Métodos Numéricos
TP3

4 de noviembre de 2015

Integrante	LU	Correo electrónico
Martin Baigorria	575/14	martinbaigorria@gmail.com
Federico Beuter	827/13	federicobeuter@gmail.com
Mauro Cherubini	835/13	cheru.mf@gmail.com
Rodrigo Kapobel	695/12	rok_35@live.com.ar

Reservado para la cátedra

Instancia	Docente	Nota
Primera entrega		
Segunda entrega		

Resumen: El siguiente trabajo práctico tiene como objetivo implementar, utilizar y evaluar diferentes métodos de interpolación (entre ellos, lineal, cuadratico y splines) para ser aplicados a la generación de frames para videos en slow motion (o cámara lenta). Se abordarán los detalles relacionados a cada método, como por ejemplo, como afectan en la generación de artifacts y que caracteristicas poseen dependiendo del escenario analizado. Además, en base a la calidad del resultado y tiempo de ejecución,,,(algo más?) compararemos cada método entre si, para luego concluir que bajo ciertas condiciones generales, splines es el mejor para resolver este tipo de problematica. Veremos además, que si el video tiene intercambios de cámara, no será suficiente con splines y se propondrá una solución a este inconveniente para no afectar el resultado empírico, que será, en términos cualitativos, el más suave en la transición de cuadros y que podremos cuantificar, intentando regenerar el video original.

Keywords: TODO

Índice

1. Introducción

En la vida real, nos encontramos situaciones en las que disponemos de información discretizada sobre un comportamiento dado, para la cual, en general, se desconoce como fue generada, es decir que desconocemos la función con la cual se obtienen estos datos, y lo que haremos será aproximar esa función a partir de los mismos para intentar obtener los valores intermedios.

Esto es lo que se conoce como interpolación. Existen diferentes métodos para lograrlo, algunos más o menos eficientes, y cual utilizaremos dependerá de la presición que se quiera alcanzar y el problema que estemos abordando. En general, se buscará que el error cometido al interpolar sea menor que la ganancia en presición.

Uno de los principales usos de la interpolación a lo largo de la historia ha sido el de hallar valores intermedios a los calculados en tablas trigonométricas o astronómicas. También puede encontrarse en matemática financiera para toma de desiciones empresariales **(Martin, acá podrias vos hablar un poco de lo que sabes al respecto, resumido para no aburrir con detalles innecesarios, economia es aburrido :p)**

Otro tipo de problemas en donde interpolación tiene mucha aplicación es en el area de procesamiento de imágenes. Actualmente los televisores LCD o los últimos LED disponen de una mejor definición que la generación anterior. Esto abarca varios aspectos en la imagen obtenida. Citaremos las más importantes:

- Mayor resolución: Es decir mayor cantidad de pixeles en alto por ancho de la pantalla, logrando así un mayor detalle.
- Colores más nitidos.
- Mayor frecuencia de muestreo: o frame rate, que es la tasa de refresco de las imágenes o cuadros en pantalla. Se mide en herzios (hz) y funciona como cota superior para la cantidad de cuadros por segundo o fps (frames per second).

En cuanto a lo relacionado puramente con la resolución de pantalla podemos encontrar el caso particular de los video juegos. Lo que sucede es que los antiguos televisores y sistemas de entretenimiento, vertían el vídeo a una resolución determinada. Todos los juegos antiguos se basaban en ella y se veían "definidos". Al llegar FullHD o el HDReady, las consolas deben interpolar la imagen anterior y mucho más pequeña hasta otra más grande y acorde con la nueva área de visión. La técnica que utilizan se denomina resampling y se basa en copiar el pixel más cercano (nearest-neighbor interpolation). La misma puede variar en su implementación, pudiendo tomar solo un vecino o promediando todos. La elección de uno u otro determinará la graduación de los pixeles generados que tendrá la imágen final, logrando mayor suavidez con el método de los promedios y un acabado algo más iregular en caso contrario.

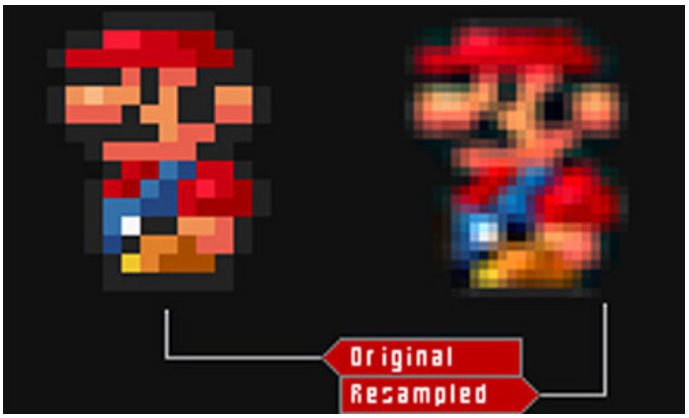

Figura 1: Aplicación de resampling a Mario Bros utilizando el promedio de los vecinos más cercanos.

