

### § Instrucciones §

- El código debe organizarse en scripts de Python (.py) debidamente documentados
  - Cada parte del proyecto tendrá asociado un script que genere los resultados
- **Se debe incluir un archivo README.md** que explique la estructura del proyecto, cómo ejecutar los scripts y cualquier dependencia necesaria.
- El archivo README.md debe contener una sección de Referencias donde se detallen todos los recursos utilizados para la resolución del proyecto. En el caso de utilizar modelos de lenguaje (LLMs) como ChatGPT debe indicarse tanto el prompt como la respuesta del mismo.
- **Se debe incluir un vídeo explicando los aspectos críticos del proyecto.** El objetivo no es explicar línea por línea el código (que deberá estar debidamente documentado), sino establecer una clara relación entre las partes relevantes del código y los procedimientos. **El vídeo deberá tener una duración máxima de 8 minutos.** El video debe incluir el audio y vídeo del alumno.
- No está permitido el uso de código procedente de referencias o modelos de lenguaje (LLMs) como ChatGPT o Github Copilot.
- Los elementos requeridos como resultados deben ser exportados a una carpeta denominada **resultados**.
- Todos los ficheros deben entregarse comprimidos en un fichero .zip.

#### Uso de Funciones Externas

Se pueden utilizar todas las funciones de las librerías `numpy`, `scipy`, `matplotlib` y las siguientes funciones de OpenCV:

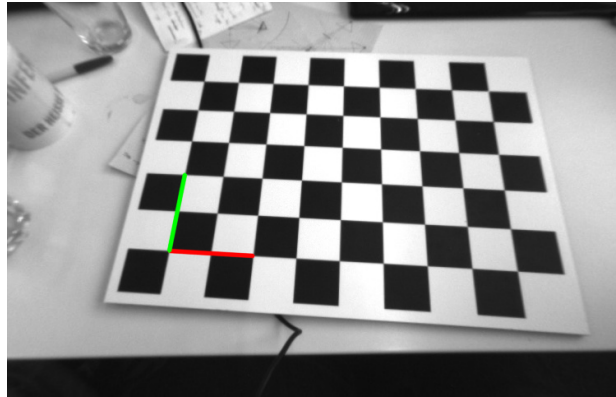
- `imread`, `imwrite`
- `imshow`, `waitKey`, `destroyAllWindows`
- `split`, `merge`, `cvtColor`
- `line`, `circle`, `rectangle`, `drawContours`, `putText`, `drawFrameAxes`, `Rodrigues`
- La clase `VideoWriter`
- `findChessboardCorners`
- Las funciones definidas en [camera\\_calibration\\_show\\_extrinsics.py](#)

El uso de otras funciones de OpenCV o librerías externas requiere de autorización previa.

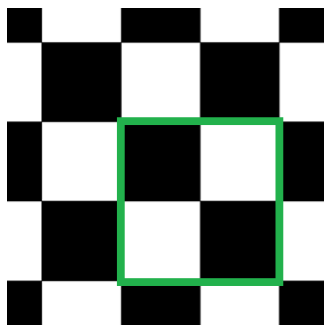
## Parte 1: Homografía y Proyección Planar (3 puntos)

Utilizando únicamente la imagen `datos\imagenes\img_001.jpg`:

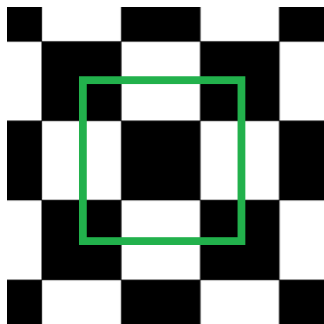
- (1) Calcular la homografía  $H$  entre el plano que contiene al *tablero de ajedrez* y el *plano de la pantalla*, situando el origen de coordenadas del plano que contiene al *tablero de ajedrez* en la esquina inferior interna izquierda del tablero (ver Figura 1) y teniendo en cuenta que el lado de cada casilla del tablero mide 4 cm.
- (2) Utilizar la homografía  $H$  obtenida en el apartado anterior, para proyectar un cuadrado de 8 cm de lado sobre el tablero. El cuadrado debe estar situado sobre 4 casillas.
- (3) Utilizar la homografía  $H$  obtenida en el apartado anterior, para proyectar un cuadrado de 8 cm de lado sobre el tablero. En este caso, los vértices del cuadrado deben estar situados en el centro de 4 casillas.



