

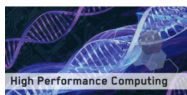


PYTHON



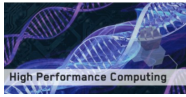
Manual de usuario Filtro de realce de color R, G ó B

VERSIÓN 1.0



Índice

Introducción	2
Configurar entorno	3
Armado del ambiente	4
Ejecución CPU	5
Ejecución GPU	7
Métricas y conclusiones	9



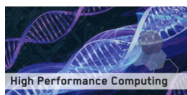
Introducción

El cuaderno de Colab de Realce de color R, G o B, permite realizar pruebas de dicho filtro, el cual consiste en realzar el color seleccionado (rojo, verde o azul) y al resto de los colores pasarlos a escala de grises. Asimismo se puede realizar la comparativa entre una ejecución con CPU y otra con GPU.

Para poder utilizarlo es necesario ejecutar el cuaderno en Colab. La dirección de github del mismo es:

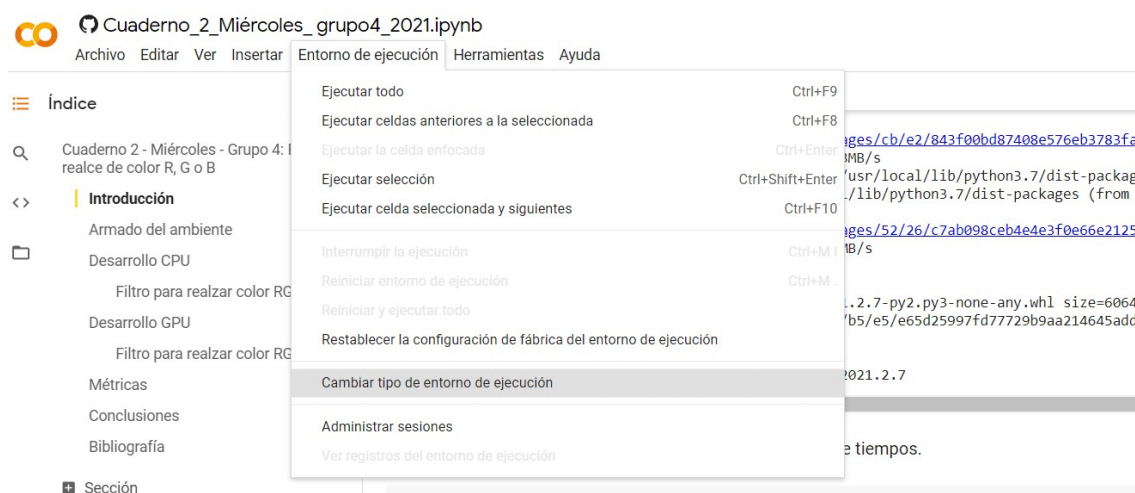
https://github.com/MartinRomano-S/soa-tp2-tp3-grupo4/blob/master/HPC/Cuaderno_2_Mi%C3%A9rcoles_grupo4_2021.ipynb/



