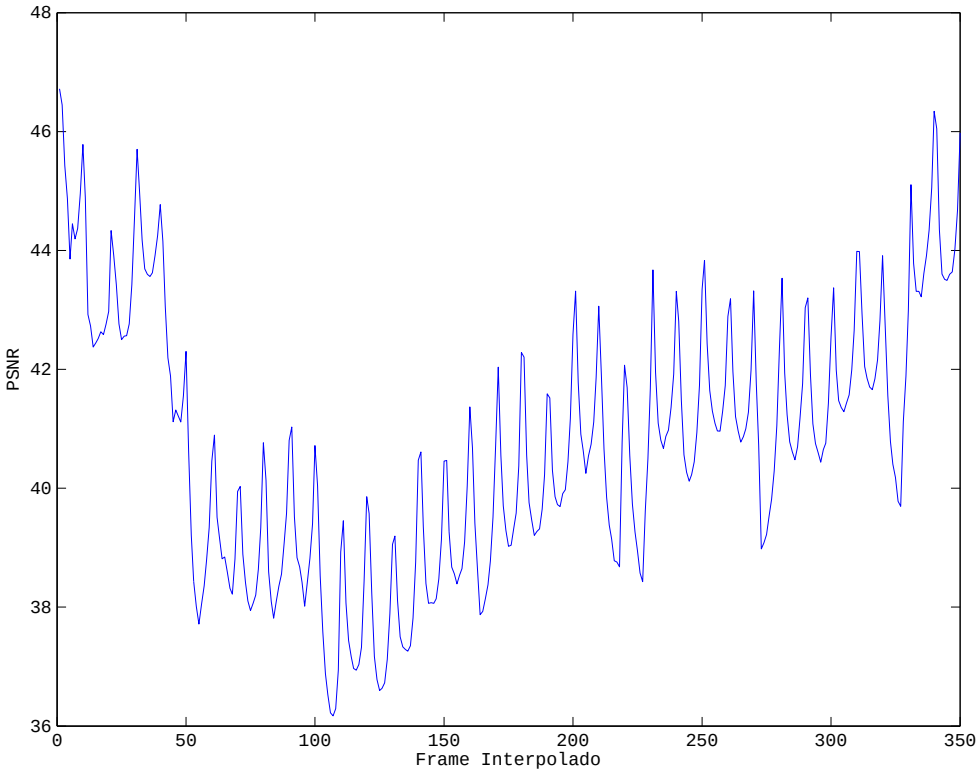
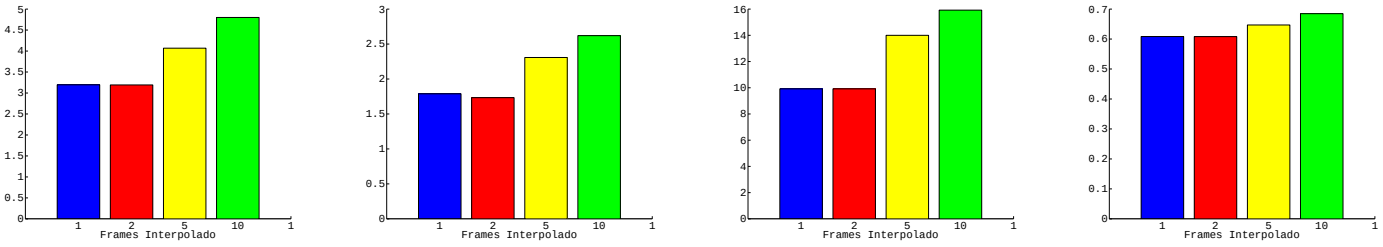


Figura 22. PSNR para 10 frames interpolados - Método Lineal

De la observación se determinó que estos picos ”altos” son los frames interpolados inmediatos a los frames originales utilizados para calclular la funcion interpoladora, y no es coincidencia que los picos ”bajos” (donde hay mayor error en la interpolación) sean justamente los frames intermedios entre dos frames utilizados para calcular el polinomio interpolador lineal.

Pasando a analizar el video comparador para 10 frames interpolados[a], observamos el mismo comportamiento visto para el caso de splines, aunque los errores cometidos para las reginoes con poco (o ningún) movimiento parecerían ser menores que para splines. En todo caso, esto se estudiará al final de esta sección al comparar los distintos métodos entre sí.