**Figura 1:** Origen de coordenadas del plano que contiene al *tablero de ajedrez*



**Figura 2:** Diagrama de la posición del cuadrado en el tablero del apartado (2)



**Figura 3:** Diagrama de la posición del cuadrado en el tablero del apartado (3)

**Resultados Parte 1**

- `resultados\parte1_H.txt`: Fichero de texto que contiene la matriz de homografía del apartado (1) exportada con la función `numpy.savetxt`. La matriz homogénea debe escalarse para que el último elemento sea 1.
- `resultados\parte1_2.png`: Imagen resultante del apartado (2).
- `resultados\parte1_3.png`: Imagen resultante del apartado (3).

**Parte 2: Video (2 puntos)**

Replicar el proceso de la **Parte 1** para cada uno de los fotogramas `img_XXX.jpg` de la carpeta `datos\imagenes` y generar un video a 30 fotogramas por segundo con las imágenes resultantes. *Nota: En el caso de que no sea posible detectar el tablero de ajedrez en alguno de los fotogramas, incluir el fotograma original sin el cuadrado.*

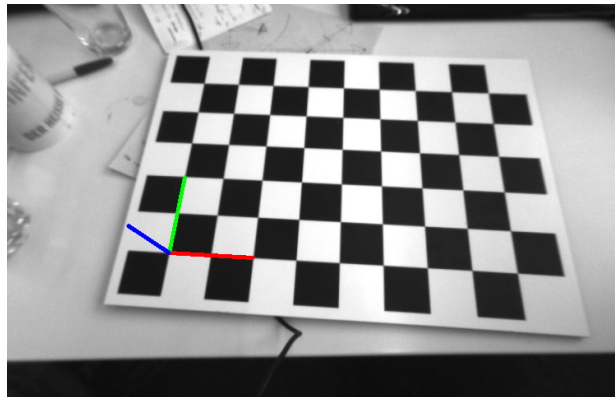
**Resultados Parte 2**

- `resultados\parte2.mp4`: Video resultante.

**Parte 3: Pose de la Cámara (3 puntos)**

Dada la componente intrínseca o matriz de calibración de la cámara  $K$ , definida en el fichero `datos\K.txt` y la matriz de homografía  $H$  del apartado (1) de la **Parte 1**:

- (1) Calcular los parámetros extrínsecos de la cámara  $R$  y  $t$  que se obtienen al situar el origen de coordenadas en la esquina inferior interna izquierda del tablero (ver Figura 4). Comprobar si la matriz de rotación obtenida  $R$  es una verdadera matriz de rotación.



**Figura 4:** Origen de coordenadas de la escena.

- (2) Si la matriz de rotación obtenida  $R$  en el apartado (1) no es una verdadera matriz de rotación, calcular de nuevo los parámetros extrínsecos de la cámara  $R$  y  $t$  utilizando un método que asegure que es una verdadera matriz de rotación.
- (3) Utilizar la función `drawFrameAxes` de OpenCV y los resultados del apartado (2) para dibujar los ejes en el origen de coordenadas sobre la imagen `datos\imagenes\img_001.jpg`.

**Resultados Parte 3**

- `resultados\parte3_1_R.txt`: Fichero de texto que contiene la matriz de rotación  $R$  del apartado (1) exportada con la función `numpy.savetxt`.
- `resultados\parte3_1_t.txt`: Fichero de texto que contiene el vector de traslación  $t$  del apartado (1) exportado con la función `numpy.savetxt`.
- `resultados\parte3_2_R.txt`: Fichero de texto que contiene la matriz de rotación  $R$  del apartado (2) exportada con la función `numpy.savetxt`.
- `resultados\parte3_2_t.txt`: Fichero de texto que contiene el vector de traslación  $t$  del apartado (2) exportado con la función `numpy.savetxt`.
- `resultados\parte3.png`: Imagen resultante del apartado (3).

**Parte 4: Proyección de un Cubo (2 puntos)**

Calcular la matriz de proyección  $P = K[R|t]$  para cada uno de los fotogramas `img_XXX.jpg` de la carpeta `datos\imagenes` y utilizarla para proyectar un cubo de lado 8 cm sobre 4 casillas del *tablero de ajedrez*. Generar un video a 30 fotogramas por segundo con las imágenes resultantes. *Nota: En el caso de que no sea posible detectar el tablero de ajedrez en alguno de los fotogramas, incluir el fotograma original sin el cubo.*

**Resultados Parte 4**

- `resultados\parte4.mp4`: Video resultante.

**Parte 5: Visualización de la Cámara en la Escena (2 puntos, Opcional)**

Estimar la pose de la cámara para cada uno de los fotogramas `img_001.jpg` a `img_360.jpg` de la carpeta `datos\imagenes`, y generar un video a 30 fotogramas por segundo que muestre el movimiento de la cámara con respecto al tablero de ajedrez.

**Resultados Parte 5**

- `resultados\parte5.mp4`: Video resultante.