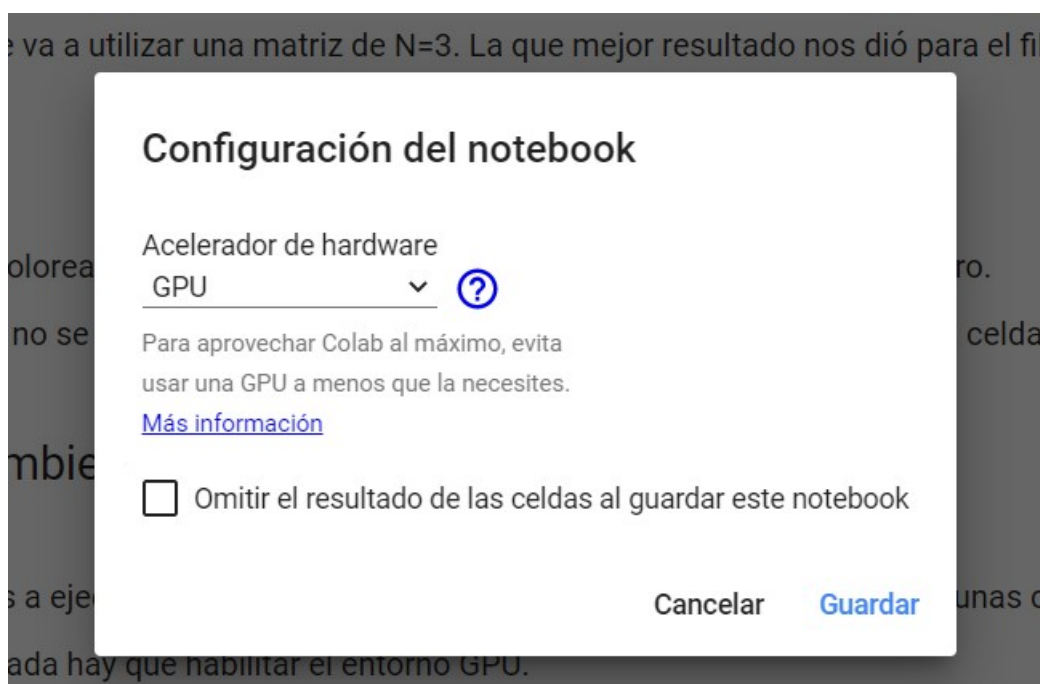


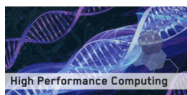
Configurar entorno

Al acceder, lo primero que se debe hacer es configurar el entorno, cambiando el entorno de ejecución a Solo GPU. Primero, clicar “Entorno de ejecución”.



A continuación, se debe cambiar a “Solo GPU” y guardar la configuración.





Armado del ambiente

A continuación, se procede a armar el ambiente. Para ello se debe ejecutar las siguientes líneas:

```
Cuaderno_2_Miércoles_grupo4_2021.ipynb
Archivo Editar Ver Insertar Entorno de ejecución Herramientas Ayuda

ce
+ Código + Texto Copiar en Drive Conectar

Introducción
Armado del ambiente
Desarrollo CPU
Filtro para realizar color RGB
Desarrollo GPU
Filtro para realizar color RGB
Métricas
Conclusiones
Bibliografía
Sección

Armado del ambiente

Debido a que vamos a ejecutar código tanto en CPU como en GPU, necesitamos tener unas consideraciones previas.

1. Primero que nada hay que habilitar el entorno GPU.
Para esto hay que ir a Entorno de ejecución > Cambiar tipo de entorno de ejecución.
Seleccionar GPU como acelerador de hardware y luego guardar.

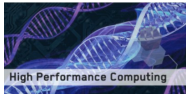
2. Luego debemos instalar OpenCL [2] ya que vamos a trabajar con ésta biblioteca.
Para ello debemos ejecutar la siguiente instrucción:

[ ] !pip install pyopencl

Collecting pyopencl
  Downloading https://files.pythonhosted.org/packages/cb/e2/843f00bd87408e576eb3783fa281836be50e2301a3cb9f86fca347ef3228/pyo
    880kB 28.3MB/s
```

Definimos una función lambda para mostrar las métricas de tiempos.

```
[ ] # Definición de función que transforma el tiempo en milisegundos
    tiempo_en_ms = lambda dt:(dt.days * 24 * 60 * 60 + dt.seconds) * 1000 + dt.microseconds / 1000.0
```



Ejecución CPU

En la opción Desarrollo CPU, al clicar en el botón de “play”, se ejecutará el filtro de realce R, G o B bajo CPU pudiéndose especificar una imagen y seleccionar el color a realzar.