4.7.3. Vecino más Cercano



(a) Valor Medio (b) Desvío Estándar (c) Máximo (d) Mínimo

Figura 23. Estádisticas ECM Según Frames Interpolados - Método Vecino Más Cercano

a. https://drive.google.com/open?id=0B0RfkWV-4-XqMHNrY3JqVHlOUXc

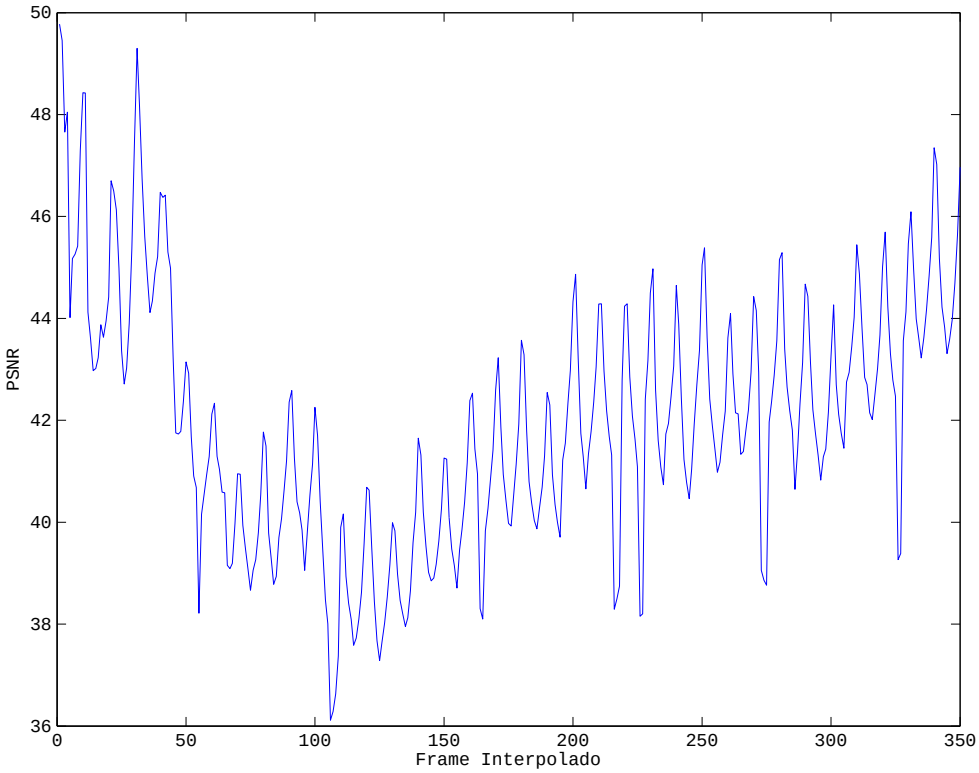


Figura 24. PSNR para 10 frames interpolados - Vecino más Cercano

Nuevamente, para este método y este video, encontramos resultados consistentes con nuestras hipótesis. A mayor cantidad de frames interpolados, mayor es el error medio. De hecho, los resultados de este método son muy similares a los de la interpolación lineal (en cuanto a la relación del mismo método para distintos frames interpolados, no en cuanto a los valores de ECM de vecino más cercano contra interpolación lineal). Aunque, salta a la vista que el desvío estándar para la interpolación de 2 frames es ligeramente menor que para la de 1 frame.

Voliendo sobre los conceptos del método, es de alguna manera razonable explicar esto debido a que al tener 2 frames interpolados por cada par consecutivo de frames del video original, cada uno de los interpolados será una copia exacta de su frame original más cercano. De esta manera, deberían haber movimientos/cambios muy bruscos en los frames originales (como por ejemplo, un cambio de cámara si se estuviera trabajando con el video de una transimición de televisión o una película) de manera que el frame copiado/interpolado fuera completamente distinto al frame original al que debería aproximar. Dado que esto no ocurre en nuestro video, es razonable pensar que la diferencia de un frame a otro no tenga grandes diferencias, de manera que al utilizar una copia del vecino más cercano para una interpolación de pocos frames no tenga demasiado error ni mucha varianza (ya que el error de todos los frames interpolados/copiados es la diferencia de 2 frames consecutivos del video original, los cuales para un video sin cambios bruscos deberia ser similar para todos los frames).

