

**LABORATORIO**

**EQUIPOS Y SISTEMAS DE**

**AUDIO Y VIDEO**

**(ESAV)**

*Práctica Final*

***Analizador de video Vectorscopio para interfaces SDI***

Curso 2018-2019

Grupo 2:

Alba Blanco, Miguel Ángel

Durán Castro, Patricia

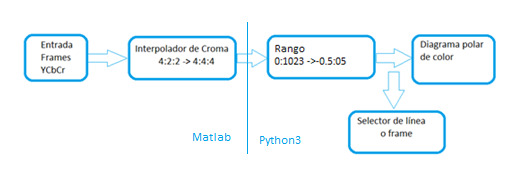
Rodrigo Castillo, Adrián

## Introducción.

**En este informe se presenta la arquitectura general y realización de un Vectorscopio para archivos SDI.**

**El sistema es capaz de realizar las transformaciones necesarias para representar en un diagrama con la información de color extraída del vector crominancia de un video digital.**

A continuación, se muestra un diagrama de bloques con los módulos necesarios los cuales se explicarán posteriormente.

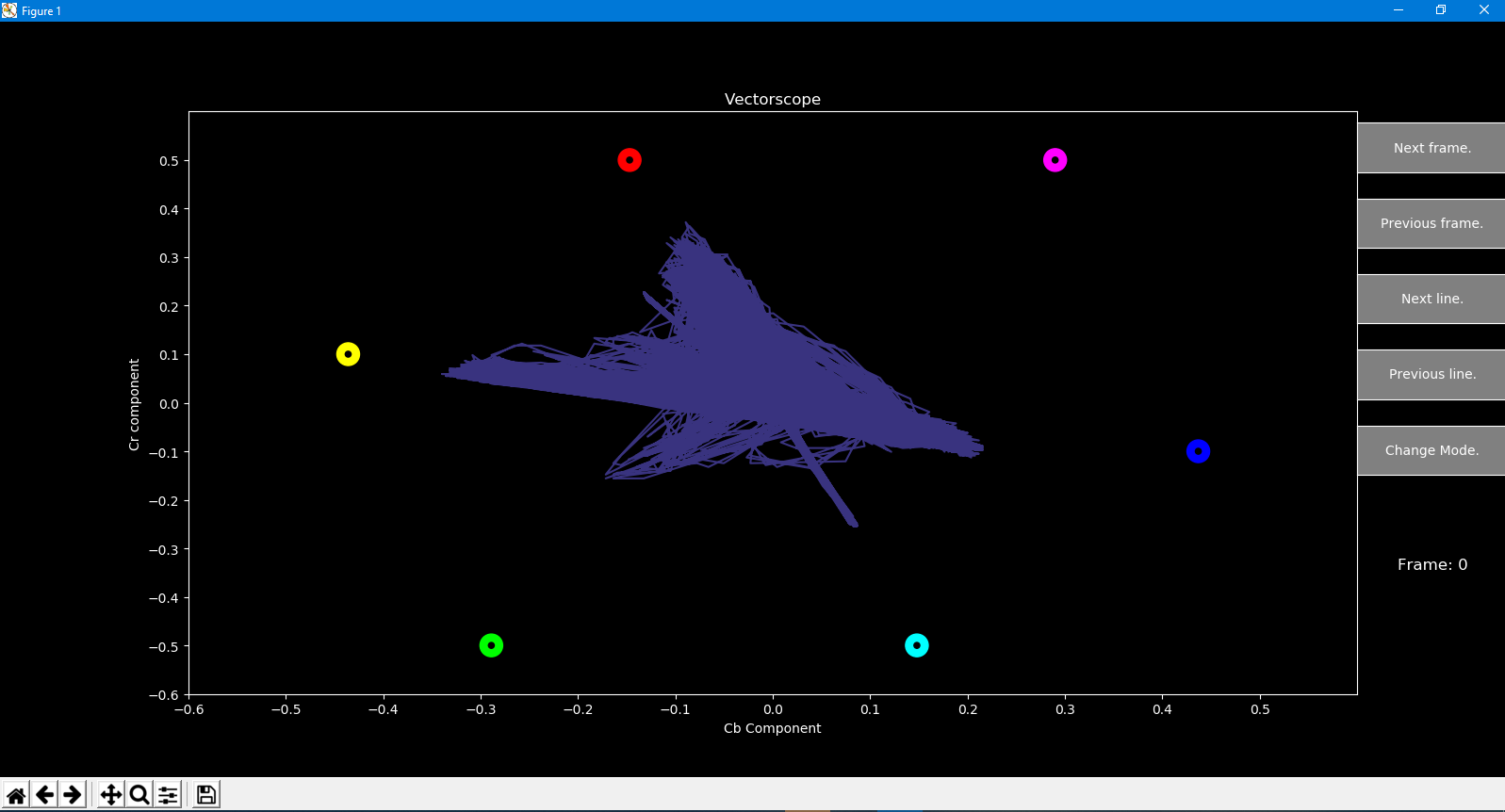


*Ilustración 1: Diagrama de bloques.*

* **Entrada de frames.** Este módulo es el encargado de recibir la información de vídeo de una trama SDI y extraer las componentes de luma y croma.
* **Interpolador de Croma.** Este módulo es el encargado de realizar una interpolación espacial de la croma en caso de que sea necesario.
* **Rango 0:1023🡪-0.5:0.5.** Este módulo aplican las transformaciones necesarias para poder representar los valores en el Vectorscopio.
