**INFORME PROYECTO FINAL LABORATORIO**

**Visión por Ordenador**

**Introducción:**

En este proyecto final hemos calibrado la cámara para posteriormente poder detectar patrones y extraer información, así como poder trackear objetos y dar funcionalidad adicional.

El sistema desarrollado implementa técnicas de detección de patrones y extracción de información para implementar una solución robusta de seguridad. Por un lado, identifica figuras geométricas en tiempo real, haciendo uso de detección de bordes y de los contornos. Por otro lado, extrae la información y la almacena en una lista ordenada, asegurando que el acceso solo sea permitido si la secuencia ingresada coincide con una contraseña predefinida. El sistema propuesto es un radar de semáforo, que trackea los coches, detecta cuando cruzan y detecta si el semáforo está en verde o en rojo.

**Metodología:**

* **Calibración de la Cámara:**

Para calibrar la cámara en primer lugar hemos tomado varias fotos desde varios ángulos a un tablero de ajedrez. Posteriormente, siguiendo la práctica del laboratorio 1 hemos calibrado la cámara obteniendo los resultados de la Figura 1. El error de reproyección es aceptable.

Observación: los coeficientes de distorsión indican que apenas hay distorsión tangencial, pero sí que hay distorsión radial, obvio ya que la cámara tiene como un recubrimiento esférico.

Texto

Descripción generada automáticamente

*Figura 1: Resultados de calibrar la cámara.*

* **Diagrama de Bloques del Sistema:**

Diagrama bloques Sistema Propuesto

**Diagrama

Descripción generada automáticamente**

Secuencia de transformación de la imagen en cada bloque del diagrama:

**Interfaz de usuario gráfica, Diagrama

Descripción generada automáticamente**

**Interfaz de usuario gráfica

Descripción generada automáticamente con confianza baja**

**Interfaz de usuario gráfica

Descripción generada automáticamente**

* **Secuencia de Transformación de la Imagen:**

La secuencia de transformación de la imagen comienza con la captura de cada frame. Para esto se utiliza cv2.VideoCapture que nos permite analizar cada frame de forma individual. Una vez conseguido el frame, la imagen en color se convierte a escala de grises utilizando la función cv2.cvtColor. Con esto se consigue eliminar el color y simplificar el análisis, lo que permite centrarse mejor en las formas y patrones.

Después de convertir la imagen a escala de grises, se aplica un filtro Gaussiano utilizando cv2.GaussianBlur para suavizar la imagen y reducir el ruido presente en ella. Una vez procesada la imagen, se utiliza el algoritmo de detección de bordes de cv2.Canny este algoritmo detecta cambios bruscos en la intensidad de los píxeles.

A continuación, se extraen los contornos de las figuras presentes en la imagen utilizando la función cv2.findContours. Mediante la función cv2.contourArea, se evalúa el área de cada contorno y se descartan aquellos que no superen los 1000 píxeles, con el fin de eliminar aquellos posibles errores que podrían captarse.

Con los contornos válidos, se aplica la función cv2.approxPolyDP para aproximar cada contorno a un polígono simplificado. Esta técnica ajusta los bordes de la figura detectada y calcula su número de vértices, que es crucial para clasificar la forma geométrica.

* **Sistema de Seguridad:**

El sistema de seguridad del proyecto se basa en dos funcionalidades: la detección de figuras y la extracción de información basada en una secuencia de figuras detectadas, las cuales actúan como una contraseña para validar el acceso al tracker.

En la detección de patrones, como se ha explicado en el apartado anterior, se hace un largo procesado de la imagen, hasta llegar al número de vértices de cada polígono, lo que determina la clasificación de la figura geométrica. Una característica clave del sistema es el uso de centroides para evitar el registro duplicado de figuras dentro de un mismo cuadro de video (ya que al ser live cada frame detectaba la misma figura y se añadía como nueva). Después de detectar y clasificar un contorno válido, se calcula el centroide de la figura utilizando los momentos geométricos del contorno. Antes de registrar una nueva figura, el sistema compara la posición del centroide con los ya almacenados en el mismo cuadro utilizando la distancia euclidiana. Si la distancia entre el centroide nuevo y uno ya existente es menor a 50 píxeles, la figura se considera un duplicado y no se registra. Gracias a este método se reducen mucho los posibles errores.