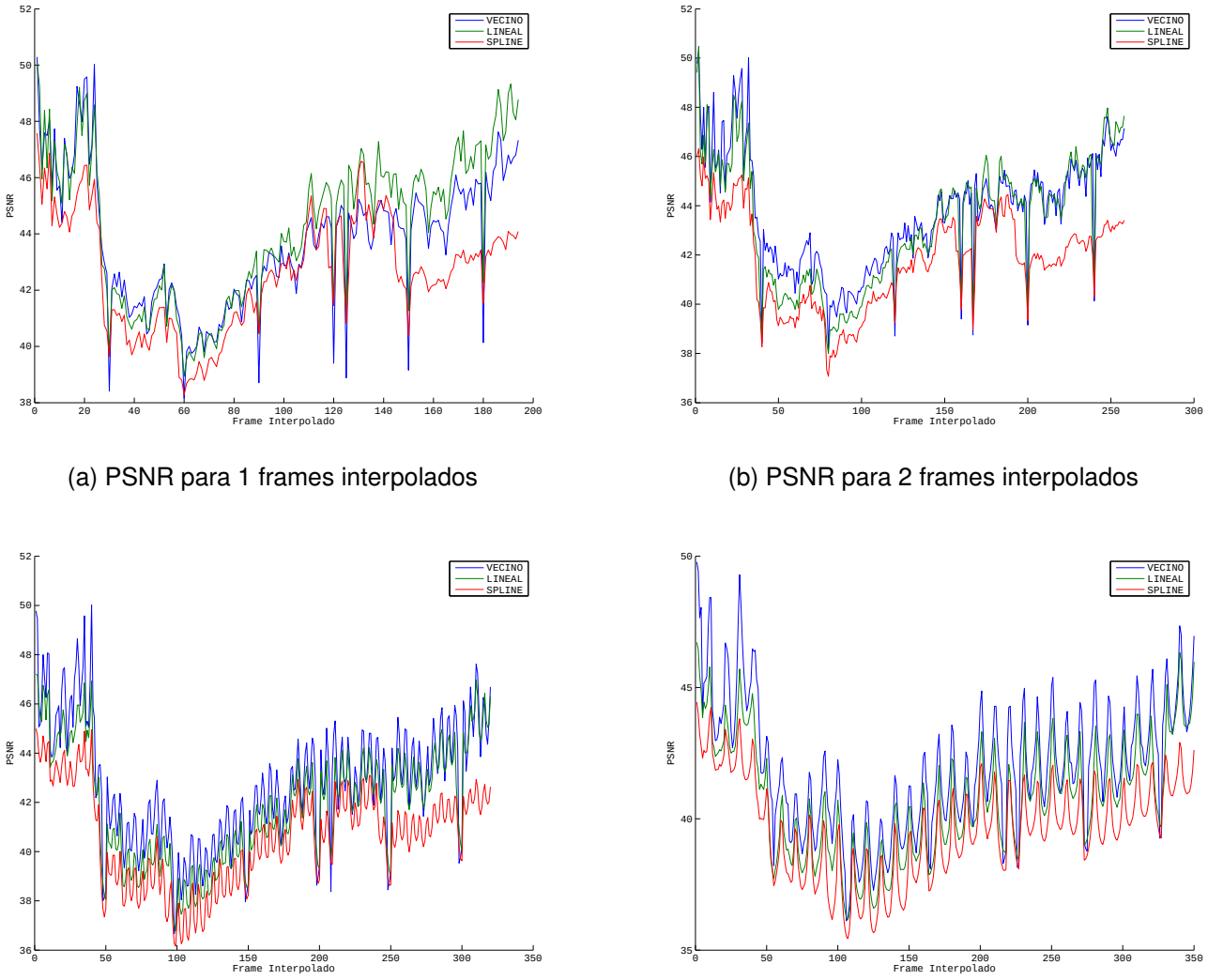
Analizando el video comparativo para este método[a], además de observar el mismo comportamiento respecto de los movimientos/planos donde se genera el error en la interpolación, observamos que hay saltos entre frames donde el error aparece ”repentinamente”. Obviamente esto se debe a el cambio de un frame a otro sin ningún intento de aproximar lo que ocurre en el medio. Esto no ocurría para el experimento previo, ya que ahora tenemos movimientos cuando antes sólo teniamos cambios de luz/tonalidad. Si bien no podemos exponer en este documento en forma de gráfico esta percepción, si podemos presentar el gráfico del PSNR frame a frame, donde se puede observar los picos de muy pronunciados de pérdida de precisión (o aumento del error) que ocurren de manera cíclica (con esto nos referímos a que ocurren a un intervalo regular de frames, no a que su valor PSNR sea el mismo).

Aquí lo que se nota de distinto respecto de los otros métodos, que por ahí no es tanto el intervalo de los picos

a. https://drive.google.com/open?id=0B0RfkWV-4-XqY1gtZDFqQ0puaTg

de error (bajo PSNR), es el hecho de que los mismos son abruptos (pasan de un PSNR alto a un PSNR bajo de manera muy rápida), mientras que en los casos anteriores observamos como estos cambios se dan de forma más ”suave”: es decir, graficamente hablando, de una manera no tan ”recta” sino más curva (ver figura 22).