Debido al alto frame rate del que dispone un televisor LED, es capaz de reproducir peliculas a más de 60 fps, logrando así mayor fluidez en la transición de imágenes. El inconveniente es que no todas las peliculas y series son grabadas a esta frecuencia, si no, a 24 fps que es el estándard historicamente para cine y television. Para poder solucionar este inconveniente los fabricantes de televisores incorporan algoritmos de interpolación que lo que realizan es doblar la cantidad de cuadros por segundo, generando cuadros intermedios, para obtener 60 fps.

Como puede verse en la Figura 2. Se dispone de dos cuadros de un video de un elefante en movimiento. El video fué grabado con pocos fps, por lo cual hay información inexistente. Como resultado, el movimiento del elefante pareciera ser menos fluido entre cuadro y cuadro. Para lograr una transición más suave, se genera un cuadro intermedio mediante interpolación entre el cuadro 1 y 2 obteniendo el cuadro 1a. Si agregaramos más cuadros, obtendriamos una transición aún más fluida, en principio, aunque en la práctica habrá factores que influirán en el resultado.

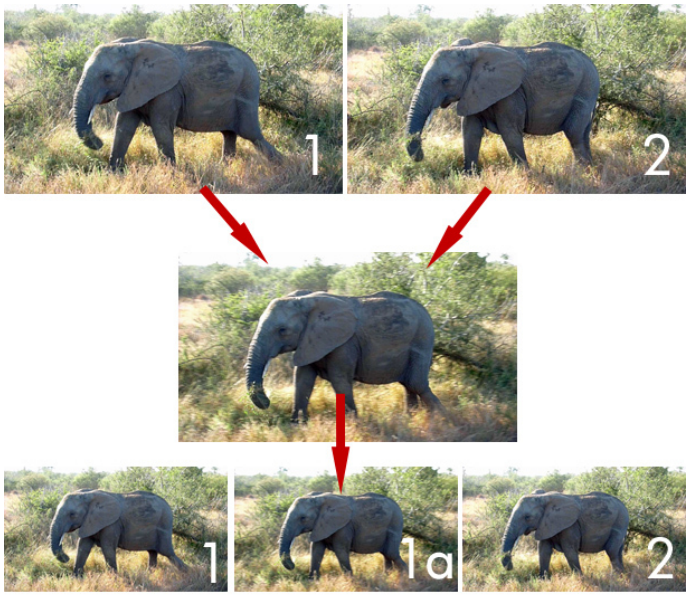

Figura 2: Elefante interpolado.

En particular, si agregamos varios cuadros intermedios y no modificamos los fps lograremos un efecto de slow motion o cámara lenta. Este mismo efecto es el que modelaremos y analizaremos en detalle en este trabajo práctico.

1.1. Motivación y objetivos

Una empresa de páginas web llamada youborn.com busca tener videos en cámara lenta. 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, busca minimizar esta dependencia y solo enviar el video original y que el trabajo de conversion se realize de manera offline, de esta manera, optimizaremos los tiempos de transferencia.

Para lograrlo, como se mencionó anteriormente, recurriremos a interpolación. En particular, nos enfocaremos en el estudio de la interpolación polinómica. La misma, como se verá en el desarrollo, se basa en aproximar una función, por un polinomio. Analizaremos tres métodos, los cuales serán: lineal, cuadrático y splines y compararemos cada uno de ellos entre si.

Luego abordaremos la generación de imágenes para obtener videos en slow motion. Plantearemos el procedimiento para llevarlo a cabo y observaremos como se comporta cada uno de los tres métodos. Veremos los comportamientos en términos de la calidad del resultado y la complejidad temporal para llevarlos a cabo, analizando el trade-off entre eficiencia, suavidad y nitidez. Concluiremos que bajo ciertas condiciones en los datos de entrada, es decir como sea el video que estamos procesando, utilizar splines será la mejor opción. Además estudiaremos que sucede con la generación de artifacts, así denominadas, a las distorciones procedentes de aplicar estos algoritmos.

2. Experimentación

3. Conclusiones

4. Apéndice A: Enunciado

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.

5. Apéndice B: Código

5.1. main.cpp