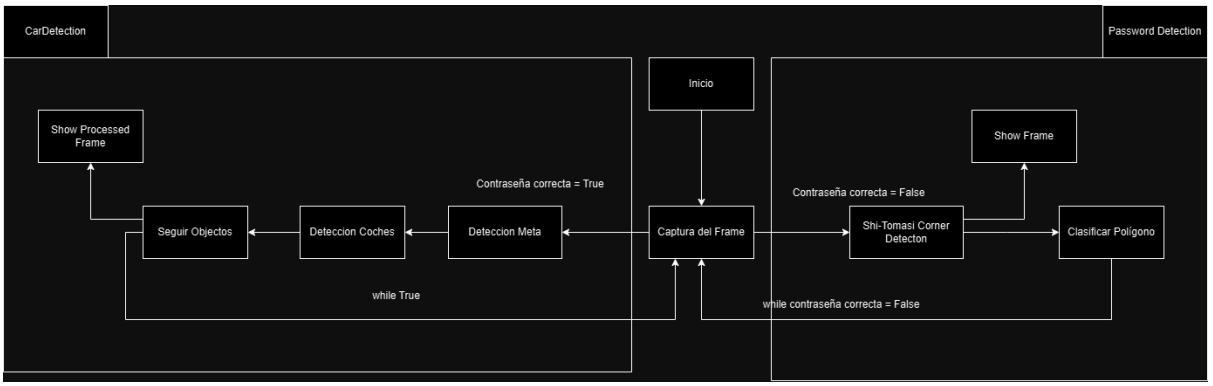
ASIGNATURA: VISIÓN POR ORDENADOR
MEMORIA DEL PROYECTO FINAL

INTRODUCCIÓN

Este proyecto consiste en crear un programa que permita ver si un determinado objeto ha cruzado una cierta barrera. Para este caso específico, se ha implementado un código que permite ver si un coche ha cruzado una línea de meta. Para ello, se han implementado diferentes técnicas y conceptos incorporados durante la asignatura. Entre estas técnicas se encuentran la segmentación por color y la detección de esquinas.
Además, para poder acceder al funcionamiento del programa, primero se debe introducir una contraseña consistente en diferentes formas que se deben ir introduciendo en la imagen de la cámara de manera secuencial.

DIAGRAMA DE BLOQUES



SECUENCIA DE TRANSFORMACIÓN DE LA IMAGEN

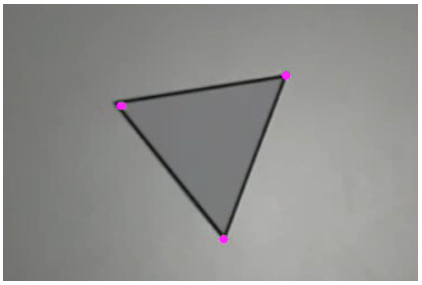
En cuanto a la transformación de la imagen hay que diferenciar dos situaciones. La primera se da cuando queremos introducir la contraseña. Para esta primera situación realizamos el detector de esquinas de Shi-Tomasi, en el que primero cambiamos la imagen a escala de grises y luego se hallan las esquinas. La segunda situación se da para la detección de los coches. Primero creamos una máscara para encontrar los colores negros de la imagen (encontrar la línea de meta) y luego otras máscaras para quedarnos con tan solo los coches del circuito. Tras habernos quedado con los elementos de la escena, los localizamos en cada frame y, dependiendo de su posición, sumamos vueltas o no.

SECUENCIA DE SEGURIDAD

Para este apartado, se ha decidido diseñar una contraseña consistente en tres formas. Primero se debe introducir un triángulo, luego un cuadrado y, finalmente, un

pentágono. Si se introduce una forma que no es acertada el usuario debe introducir la contraseña de nuevo desde el principio.
Para crear este programa se ha utilizado el detector de esquinas de Shi-Tomasi, el cual permite hallar las esquinas de una imagen. Dado que este sistema funciona en tiempo real, por cada frame que aparece en pantalla es necesario calcular sus correspondientes esquinas.
Una vez calculadas sus esquinas, se muestra la imagen modificada y el usuario decide si introducir esa componente de la contraseña mediante la tecla “f”. Una vez el usuario pulsa esta tecla, el programa automáticamente comprueba qué tipo de polígono es el elemento mostrado por pantalla. Esta comprobación depende del número de esquinas y de los ángulos que tienen los puntos del polígono.
Por cada componente acertada de la contraseña el sistema muestra un texto diciendo que dicha forma es correcta. Si no es correcta, el sistema también avisa al usuario.
Si todas las formas son correctas se permite al usuario acceder al usuario al programa en sí.
Las funciones usadas para la secuencia de seguridad son:

- shi_tomasi_corner_detection()
- ordenar_puntos()
- calcular_angulos()
- clasificar_poligono()



Detección de esquinas del polígono

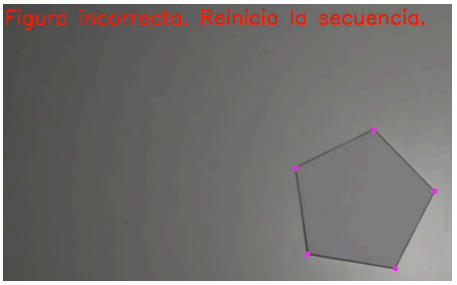


Figura Incorrecta

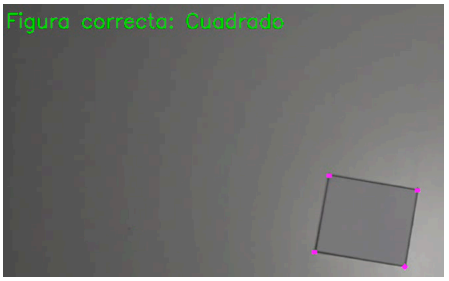


Figura Correcta

SISTEMA PROPUESTO

El sistema propuesto en este proyecto, como ya se ha mencionado previamente, es un detector de elementos que se desplazan por la imagen. Para hacerlo mucho más intuitivo, se ha creado un programa que analiza si los coches han pasado por una determinada línea de meta.

Para poder analizar la posición de los coches se ha utilizado una segmentación por color que permite ver qué coche ha pasado por la meta. Para el ejemplo mostrado en el vídeo se han utilizado un coche azul y otro verde. Para cada uno de estos coches se ha diseñado una máscara específica que permite diferenciarlos. Además, para la línea

de meta se ha usado un pedazo de papel pintado de color negro y que también es detectado por el programa mediante otra máscara.

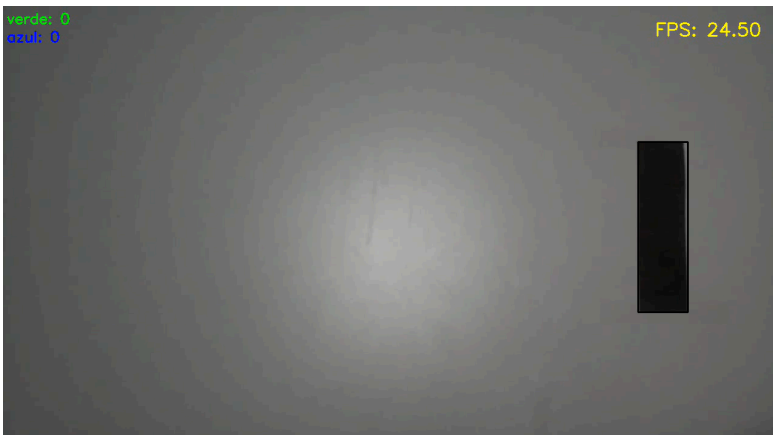
Una vez identificados los distintos elementos de la escena se diseña un rectángulo para cada uno de ellos. Este rectángulo permite saber si el elemento ha cruzado la franja de meta. Esto se calcula viendo si el punto medio del rectángulo de los coches se encuentra dentro del rectángulo correspondiente a la línea de meta. Si es así, el programa suma uno a un contador global que cuenta el número de "vueltas" que el coche ha dado por el "circuito". Además, para tener más información sobre los coches, mediante un medidor de tiempo conseguimos saber cuánto tiempo ha tardado el coche en dar esa última vuelta. El número de vueltas y el tiempo que tarda el coche en realizarlas aparecen por pantalla constantemente.

Un problema que surgía al realizar esto es que, como los coches que se utilizaron para simular la carrera tenían partes negras, dependiendo de la iluminación podía ocurrir que la línea de meta se detectase en el propio coche. Esto provocaba que ese coche comenzase a sumar vueltas de manera descontrolada. Para solucionar esto se decidió que el área de estos rectángulos debía superar un cierto umbral.