4.7.4. Análisis entre Métodos



(a) PSNR para 1 frames interpolados

(b) PSNR para 2 frames interpolados

(c) PSNR para 5 frames interpolados

(d) PSNR para 10 frames interpolados

Figura 25. Comparativa de métodos

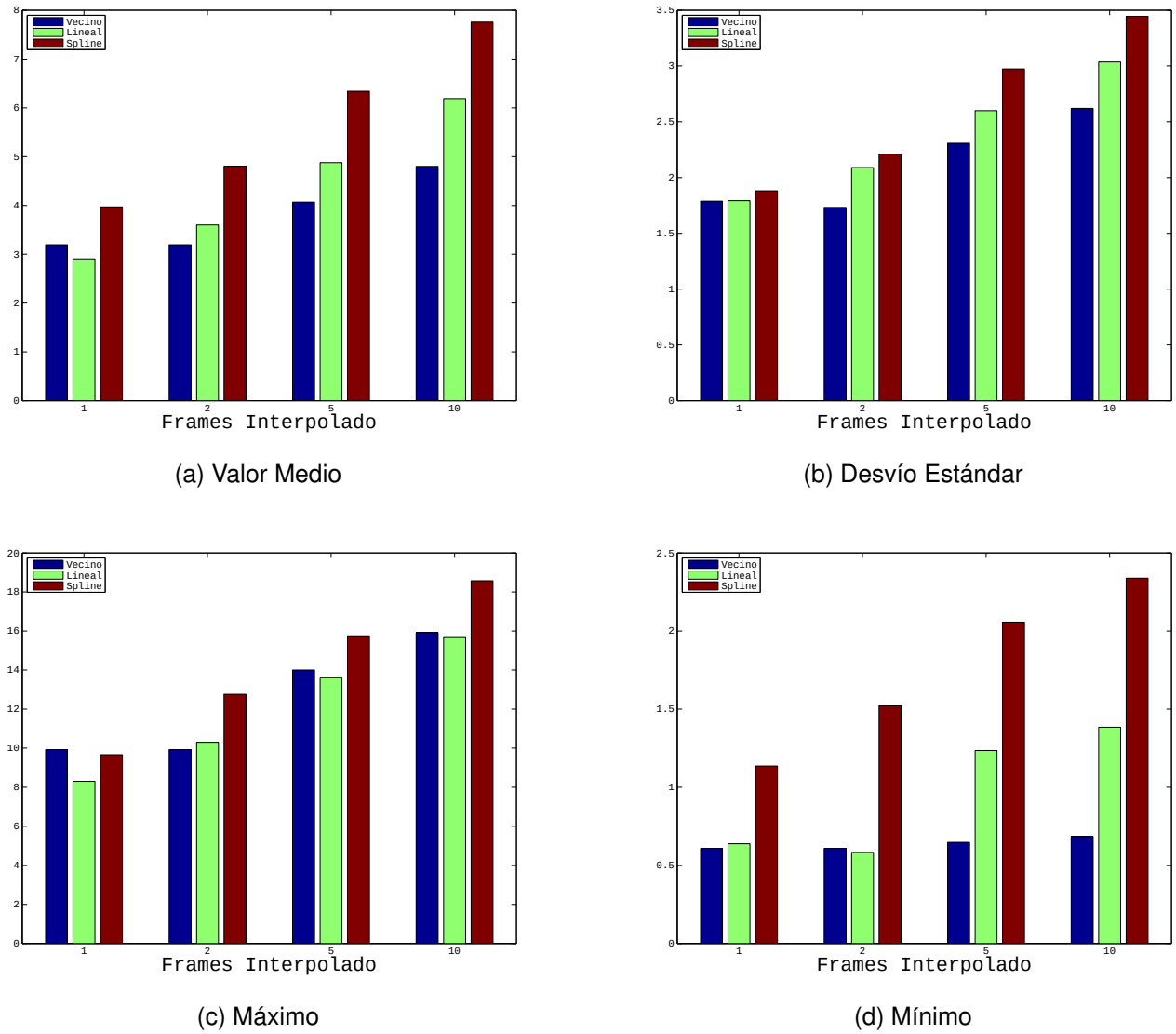
En la figura 25 se presentan el PSNR para las distintas variantes de cantidad de frames interpolados, comparando los métodos que objeto de estudio de este trabajo. En el caso particular del método de spline, se tomó el tamaño de bloque 4, habiéndose visto que este era el que menor error presentaba para cualquiera de las variantes de cantidad de frames interpodos.

Para todos las variantes expuestas, el primer resultado obvio es que spline parecierá ser el método que pero estima los frames. Se observa que para todos los frames interpolados, es siempre el método de menor PSNR. Y, como ya se ha visto, splines es el método más costoso de los 3 que están siendo estudiados, con lo cual este resultado ya nos permite proponer que a la hora de utilizar este tipo de videos (cámara fija, imágen móvil -sin movimientos bruscos-) splines no parece ser un método a ser considerado[a].

a. Quizás si evaluásemos variantes donde se interpolen una mayor cantidad de frames, estos resultados podrían variar. Pero ya tener una interpolación de 10 frames es considerable, sumado las limitantes de tiempo para experimentar con valores más altos.

