

Diseño de Sistemas Multimedia - Práctica 2

Parte 1 de la práctica

En esta parte de la práctica hemos creado un programa que ecualiza una imagen oscurecida. Para ello expandiremos los tonos de gris para aumentar el contraste mediante código ensamblador.

El resultado es el siguiente:

```
Microsoft Windows [Versión 10.0.14393]
(c) 2016 Microsoft Corporation. Todos los derechos reservados.

C:\Users\Jose>D:

D:\>cd Documents\GitHub\P2DISM\ConsoleApplication1\Release

D:\Documents\GitHub\P2DISM\ConsoleApplication1\Release>ConsoleApplication1.exe cara2.bmp
Procesando imagen: cara2.bmp ... FIN. CORRECTO. TIEMPO = 1.232351
FIN MMX. CORRECTO. TIEMPO = 0.662085

D:\Documents\GitHub\P2DISM\ConsoleApplication1\Release>
```

Como podemos comprobar el **tiempo de ejecución es la mitad**, para el mismo resultado.



Parte 2 de la práctica

En esta parte de la práctica tenemos que leer dos imágenes para que se realice un fundido entre las dos imágenes, por lo que deberemos sumarlas y modificarlas por el valor del fade. También lo haremos en alto nivel (C++) y en bajo nivel (ensamblador)

```
Selecciónar Símbolo del sistema
17/11/2016 10:05 13.312 ConsoleApplication2.exe
17/11/2016 10:05 38.569 ConsoleApplication2.iobj
17/11/2016 10:05 12.496 ConsoleApplication2.ipdb
17/11/2016 10:05 471.040 ConsoleApplication2.pdb
13/11/2012 06:27 263.222 lena.bmp
17/11/2016 10:07 263.222 salidammx_barbara.bmp
17/11/2016 10:07 263.222 salida_barbara.bmp
      8 archivos      1.588.305 bytes
      2 dirs 494.281.764.864 bytes libres

D:\Documents\GitHub\P2DISM\ConsoleApplication2\Release>ConsoleApplication2.exe lena.bmp ba
rbara.bmp 225
Procesando imagenes: lena.bmp y barbara.bmp... FIN. CORRECTO. TIEMPO = 0.000376
FIN MMX. CORRECTO. TIEMPO = 0.000994

D:\Documents\GitHub\P2DISM\ConsoleApplication2\Release>
```



Problemas:

Visual Studio en un momento no compilaba debido a que decía que funciones como por ejemplo, Fopen estaban obsoletas, pero simplemente cambiando una propiedad del proyecto dejaba de dar estos problemas

Conclusión

En bajo nivel la velocidad aumenta notablemente de tal manera que podría ser lo más indicado para aplicaciones que pudieran requerir una alta velocidad a costa de aumentar la

complejidad del código. Como por ejemplo, los videojuegos necesitan una muy rápida velocidad de procesamiento para mostrar entornos en 3 dimensiones, y por tanto, suelen requerir una alta velocidad. Otro ejemplo sería la descryptación y la seguridad.