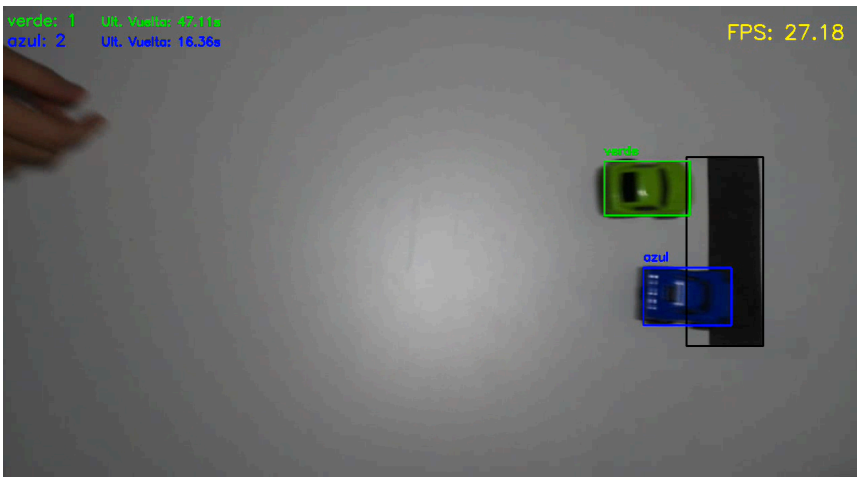
En resumen, el flujo de este programa consistía en los siguientes pasos:

1. Obtener el frame de la imagen
2. Detectar la meta: mediante la función "detectar_meta()".
3. Detectar los coches: mediante la función "detectar_colores()".
4. Análisis de la escena: mediante la función "seguir_objetos()".

Estos pasos se seguían constantemente mediante un bucle que finaliza cuando el usuario presiona la tecla "q".



Detección de la línea de meta



Detección de los coches y contadores de tiempo y vueltas

RESULTADOS Y DIFICULTADES

Los resultados obtenidos mediante este programa fueron los esperados. Sin embargo, durante el proceso de realización hubo diferentes inconvenientes que dificultaron la finalización del mismo.

El mayor problema se debió a la iluminación. Este problema se presentaba sobre todo en el apartado de la detección de esquinas en la contraseña. Debido a que el método de obtención de esquinas se realiza mediante el uso del gradiente de la intensidad de los puntos de la imagen, el hecho de que aparecieran sombras o puntos con una mayor iluminación perjudicaba el resultado, obteniendo falsos positivos. Para solventar este problema se tuvo que buscar un lugar oscuro y encender una única luz justo detrás de la cámara de la Raspberry Pi. Además, jugando con las limitaciones de la propia cámara se decidió alejar la cámara lo máximo posible de las figuras para que no fuese capaz de detectar las imperfecciones del diseño.

Otra dificultad que se presentó en el proyecto fue la detección de los colores mediante las máscaras. Existían algunos colores como el rojo que no podían ser capturados fácilmente, pues la máscara también detectaba otros colores del fondo o de otros coches. Por ello, se tomó la decisión de evitar estos colores y seleccionar otros más llamativos como el verde mostrado en las imágenes anteriores.

En conclusión, las mayores dificultades de este proyecto no residían en los códigos en sí, sino en los problemas que generaban las imágenes a detectar.

FUTUROS DESARROLLOS

Este proyecto puede tener múltiples aplicaciones en el mundo del entretenimiento. En el ámbito de los coches de carreras esta herramienta podría ser de gran utilidad para detectar posibles dudas con respecto al momento de llegada de los coches.

Para poder aplicar este proyecto a estas situaciones serían necesarios varios cambios para garantizar su funcionalidad.
En primer lugar, nuestro proyecto se basa en el color del vehículo para poder detectar coches diferentes. Sin embargo, esto no sería de utilidad en caso de contar con múltiples vehículos del mismo color. Por ello, sería necesario recurrir a otros tipos de características para diferenciar los distintos coches. Algunas de ellas podrían ser la detección de esquinas para conocer la forma del vehículo o incluso para conocer su matrícula del mismo.
Esto nos llevaría al segundo cambio. Dado que la cámara con la que se cuenta para realizar este proyecto no es la más moderna ni la más potente, cuando los coches alcanzan altas velocidades el programa encuentra serias dificultades para detectar los coches. Por ello, sería un requisito imprescindible mejorar tanto el hardware de la máquina como el de la cámara.