Queda entonces evaluar que ocurre con los métodos del vecino e interpolación lineal. Observando las figuras 25a y 25b, se ve que en la primera mitad del video el PSNR de ambos métodos es similar, para luego en la segunda mitad obtener la interpolación lineal mejores resultados (en el caso de 2 frames interpolados, esta ventaja es mínima y quizás desestimable). En los casos restantes, es claro que durante todo el video el método del vecino estima de mejor manera los frames reconstruídos. No es casualidad que, en los casos donde la interpolación lineal supera al método del vecino, se de justo en el momento en que aparece un jugador en el primer plano del video (alrededor del frame 110). Pareciera que al interpolar pocos frames y haber movimiento en un área considerable de los frames el método lineal se comporta mejor que el resto. Lo sorpresivo es, dadas nuestras hipótesis, que al aumentar la cantidad de frames el PSNR sea mucho mejor para el metodo del vecino.

Si observamos las estadísticas del ECM de la figura 26, veremos que los resultados reflejan este mismo comportamiento (bien podría, debido a un desvío estándar alto o una cantidad de *outliers* suficiente, no hacerlo). Se observa que para pocos frames el ECM medio del métdo lineal es menor, y que a medida que aumenta la cantidad de frames estimados el método del vecino comienza a tener un ECM medio más bajo relativo a los otros dos métodos. Incluso esto mismo ocurre con el desvío estándar (aunque la diferencia relativa que va "ganando" el método del vecino crece a menor ratio).



(a) Valor Medio

(b) Desvío Estándar

(c) Máximo

(d) Mínimo

Figura 26. Estádisticas ECM - Comparativa de Métodos

Continuando el análisis de estos datos, se ve que el ECM mínimo para el método del vecino se ve casi inalterado sin importar la cantidad de frames que se interpolen. Esto tiene sentido pues dado un frame, su frame interpolado

inmediato en la reproducción será siempre el mismo sin importar cuantos otros frames se copien hasta ”llegar” al siguiente frame no-interpolado/original. Por otro lado, si vemos que el comportamiento de los métodos restantes comienza cada vez a tener una peor cota para el frame mejor aproximado a medida que se estima una mayor cantidad de frames. Claramente estos métodos son mucho más sensibles a la ”distancia” entre los frames originales utilizados para calcular su polinomio interpolador[a].

En cuanto al frame peor estimado (máximo ECM) observamos que todas las técnicas evaluadas comienzan a tener un frame cada vez peor aproximado. Esto, de hecho, tiene sentido, ya que a medida que se incrementa la cantidad de frames interpolados se deben generar artificialmente más cuadros con menos información (frames originales más ”distantes” en la reproducción del video original). Claramente aproximar con menos información reduce las posibilidades de estimar de manera precisa.

Por último, y para finalizar este análisis de los métodos para este video, observamos los videos comparadores de utilizados en los análisis independientes de los métodos para este video. A modo de presentar un resultado en este trabajo, presentamos en la figura 27 3 capturas del mismo frame, que fue interpolado por los 3 métodos.

(a) Spline (tamaño de bloque 4)

(b) Interpolación Lineal

(c) Vecino más Cercano

Figura 27. Captura del mismo frame de Videos comparativos para interpolación de 10 frames

Como se puede observar en las figuras (o en los videos, cuyos *links* pueden encontrarse en las secciones previas) se ve que el movimiento de los jugarores es el principal área de los frames que son estimados de la manera menos precisa por cualquiera de los métodos. Lo intersante es obsevar las áreas de los frames que no presentan movimientos (o que son imperceptibles o claramente de menor intensidad que los pertenecientes a los jugadores). Así se puede observar que el método de splines comete errores pequeños a lo largo de todo el cuadro, mientras que el método lineal lo hace en menor medida y el vecino más cercano aún menos.

a. Recordar que en nuestra experimentación se está tomando un video y se le remueven frames que luego serán reconstruídos mediante los métodos estudiados.

Esto nos explica, probablemente, porque el método del vecino tiene un ECM menor que los otros dos métodos (o PSNR más alto). Observando las tonalidades de los errores de los jugadores, y el área que ocupan en el frame estos puntos **negros**, observamos que los métodos cometen errores similares a la hora de estimar a los mismos. Por lo tanto, queda claro que es la estimación del resto del frame que hace que un método tenga menor error que otro.