En lo que respecta a la extracción de información, las figuras detectadas se utilizan para construir una lista que simula una contraseña. El sistema espera una combinación específica de figuras geométricas en un orden predefinido, como ["Cuadrado", "Triángulo", "Hexágono", “Triángulo]. Cada figura nueva se añade a la lista, y una vez que esta contiene cuatro elementos, el sistema evalúa si la secuencia coincide. Si es incorrecto, se muestra un mensaje de “Contraseña incorrecta” y se reinicia la lista para probar otro intento. Por otro lado, si es correcto, se muestra "Contraseña correcta" y el sistema se considera desbloqueado.

* **Sistema Propuesto:**

El sistema propuesto es una de las dos partes que conforman un radar de semáforo, 1.) un tracker de vehículos que detectan cuando cruzan y 2.) el lector de la matrícula del vehículo que ha cruzado inapropiadamente. En este proyecto se ha desarrollado la parte 1.

El sistema se puede alimentar con frames provenientes de un video (nuestro caso) o de frames en tiempo real. Se parte de la idea de que la cámara está fija y apuntando a la carretera, por lo que por defecto se fijan parámetros como la zona en la que trackeará y las coordenadas en las que está el paso de peatones. Debido a la complejidad de mantener la cámara fija el sistema propuesto es muy difícil hacerlo en tiempo real, por lo que se alimentará de un video, en el que se mueve ligeramente la cámara, por lo que hay varias zonas en las que se buscarán coches a trackear (para una aplicación real hay que modificar el código).

Explicación del algoritmo:

* Para detectar si el semáforo está en rojo, se extrae una ventana en la que está el semáforo y se aplica una máscara para extraer el rojo. Finalmente se pasa esa ventana a RGB y comparando proporciones de colores se decide si está en verde o en rojo.
* Para detectar un coche en primer lugar se define el rectángulo por el que debe circular. Posteriormente se aplica una máscara sobre el frame quedándose únicamente la carretera (en nuestro caso es blanca), por lo que cuando pasa un coche hay un rectángulo negro de entre todo blanco.

El algoritmo actual para extraer la bounding box inicial del coche obtiene las coordenadas mínimas y máximas donde hay píxeles negros (al frame se le ha aplicado dilataciones para evitar que haya píxeles negros sueltos), para evitar bounding boxes raras se establece la necesitas que el largo entre el ancho esté dentro de un rango [1.25, 1.65]. Este algoritmo es muy específico, no puede detectar varios coches a la vez en el mismo carril ni con varios carriles, debería ampliarse la funcionalidad (ver en futuros desarrollos).

Una vez se tiene la bounding box inicial, se inicia un filtro de Kalman (6, 2), que tiene en cuenta la aceleración. Mientras se está trackeando un coche no se ejecuta el encontrar la bounding box mínima. Cuando la posición del objeto trackeado supera el límite de donde está el cruce deja de trackear el vehículo y dependiendo del color del semáforo imprime un mensaje u otro.

**Resultados:**

* **Sistema Seguridad:**

Texto

Descripción generada automáticamente **Una pantalla de un celular con la imagen de una persona

Descripción generada automáticamente con confianza media**

* **Sistema Propuesto:**

**Imagen de la pantalla de un video juego

Descripción generada automáticamente con confianza baja Imagen que contiene verde, pequeño, tabla, refrigerador

Descripción generada automáticamente**

**Imagen que contiene Diagrama

Descripción generada automáticamente Imagen que contiene interior, verde, gato, tabla

Descripción generada automáticamente**

**Futuros Desarrollos:**

1. Se puede hacer el detector de matrículas para completar el radar se semáforo.
2. Se puede hacer más sofisticado el sistema de detección de bounding boxes para que aplique a varios carriles y coches. Podría utilizarse k-means clustering para.