* **Diagrama de color.** Representa gráficamente la información de color en un diagrama polar.
* **Selector de línea o frame.** Ofrece la posibilidad de poder representar en el vectorscopio un frame completo o una línea determinada.

|  |  |
| --- | --- |
| *Ilustración 2: Imagen diagrama de barras laboratorio* | *Ilustración 3: Representación del diagrama de barras en Vectorscopio del laboratorio* |

Todos los módulos presentados se desarrollan en lenguaje de programación Matlab excepto el módulo de visualización que es desarrollado en lenguaje de programación Python3.



*Ilustración 2. Captura de la ventana del Vectorscopio.*

## Matlab

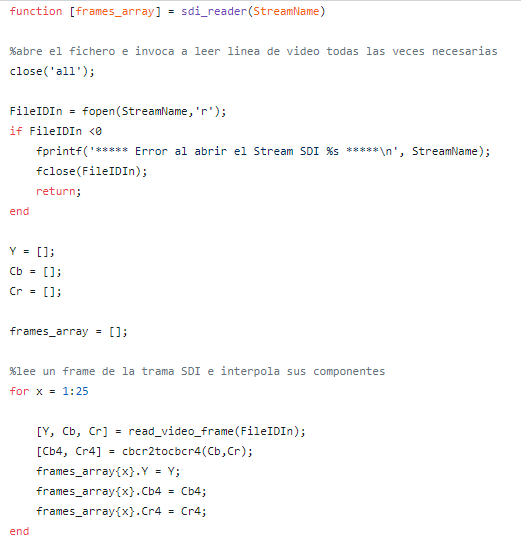
## Entrada de frames

La lectura de una trama SDI se realizará de la siguiente manera:

El bucle principal del programa **sdi\_reader.m** se puede configurar para que extraiga un número determinado de frames del fichero SDI. Por cada iteración, se leerá un frame entero de la trama SDI y se devolverá un objeto que contiene una estructura tipo Cell de Matlab. Esta estructura tiene 3 matrices: Y, Cb, Cr. Estas 3 matrices tienen la misma dimensión (720x576) que se corresponde con una secuencia SDI SD, que es el formato que acepta el programa.