El porque de este comportamiento podría explicarse dado que el método del vecino es más ”bobo” (aunque en este caso pareciera ser una ventaja), ya que no asume nada sobre los pixeles y solo los copia. De esta manera, al tener una gran parte de los cuadros del video que son estáticos (o con poquísimo movimiento) y no llegan a tener cambios de iluminación (como en el caso del experimento previo), logra replicar este estatismo. Los métodos restantes tratan a cada pixel como una función a ser interpolada, con lo cual asumen que entre pixel y pixel siempre habrá algún cambio (salvo que todos los pixeles utilizados para calcular el polinomio interpolador tengan el mismo valor, en cuyo caso el polinomio resultante corresponde a una función constante, pero como en el video estos píxeles sufren cambios muy pequeños esto no ocurre).

4.7.5. Conclusiones

A lo largo de este experimento se ha visto, nuevamente, un patrón de comportamiento para el método de spline según su tamaño de bloque. A diferencia del experimento previo donde se vió que el tamaño parecía no afectar a la precisión de la estimación, se observó que en el caso de este video un tamaño pequeño de bloque resulta más efectivo.

Sorprendente es el resultado de que el método del vecino más cercano estima de mejor manera que sus alternativas (salvo en el caso de pocos frames interpolados, en cuyo caso el método de interpolación lineal podría llegar a ser mejor, dependiendo de las características del video). Más interesante aún es que este comportamiento pudo ser explicado, y es razonable aunque no sea (al menos para los autores) intuitivo.

Ahora bien, habiendo comparado los métodos mediante las métricas y concluído que método resulta mejor para este tipo de casos, queda decir si esto coincide con la percepción subjetiva de quienes observan el video interpolado. Y es aquí donde los resultados obtenidos a partir de las métricas no se condicen con la percepción humana[a].

Es la opinión de los autores que el método del vecino consigue mejores resultados en el caso de interpolar 1 o 2 frames. Pero en el caso de interpolar 10 frames, consideran que el método del vecino es claramente el que peor logra aproximar al video original Esto mismo se da ya que al ver el video, el mismo pareciera tener algún tipo de error de reproducción o *freeze*, debido a que se queda ”congelado” en varios cuadros y da la sensación de que muchos cuadros no se llegan a reproducir por algún tipo de problema (que no existe, ya que lo que ocurres es que se está reproduciendo el mismo frame unas 5 veces, ya que el método copio el mismo). En este caso, la percepción del método lineal o de splines es similar, con lo cual a la hora de elegir se sugeriría elegir el método de interpolación lineal ya que es menos costoso que splines, en cuánto al tiempo de cómputo requerido para procesarlo.

Así pues, se ha llegado a un caso donde los resultados de laboratorio basados en métricas objetivas se contraponen con la percepción de los espectadores. Claramente los videos tienen otras variables que hacen a la percepción humana que nuestras métricas no logran reflejar, justificando para futuros trabajos la búsqueda de métricas o variables que puedan modelar esta percepción subjetiva.

a. Sí, somos humanos los estudiantes de exactas.

4.8. Cámara Móvil - Imágen Fija

4.9. Cámara Móvil - Imágen Móvil

4.10. Análisis por Método en función del Video

4.11. Artifacts

Para comenzar esta sección, consideremos este trabajo en un marco de compresión de video para transmisión en redes de baja velocidad [a].[b] Los videos de entrada pueden verse como los videos `comprimidos` y los videos de salida[c] como los videos `descomprimidos`.

Consideremos la eliminacion de frames intermedios de un video al algoritmo de compresión y a la copia o predicción de frames intermedios como algoritmo de descompresión. Los algoritmos de descompresión pueden clasificarse en los siguientes tipos:

- Métodos predictivos: Interpolación polinomial de frames intermedios, intenta *llenar los huecos* de forma natural entre cada par de frames, intentando predecir el comportamiento del video entre 2 frames conocidos.
- Métodos no predictivos: Por ejemplo, vecino mas cercano, se limita a copiar información, sin interés en regenerar la pérdida provocada al momento de la compresión.

En este contexto, consideraremos *artifacts* a aquellos errores visuales resultantes de la aplicación de un método o técnica. En particular a aquellos que no reflejan una situación verosimil en la vida real[de].

Dividiremos nuestro analisis en diferentes casos, en funcion de las siguientes variables:

- Metodo utilizado para la interpolacion
- Tipo Movimiento grabado por la Cámara
- Tipo de Movimiento de la Cámara
- En el caso de splines, se considera además el tamaño del bloque utilizado

4.12. Interpolacion por vecino mas cercano

Dado que este método de *reconstrucción* es un método *no predictivo* no se producirán artifacts bajo nuestra definición, simplemente el video descomprimido tendrá un efecto de `lag` en donde la reproducción parece trabada. Esto es esperable, ya que este método copia frames consecutivamente para lograr un efecto de mayor tiempo de visualización por frame en la reproducción en tiempo real.