+ Código + Texto Copiar en Drive Conectar Editando

▼ Desarrollo CPU

Imágenes propuestas:

Bombonera
https://as01.epimg.net/argentina/imagenes/2021/05/25/futbol/1621973866_916305_1621974052_noticia_normal_recorte1.jpg

Estrellas (Pequeña):
<https://i.ibb.co/rygm5Nt/stars.png>

Salzburgo (Grande):
https://press-music.com/wp-content/uploads/2019/12/Festung_Salzach.jpg

[] #@title Filtro para realzar color RGB

#@markdown Ingrese la URL de la imagen que desea procesar:

url_de_la_imagen = "https://as01.epimg.net/argentina/imagenes/2021/05/25/futbol/1621973866_916305_1621974052_noticia_normal_recorte1.jpg"

if not (type(url_de_la_imagen) is str) or (url_de_la_imagen == ""):

raise TypeError("URL inválida")

!wget {url_de_la_imagen} -O imagen_usuario

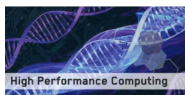
Filtro para realzar color RGB

Ingrese la URL de la imagen que desea procesar:

url_de_la_imagen: "https://as01.epimg.net/argentina/imagenes/2021/05/25/futbol/1621973866_916305_1621974052_noticia_normal_recorte1.jpg"

Seleccionar el color que desea realzar:

color: Azul



Al finalizar la ejecución veremos el resultado final en modalidad CPU.

+ Código + Texto Copiar en Drive

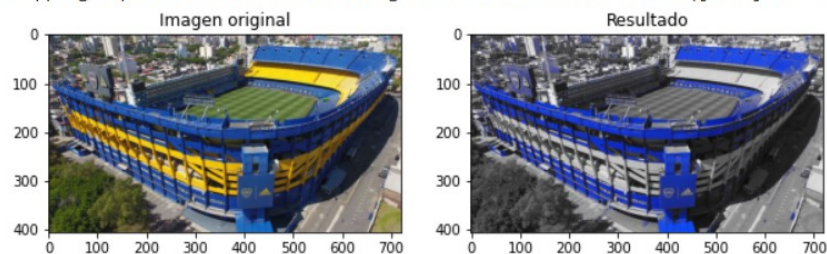
```
[ ] plt.title("Resultado")
plt.imshow(img_R_cpu)
tiempo_cpu_total = datetime.now() - tiempo_cpu_total
```

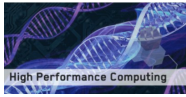
```
--2021-06-27 16:47:33-- https://as01.epimg.net/argentina/imagenes/2021/05/25/futbol/1621973866_916
Resolving as01.epimg.net (as01.epimg.net)... 199.232.194.133, 199.232.198.133
Connecting to as01.epimg.net (as01.epimg.net)|199.232.194.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 91447 (89K) [image/jpeg]
Saving to: 'imagen_usuario'
```

```
imagen_usuario      100%[=====>]  89.30K  --.-KB/s   in 0.002s
```

```
2021-06-27 16:47:33 (51.1 MB/s) - 'imagen_usuario' saved [91447/91447]
```

Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for





Ejecución GPU

En la opción Desarrollo GPU, se procederá a ejecutar el filtro R, G o B bajo GPU.

Primero se ejecutará un código para poder generar el kernel.

▼ Desarrollo GPU

Para el desarrollo en GPU, primero ejecutamos el código que genera el Kernel necesario para la aplicación de los filtros.

```
codigo = """
const sampler_t sampler = CLK_NORMALIZED_COORDS_FALSE |
                          CLK_ADDRESS_CLAMP_TO_EDGE |
                          CLK_FILTER_NEAREST;

//Lo que hacemos con estos filtros es detectar si la componente R, G o B del pixel es superior a 100 y las
//Si pasa el filtro, realizamos la componente un 25% y reducimos el resto de las componentes del pixel un 25%
//Si no pasa el filtro, convertimos el pixel a escala de grises.

__kernel void realzar_color_rojo(__read_only image2d_t original, __write_only image2d_t resultado, __global const int *img ancho)
{
    int ancho_imagen = *img ancho;
```

A continuación, se ejecutará el código en el cual se puede especificar una imagen y seleccionar el color a realzar.