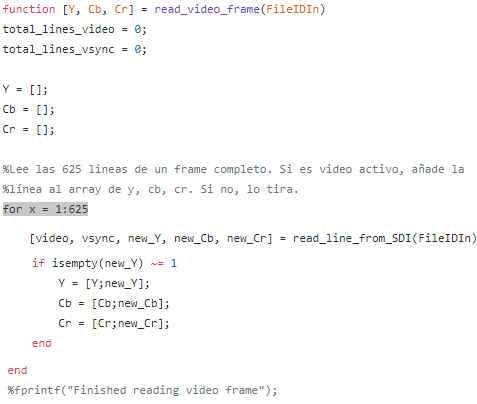
* **Sdi\_reader** devuelve un array con los frames obtenidos.

Este procedimiento principal utilizará la función **read\_video\_frame** para obtener un frame por cada iteración del bucle for e interpola las componentes Cb y Cr, dado que la información original de vídeo es 4:2:2 a través de la función **cbcr2tocbcr4**.



*Ilustración 3. Estructura general de sdi\_reader*

* La función **read\_video\_frame** lee un frame de la trama SDI y devuelve los arrays Y, Cb, Cr, iterando 625 veces, una vez por cada línea de video del formato SD. Esta función utiliza la función **read\_line\_from\_SDI** para obtener una línea de la trama. En función del tipo de línea que se ha encontrado, **read\_video\_frame** la guarda en el array Y, Cb y Cr si es de video activo o la descarta si pertenece al sincronismo vertical.



*Ilustración 4. Estructura general de read\_video\_frame*

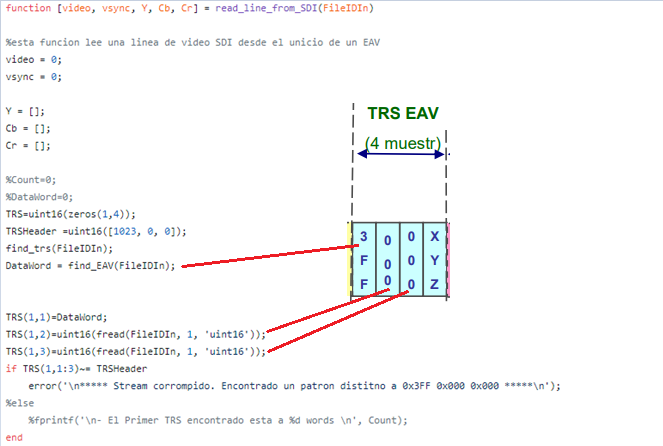
* La función **read\_line\_from\_sdi** leeuna línea de la trama SDI y devuelve 2 flags que indican si la línea es de video activo o no y arrays de tamaño 720, 360 y 360 respectivamente de las componentes Y, Cb, Cr pertenecientes a la línea.

Esta función utiliza la función **find\_trs**, que se encarga de encontrar el primer TRS de la trama.

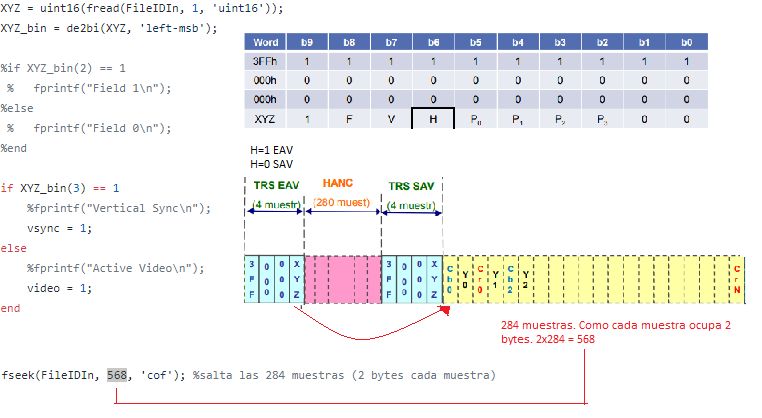
Una vez localizado el TRS se llama a la función **find\_eav,** esta función se encarga de ver si está activo el flag H de la parte XYZ, indicando que se trata de un EAV. De esta manera el puntero lector del fichero de video queda colocado en la posición correcta para poder leer la trama siempre a partir de un EAV. Esto permite leer tramas SDI que no empiecen justo en un EAV.

