**[ce\_ff\_07.py]:**

**======================================**

Comunicaciones Especulativas Flow Field v 0.7

Es un sistema para análisis de las secuencias de imágenes generadas por raspyLapse.

La herramienta usa la implementación del algoritmo de Lucas-Kanade[[1]](#footnote-1) de OpenCV[[2]](#footnote-2) para calcular un campo de flujo óptico a partir de la diferencia de dos imágenes sucesivas en el tiempo. Este campo de flujo es una representación, en un espacio de alta dimensionalidad del cambio ocurrido entre imágenes, que se acumulan para conformar un batch de datos de entrenamiento. Luego, se lleva a cabo un agrupamiento o clustering sobre estos datos usando la técnica KMeans[[3]](#footnote-3), que busca los patrones de regularidad entre estos vectores y los agrupa según la semejanza o diferencias de su contenido. La herramienta puede producir un stream de imágenes como salida y almacena en carpetas los modelos y datos que obtiene en cada ejecución. Finalmente, también permite cargar un modelo de agrupamiento antes entrenado y generar un stream de datos OSC a partir del análisis y clasificación en tiempo real.

|  |  |
| --- | --- |
| C:\Users\E\AppData\Local\Microsoft\Windows\INetCache\Content.Word\img_00084.png  **Fig:1.a** Frame 84 del timelapse TL09 | C:\Users\E\AppData\Local\Microsoft\Windows\INetCache\Content.Word\img_out_00084.png  **Fig:1.b** Con vectores de movimiento sobrepuestos |

La herramienta acepta distintos parámetros de configuración para habilitar y deshabilitar etapas o funciones del proceso. La lista completa de argumentos que acepta en la actual versión 0.7 es:

**Input folder path** :: "*-i*", "*--in-dir*", default="*./imgs\_in*" :: ruta del directorio donde se encuentran las imágenes de entrada, cuando no se usa el parámetro “*-v*”

**Output folder path** :: "*-o*", "*--out-dir*", default="*./imgs\_out*" :: ruta del directorio para escribir imágenes de salida, cuando se usa el parámetro “-w”

**Video Input path** :: "*-v*", "*--video*" :: ruta del archivo de video de entrada, si se especifica el valor “0” el programa usará la webcam como stream de video

**Enable show** :: "*-s*", "*--show*", default="*False*" :: habilita el despliegue de información en video, desabilitada por defecto para funcionar por conexión remota

**Enable write output files** :: "*-w*", "*--write*", default="*True*" :: si está habilitada, guarda cada frame con las flechas del campo de flujo o direcciones discretas en el folder de “*-o*”

**Model file path** :: "*-m*", "*--model*", default="*./models/kmeans\_1500555372.cpk*" :: carga el archivo indicado como modelo para asignar una etiqueta de grupo a cada nuevo frame

**Canny detector enable** :: "*-n*", "*--canny*", default="*False*" :: Si está habilitado, usa el algoritmo de detección de bordes para mejorar el contraste y detección de movimiento

|  |  |
| --- | --- |
|  |  |
|  |  |

**Fig 2:** Frames 18[UL], 19[UR], 20[DL], 21[DR] con Canny edge detector habilitado

**OSC stream enable** :: "*-c*", "*--osc*", default="*False*" :: habilita la salida de mensajes OSC

**Receiver IP address** :: "*-r*", "*--receiver-ip*", default="*192.168.1.255*" :: dirección IP del receptor

**Receiver OSC port** :: "*-p*", "*--receiver-port*", default="*57120*" :: puerto IP del receptour

**Discrete direction flowfield enable** :: "*-d*", "*--direct*", default="*False*" :: Cuando habilitado, discretiza la dirección del movimiento hacia los 5 rumbos cardinales: NESWZ; en otro caso proporciona el valor del ángulo de movimiento medido desde +X en el rango de [-PI, PI].

|  |  |
| --- | --- |
|  |  |
|  |  |

**Fig 3**: Diferencias en la detección de movimiento. Las imágenes superiores son frames sucesivos donde se han marcado únicamente los puntos con actividad luego de aplicar la detección de bordes. Las imágenes de abajo son la salida producida por las mismas imágenes, pero sin emplear la detección de bordes. Para estas capturas se habilitó también la discretización de direcciones, por lo que en cada casilla se indica con una pequeña letra la dirección del movimiento.

El script con la herramienta se ejecuta desde la términal y requiere una instalación con OpenCV (probado en las versiones 2.4 y 3.2), Python 2.7, y los paquetes numpy, scikit-learn >0.14, pyOSC, y cv2.

Ejemplos de uso:

1. Analiza el archivo TL6.mp4 en la carpeta local, no escribas archivos de salida pero sí muestra el proceso. Habilita el envío de OSC a la dirección por default:

$ python ce\_ff\_07.py -v "./TL6.mp4" -w "False" -s "True" -c "True"

2. Analiza el archivo TL9.mp4 en la subcarpeta /videos, escribe cada frame a la carpeta ./TL9\_out, muestra el proceso y envía OSC a la dirección por default:

$ python ce\_ff\_4.py -v "./videos/TL9.mp4" -w "True" -s "True" -c "False" -o "./TL9\_out"

3. Analiza el archivo TL3.mp4, no escribas archivos de salida, pero sí muestra el proceso, aplica el detector de bordes, también discretiza las direcciones y envía mensajes OSC a todas las instancias de supercollider conectadas a la red local (192.168.1.255::57120)

$ python ce\_ff\_07.py -v "./TL3.mp4" -w "False" -s "False" -n "True" -c "True" -d "True" -r "192.168.1.25" -p "57120"

1. <http://www.inf.fu-berlin.de/inst/ag-ki/rojas_home/documents/tutorials/Lucas-Kanade2.pdf> [↑](#footnote-ref-1)
2. <http://docs.opencv.org/3.2.0/d7/d8b/tutorial_py_lucas_kanade.html> [↑](#footnote-ref-2)
3. <http://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html> [↑](#footnote-ref-3)