+ Código + Texto Copiar en Drive Conectar Editando

Luego ejecutamos el código

Imágenes propuestas:

Bombonera
https://as01.epimg.net/argentina/imagenes/2021/05/25/futbol/1621973866_916305_1621974052_noticia_normal_recorte1.jpg

Estrellas (Pequeña):
<https://i.ibb.co/rygm5Nt/stars.png>

Salzburgo (Grande):
https://press-music.com/wp-content/uploads/2019/12/Festung_Salzach.jpg

#@title Filtro para realzar color RGB

#@markdown Ingrese la URL de la imagen que desea procesar:

url_de_la_imagen = "https://as01.epimg.net/argentina/imagenes/2021/05/25/futbol/1621973866_916305_1621974052_noticia_normal_recorte1.jpg"

if not (type(url_de_la_imagen) is str) or (url_de_la_imagen == ""):

raise TypeError("URL inválida")

!wget {url_de_la_imagen} -O imagen_usuario

#@markdown Seleccionar el color que desea realzar:

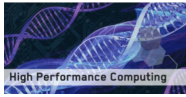
Filtro para realzar color RGB

Ingrese la URL de la imagen que desea procesar:

url_de_la_imagen: "https://as01.epimg.net/argentina/imagenes/2021/05/25/futbol/1621973866_916305_1621974052_noticia_normal_recorte1.jpg"

Seleccionar el color que desea realzar:

color: Azul

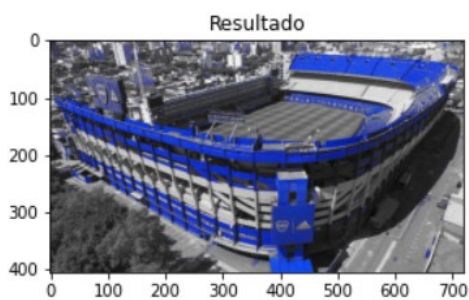


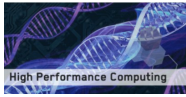
Al terminar, se verá el resultado de la imagen.

```
--2021-06-27 16:47:47-- https://as01.epimg.net/argentina/imagenes/2021/05/25/futbol/  
Resolving as01.epimg.net (as01.epimg.net)... 199.232.194.133, 199.232.198.133  
Connecting to as01.epimg.net (as01.epimg.net)|199.232.194.133|:443... connected.  
HTTP request sent, awaiting response... 200 OK  
Length: 91447 (89K) [image/jpeg]  
Saving to: 'imagen_usuario'
```

```
imagen_usuario      100%[=====>]  89.30K  --.-KB/s    in 0.003s
```

```
2021-06-27 16:47:47 (28.2 MB/s) - 'imagen_usuario' saved [91447/91447]
```





Métricas y conclusiones

Finalmente, se pueden ver las métricas a presionar el botón de “play” en dicha sección para poder comparar los tiempos tanto de CPU como GPU y además más abajo se podrán apreciar las métricas.

▼ Métricas

```
[ ] print ('')
print ("|{:<15}|{:<20}|{:<20}|".format('Tiempos', 'Implementación CPU', 'Implementación GPU'))
print ('|-----|-----|-----|')
print ("|{:<15}|{:>20}|{:>20}|".format('Tiempo CPU', str(tiempo_en_ms(tiempo_cpu)) + 'ms', tiempo_gpu_gpu_total_string + 'ms'))
print ("|{:<15}|{:>20}|{:>20}|".format('Tiempo GPU', '0.0ms', str(tiempo_en_ms(tiempo_gpu)) + 'ms'))
print ("|{:<15}|{:>20}|{:>20}|".format('Tiempo Total', str(tiempo_en_ms(tiempo_cpu_total)) + 'ms', tiempo_gpu_total_string +
print ('|-----|-----|-----|')
```

Tiempos	Implementación CPU	Implementación GPU
Tiempo CPU	3617.422ms	222.31ms
Tiempo GPU	0.0ms	35.969ms
Tiempo Total	3674.834ms	258.279ms

▼ Conclusiones

A partir de los resultados obtenidos en las métricas, se ve que el tiempo de ejecución total del programa utilizando la GPU para la imagen propuesta es de tan sólo el 7.02% del tiempo de ejecución total utilizando sólo la CPU. Es decir, aproximadamente 14 veces menos. Ésto nos da una clara demostración de la mejora en tiempo que nos proveen los threads.