Cuando **read\_line\_from\_SDI** se está listo para la lectura, descarta la información de HANC y SAV, deja el puntero de fichero listo para leer el payload y activa el flag **video** o **vsync** dependiendo de si la línea es video activo o no. Esto se determina mediante el flag V de la parte XYZ del EAV.

Para leer el payload, se utiliza la función **read\_payload**.



*Ilustración 5. Estructura general de read\_line\_from\_SDI.*



*Ilustración . Estructura general de read\_line\_fom\_SDI.*

*Ilustración 6. Lectura de la cabecera XYZ*

* La función **read\_payload** lee las 1440 de video, en grupos de 4 muestras para poder iterar sobre todas las muestras y devuelve un array de dimensiones 720, 360 y 360 de las muestras Y, Cb, Cr respectivamente. Si el payload es de sincronismo vertical, todo se descarta, avanzando el puntero de fichero.

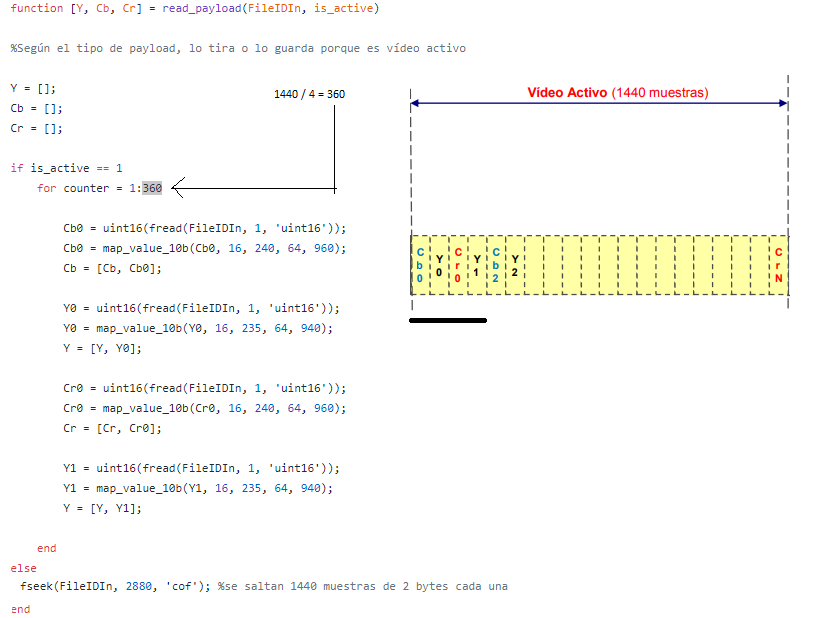


Ilustración 7. Lectura del payload de la trama.

*Ilustración . Estructura general de read\_payload.*

Cabe mencionar que la función **map\_value\_10b()** es una conversión de profundidad de bit de

8 bit por pixel a 10. Esto es debido a que en este caso, las secuencias de test que se utilizaron

tenían una profundidad de pixel de 8 bits, aunque en la realidad toda trama acorde al estándar SD-SDI debería tener una profundidad de 10 bits por muestra.

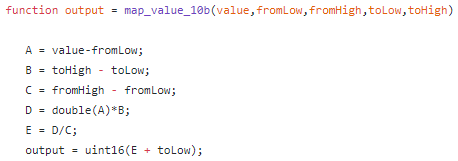


Ilustración 8. Estructura de la función de mapeo.

## Interpolador de croma.

Dado el hecho de que la información de vídeo en una trama SDI está submuestreada en 4:2:2,

es necesario interpolar Cb y Cr para obtener las 3 componentes Y, Cb y Cr con las mismas

dimensiones de 720x576.