4.13. Interpolacion lineal

4.13.1. Cámara fija e imagen fija - Caso de laboratorio

En nuestro caso de laboratorio [f] puede observarse que al ser todos los pares de frames iguales los frames interpolados linealmente tambien lo serán ya que la recta que une a todos los pixeles entre los 2 frames consecutivos es una constante. No se observan artifacts.

a. Consideramos que la motivación de este trabajo, que la empresa `youborn` quiera transmitir videos en slowmotion pero enviando videos originales, es análogo a transmitir un video comprimido, para luego reconstruir el original.

b. Nuestro algoritmo de compresion consiste en quitarle frames al video original y luego regenerarlos, ya sea de forma predictiva o no predictiva.

c. Los videos en `cámara lenta`

d. Esto es muy subjetivo, pero no me explayo mas porque bordea la filosofía.

e. Por ejemplo la pierna fantasma de la slide de la presentacion del tp.

f. Un video conteniendo un único frame repetido muchas veces.

Experimentos a Futuro

5. CONCLUSIONES

A lo largo de este trabajo pudimos vislumbrar las complejidades propias del problema de ordenar una colección en principio desordenada de elementos. Tomamos el ejemplo de las páginas web por un lado y el de las competiciones deportivas por otro, enfocándonos particularmente en el fútbol. Usamos el algoritmo de PageRank y su adaptación GeM para resolver ambos problemas respectivamente, y estudiamos su comportamiento al variar diferentes parámetros de entrada, particularmente el factor de teletransportación α.

APÉNDICE A
ENUNCIADO DEL TRABAJO PRÁCTICO

Métodos Numéricos

Segundo Cuatrimestre 2015

Trabajo Práctico 3

Departamento de Computación

Facultad de Ciencias Exactas y Naturales

Universidad de Buenos Aires

Un juego de niños

Introducción

¿Quién nunca ha visto un video gracioso de bebés? El éxito de esas producciones audiovisuales ha sido tal que el sitio youborn.com es uno de los más visitados diariamente. Los dueños de este gran sitio, encargado de la importantísima tarea de llevar videos graciosos con bebés a todo el mundo, nos ha pedido que mejoremos su sistema de reproducción de videos.

Su objetivo es tener videos en cámara lenta (ya que todos deseamos tener lujo de detalle en las expresiones de los chiquilines en esos videos) pero teniendo en cuenta que las conexiones a internet no necesariamente son capaces de transportar la gran cantidad de datos que implica un video en *slow motion*. La gran idea es minimizar la dependencia de la velocidad de conexión y sólo enviar el video original. Una vez que el usuario recibe esos datos, todo el trabajo de la cámara lenta puede hacerse de modo offline del lado del cliente, optimizando los tiempos de transferencia. Para tal fin utilizaremos técnicas de interpolación, buscando generar, entre cada par de cuadros del video original, otros ficticios que nos ayuden a generar un efecto de slow motion.

Definición del problema y metodología

Para resolver el problema planteado en la sección anterior, se considera el siguiente contexto. Un video está compuesto por cuadros (denominados también *frames* en inglés) donde cada uno de ellos es una imagen. Al reproducirse rápidamente una después de la otra percibimos el efecto de movimiento a partir de tener un "buen frame rate", es decir una alta cantidad de cuadros por segundo o fps (frames per second). Por lo general las tomas de cámara lenta se generan con cámaras que permiten tomar altísimos números de cuadros por segundo, unos 100 o más en comparación con entre 24 y 30 que se utilizan normalmente.

En el caso del trabajo práctico crearemos una cámara lenta sobre un video grabado normalmente. Para ello colocaremos más cuadros entre cada par de cuadros consecutivos del video original de forma que representen la información que debería haber en la transición y reproduciremos el resultado a la misma velocidad que el original. Las imágenes correspondientes a cada cuadro están conformadas por píxeles. En particular, en este trabajo utilizaremos imágenes en escala de grises para disminuir los costos en tiempo necesarios para procesar los datos y simplificar la implementación; sin embargo, la misma idea puede ser utilizada para videos en color.

El objetivo del trabajo es generar, para cada posición (i, j), los valores de los cuadros agregados en función de los cuadros conocidos. Lo que haremos será interpolar en el tiempo y para ello, se propone considerar al menos los siguientes tres métodos de interpolación:

1. *Vecino más cercano:* Consiste en rellenar el nuevo cuadro replicando los valores de los píxeles del cuadro original que se encuentra más cerca.
2. *Interpolación lineal:* Consiste en rellenar los píxeles utilizando interpolaciones lineales entre píxeles de cuadros originales consecutivos.
3. *Interpolación por Splines:* Simliar al anterior, pero considerando interpolar utilizando splines y tomando una cantidad de cuadros mayor. Una alternativa a considerar es tomar la información de bloques de un tamaño fijo (por ejemplo, 4 cuadros, 8 cuadros, etc.), con el tamaño de bloque a ser determinado experimentalmente.