Para ello, se utiliza la función **cbcr2tocbcr4.** Esta función utiliza las funciones **interpola2to4**

**y create\_resize\_matrix**, así como la función predefinida **interp1** para realizar la

interpolación de los valores de croma con una aproximación cúbica. De esta manera,

los arrays de Cb y Cr que originalmente tienen dimensiones de 360, pasan a tener dimensión

720.

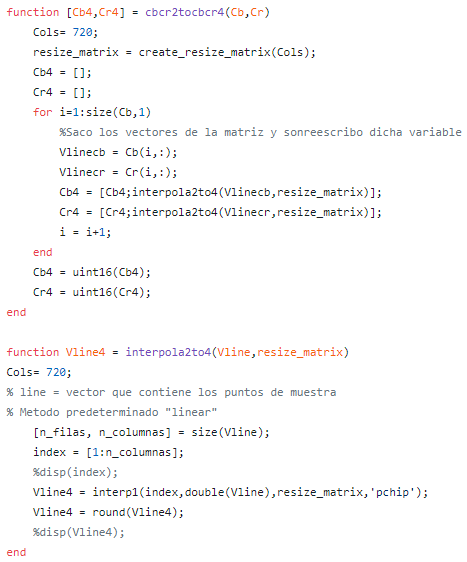


Ilustración 9. Estructura del interpolador de croma.

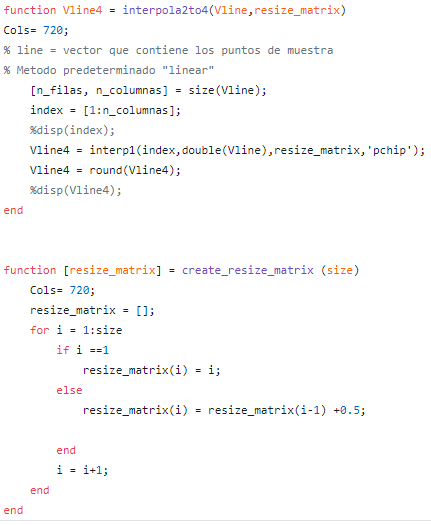


Ilustración 10. Estructura del interpolador de croma.

## Python3

## El programa principal del Vectorscopio es la función main

## 

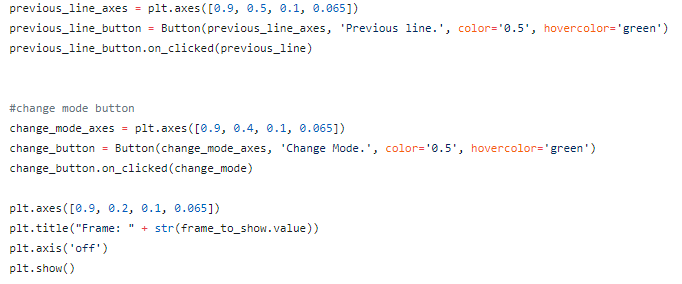


Ilustración 11. Estructura del procedimiento principal.

## Programa principal.

La función **main** de este programa se encarga de llamar al motor de cálculo Matlab y a las funciones descritas anteriormente de este informe para obtener la información de vídeo de los frames, en sus componentes Y, Cb, Cr. Esta información es almacenada en el array **frames\_array**.

Además, la función **main** inicializa la interfaz gráfica donde se mostrará el vectorscopio y la información de video(botones, ventana principal, etc).

## Vectorscopio.

La función **draw\_vectorscope** representa en un plano con dos ejes los valores del vector de

crominancia, donde el eje Y es la componente Cr y el eje X es la componente Cb.

El Vectorscopio base se dibuja mediante la función **plt.scatter**, que dibuja puntos discretos

en en el plano. En este caso, se presentan los puntos que corresponden con los colores

primarios y secundarios R, G, B, M, C, Y.

s= [] representa el área del círculo para dichos colores. El valor de 300 sirve para poder

percibir de manera apropiada el color y 20 el punto exacto en el que se centran.

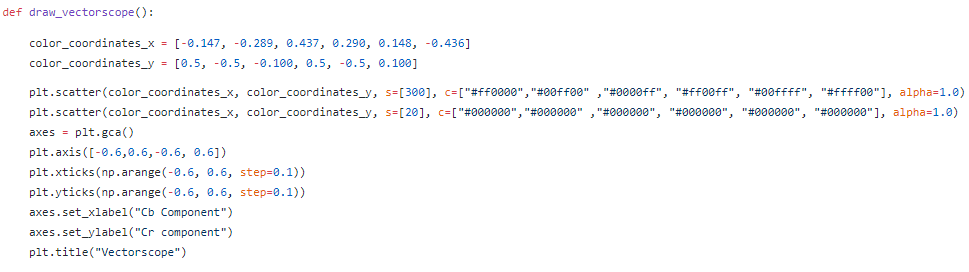
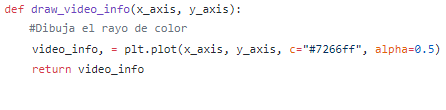
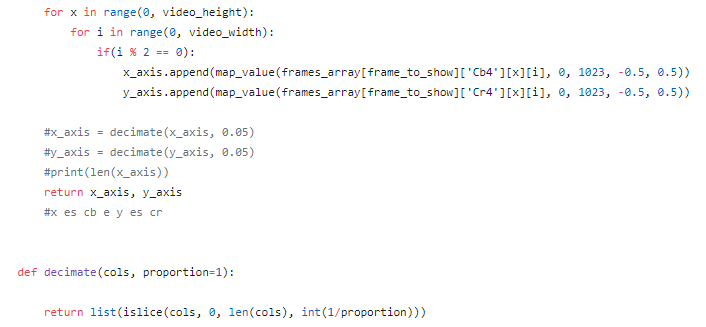


Ilustración 12. Dibujado del vectorscopio.

La función **draw\_video\_info** representa en el plano el trazo que une los valores de las muestras de crominancia. Esto da la información de color que se ha obtenido de la trama SDI. Para que se pueda representar, la información obtenida de **frames\_array** debe introducirse en los arrays **x\_axis** y **y\_axis**. Esto se consigue mediante la función **get\_video\_frame.**





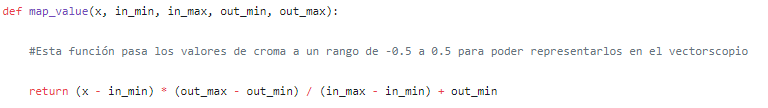


Ilustración 13. Obtención de los valores de crominancia a representar.

Dado que los valores de Cb y Cr son de 10 bit, cuyo rango teórico es de 0 a 1023, deben ser convertidos a un rango de -0.5 a 0.5, para poder representarse correctamente en la gráfica.

Históricamente, el vectorscopio se utilizaba para comprobar señales PAL o NTSC, cuyo espacio de color es YUV. Las componentes U y V tienen un rango nominal de -0.5 a 0.5, pero en el momento de digitalizar estas componentes deben tener valores discretos de 10 bits (0 a 1023). Para representar un vectorscopio, por ende, se debe volver a transformar el rango de valores de 10 bits al rango -0.5:05.

Esto se hace mediante el bucle de la figura (poner ilustración) y la función **map\_value.**

Dado que la librería de gráficos utilizada en este proyecto (Matplotlib) no está optimizada para grandes cantidades de datos, se ha optado por hacer un diezmado de las muestras a la mitad y así mejorar la velocidad de la interfaz de usuario a la hora de cambiar de frame o cambiar de modo línea/frame. Esto se realiza mediante la función **decimate.**

De esta manera se pasa de tener dos arrays de cada uno 414720 elementos a tener 207360.

Este procedimiento no afecta de manera visual a la correcta lectura del vectorscopio.