Cada método tiene sus propias características, ventajas y desventajas particulares. Para realizar un análisis cuantitativo, llamamos F al frame del video real (ideal) que deberíamos obtener con nuestro algoritmo, y sea $\bar{F}$ al frame del video efectivamente construido. Consideramos entonces dos medidas, directamente relacionadas entre ellas, como el *Error Cuadrático Medio* (ECM) y *Peak to Signal Noise Ratio* (PSNR), denotados por $\texttt{ECM}(F, \bar{F})$ y

$\texttt{PSNR}(F, \bar{F})$, respectivamente, y definidos como:

$$\texttt{ECM}(F, \bar{F}) = \frac{1}{mn} \sum_{i=1}^{m} \sum_{j=1}^{n} |F_{k_{ij}} - \bar{F}_{k_{ij}}|^2 \tag{1}$$

y

$$\texttt{PSNR}(F, \bar{F}) = 10 \log_{10} \left(\frac{255^2}{\texttt{ECM}(F, \bar{F})} \right). \tag{2}$$

Donde m es la cantidad de filas de píxeles en cada imagen y n es la cantidad de columnas. Esta métrica puede extenderse para todo el video.

En conjunto con los valores obtenidos para estas métricas, es importante además realizar un análisis del tiempo de ejecución de cada método y los denominados *artifacts* que produce cada uno de ellos. Se denominan *artifacts* a aquellos errores visuales resultantes de la aplicación de un método o técnica. La búsqueda de este tipo de errores complementa el estudio cuantitativo mencionado anteriormente incorporando un análisis cualitativo (y eventualmente subjetivo) sobre las imágenes generadas.

Enunciado

Se pide implementar un programa en `C` o `C++` que implemente como mínimo los tres métodos mencionados anteriormente y que dado un video y una cantidad de cuadros a agregar aplique estas técnicas para generar un video de cámara lenta. A su vez, es necesario explicar en detalle cómo se utilizan y aplican los métodos descriptos en 1, 2 y 3 (y todos aquellos otros métodos que decidan considerar opcionalmente) en el contexto propuesto. Los grupos deben a su vez plantear, describir y realizar de forma adecuada los experimentos que consideren pertinentes para la evaluación de los métodos, justificando debidamente las decisiones tomadas y analizando en detalle los resultados obtenidos así como también plantear qué pruebas realizaron para convencerse de que los métodos funcionan correctamente.

Programa y formato de entrada

Se deberán entregar los archivos fuentes que contengan la resolución del trabajo práctico. El ejecutable tomará cuatro parámetros por línea de comando que serán el archivo de entrada, el archivo de salida, el método a ejecutar (0 para vecinos más cercanos, 1 para lineal, 2 para splines y otros números si consideran más métodos) y la cantidad de cuadros a agregar entre cada par del video original.

Tanto el archivo de entrada como el de salida tendrán la siguiente estructura:

- En la primera línea está la cantidad de cuadros que tiene el video (`c`).
- En la segunda línea está el tamaño del cuadro donde el primer número es la cantidad de filas y el segundo es la cantidad de columnas (`height width`).
- En la tercera línea está el framerate del video (`f`).
- A partir de allí siguen las imágenes del video una después de la otra en forma de matriz. Las primeras `height` líneas son las filas de la primera imagen donde cada una tiene `width` números correspondientes a los valores de cada píxel en esa fila. Luego siguen las filas de la siguiente imagen y así sucesivamente.

Además se presentan herramientas en Matlab para transformar videos (la herramienta fue probada con la extensión .avi pero es posible que funcione para otras) en archivos de entrada para el enunciado y archivos de salida en videos para poder observar el resultado visualmente. También se recomienda leer el archivo de README sobre la utilización.

Sobre la entrega

- FORMATO ELECTRÓNICO: Martes 10 de Noviembre de 2015, **hasta las 23:59**, enviando el trabajo (`informe` + `código`) a `metnum.lab@gmail.com`. El `asunto` del email debe comenzar con el texto `[TP3]` seguido de la lista de apellidos de los integrantes del grupo. Ejemplo: `[TP3] Artuso, Belloli, Landini`
- FORMATO FÍSICO: Miércoles 11 de Noviembre de 2015, en la clase práctica.

APÉNDICE B
CÓDIGO FUENTE RELEVANTE