## Selector de línea o frame.

Para seleccionar entre un frame o una línea se ha declarado la función **change\_mode**. En

caso de esté seleccionado el modo frame se mostrará el frame actual y su número debajo del panel de control y se mostrará el número de línea y frame en caso de que elo modo línea esté seleccionado. Para poder cambiar entre un modo, se debe pulsar el botón change mode.

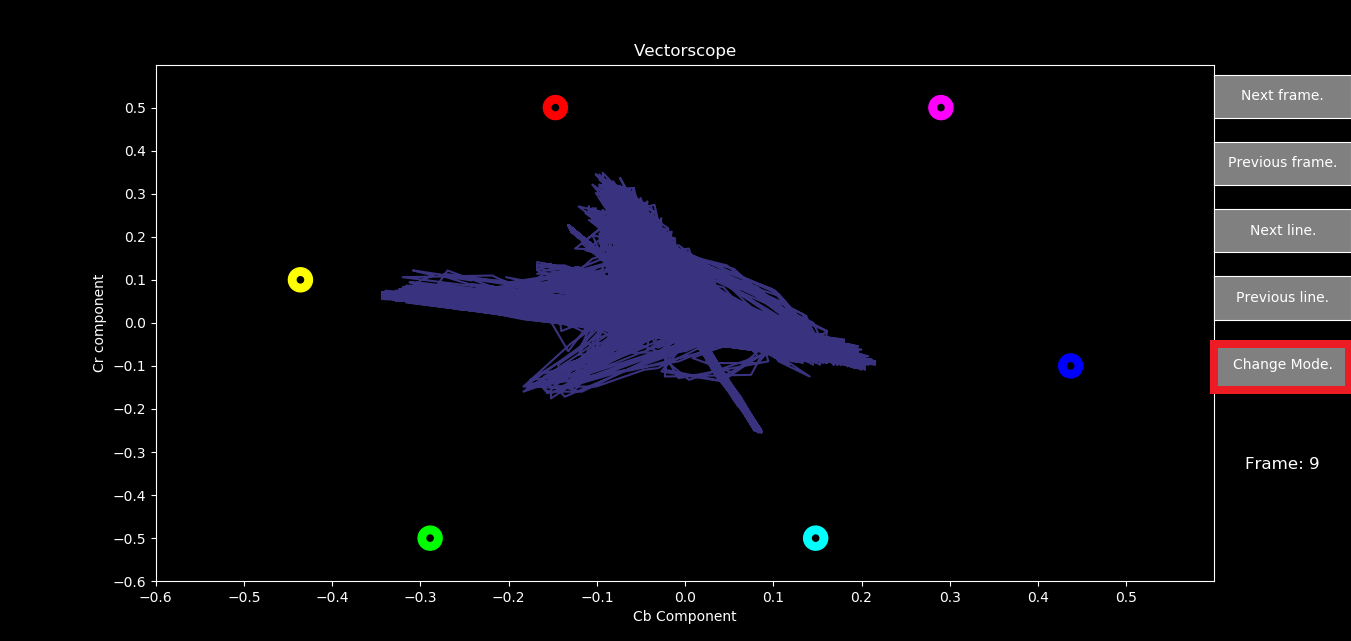


Ilustración 13. Pantalla principal del vectorscopio y función de cambio de modo.

**Funcionamiento del Vectorscopio**

Para que el programa funcione correctamente, se deben instalar las siguientes dependencias de Python:

* Python 3.6.x
* Matplotlib. Para instalar esta librería, se debe ejecutar el siguiente comando:
* Motor de Matlab para python. Para ello, se debe navegar hasta el siguiente directorio:

Y ejecutar el comando:

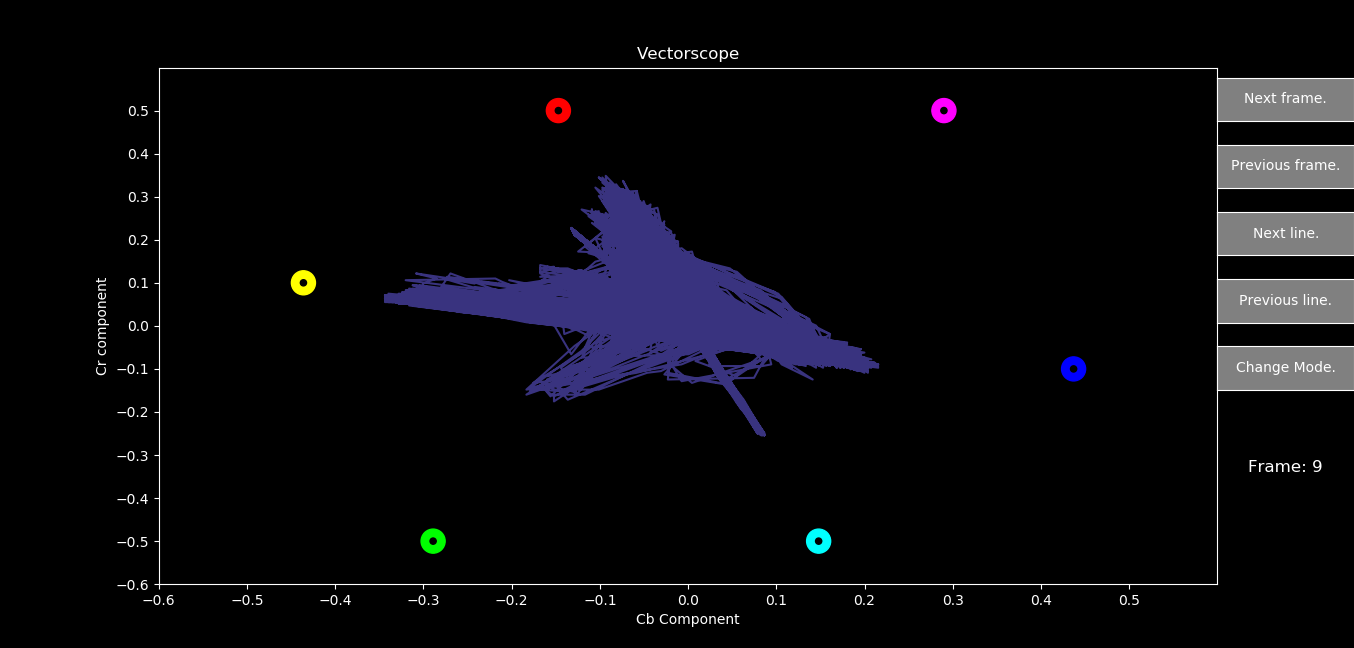
Es necesario asegurarse de que en el sistema operativo utilizado, la instrucción python corresponde con la ejecución de Python 3.6.x. En caso de que estén instaladas varias versiones de python en el sistema operativo, como por ejemplo, python 2 y 3, la instrucción python debe cambiarse por python3.

Para ejecutar el programa, debe utilizarse la línea de comandos de windows y ejecutar la siguiente instrucción dentro de la carpeta con el código fuente del programa:

python Vectorscope.py [nombre de la trama SDI] [número de frames que leer]

El vectorscopio tiene las siguientes funciones implementadas:

* Next frame: Permite avanzar un frame. En el caso de estar en modo línea cambiará automáticamente a modo frame y avanzará.
* Previous frame: Retrocede un frame. Al igual que en el caso anterior se cambiará automáticamente a modo frame en caso de estar en modo línea.
* Next line: Avanzará a la siguiente línea del frame en el que se encuentre.
* Previous line: Retrocede a la línea anterior del frame en el que se encuentre.
* Change Mode: Cambiar del modo frame al modo línea.

****

En la pantalla principal, se muestran las coordenadas de los colores primarios

R, G, B y los secundarios C, Y, M coloreados con los colores correspondientes. En Morado, se dibuja la información de crominancia de la línea o frame seleccionados. El eje Y representa los valores de Cr y el eje X representa los valores Cb.