

# Lab04

1. Camera Calibration (50%)
2. Warping practice (50%)

# How to get image from webcam?

```
import cv2

cap = cv2.VideoCapture(1) #device

while(True):

    ret, frame = cap.read()

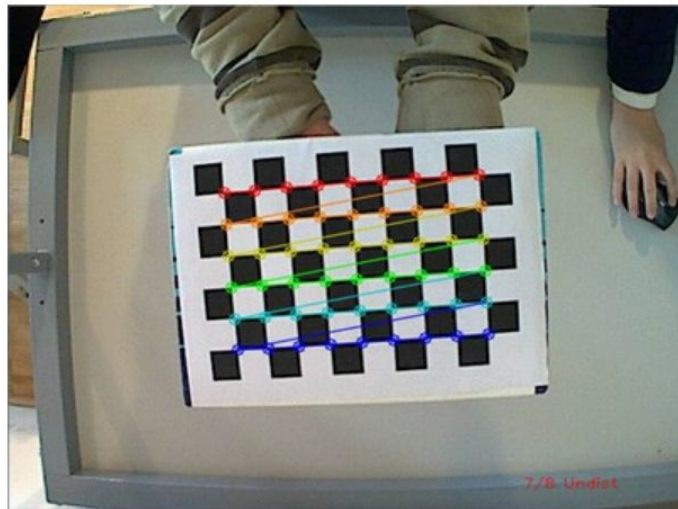
    #ret is True if read() succeeded

    cv2.imshow('frame', frame)

    cv2.waitKey(33)
```

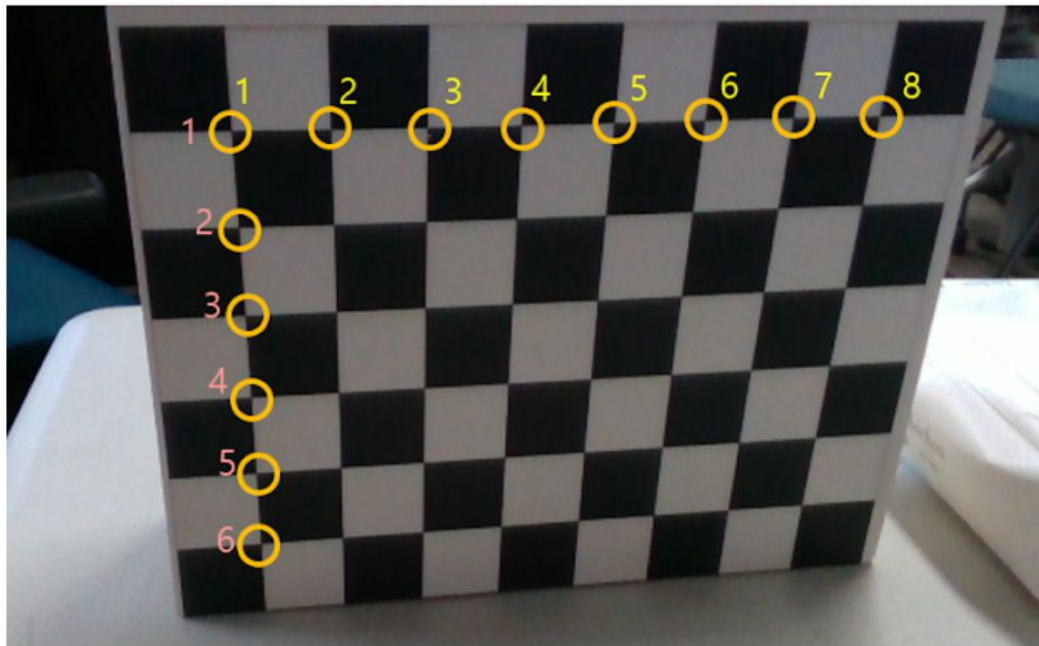
# 1. Camera Calibration(50%)

1. 假設好棋盤格的object point
2. 利用webcam讀取即時影像, 將影像轉成灰階
3. 拍攝棋盤格, 若有偵測到則儲存該影像中棋盤格的image point
4. 當儲存影像多於四張時, 開始計算參數
5. 得到參數並儲存於xml檔



# 1. Camera Calibration(50%)

- 假設棋盤格的object point,  $z = 0$
- Prepare object points, like  $(0,0,0)$ ,  $(1,0,0)$ ,  $(2,0,0)$  ....,  $(7,5,0)$



# 1. Camera Calibration(50%)

- `ret, corner = cv2.findChessboardCorners(image, patternSize, None)`
  - `patternSize` – Number of inner corners per a chessboard row and column ( `patternSize = cvSize(points_per_row,points_per_colum) = cvSize(columns,rows) ).`
  - `ret == True`, chessboard detected
- `cv2.cornerSubPix(image, corners, winSize, zeroZone, criteria)`
  - `image` – Input image.
  - `corners` – Initial coordinates of the input corners and refined coordinates provided for output.
  - `winSize` - (11, 11)
  - `zeroZone` - (-1,-1)
  - `criteria` - `criteria = (cv2.TERM_CRITERIA_EPS + cv2.TERM_CRITERIA_MAX_ITER, 30, 0.1)`

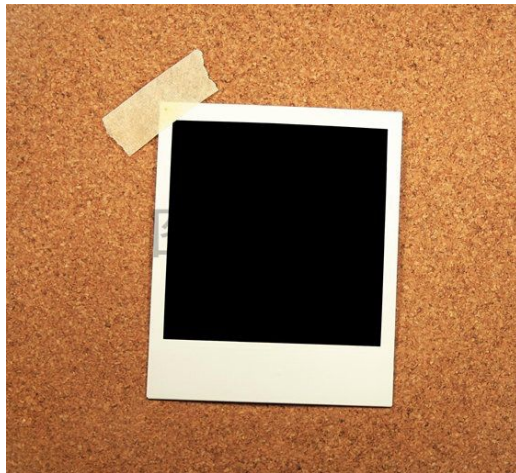
# 1. Camera Calibration(50%)

- `ret, cameraMatrix, distCoeffs, rvecs, tvecs = cv2.calibrateCamera(objectPoints, imagePoints, imageSize, None)`
  - **cameraMatrix** – Output 3x3 floating-point camera matrix
  - **distCoeffs** – Output vector of distortion coefficients
  - `rvecs, tvecs` - rotation and translation matrix
  - 有多少組imagepoint就要有多少組objectpoint
- `f = cv2.FileStorage(filename, cv2.FILE_STORAGE_WRITE)`
  - `f.write("intrinsic", mtx) //cameraMatrix`
  - `f.write("distortion", dist) //distCoeffs`
  - `f.release()`

## 2. Warping (50%)

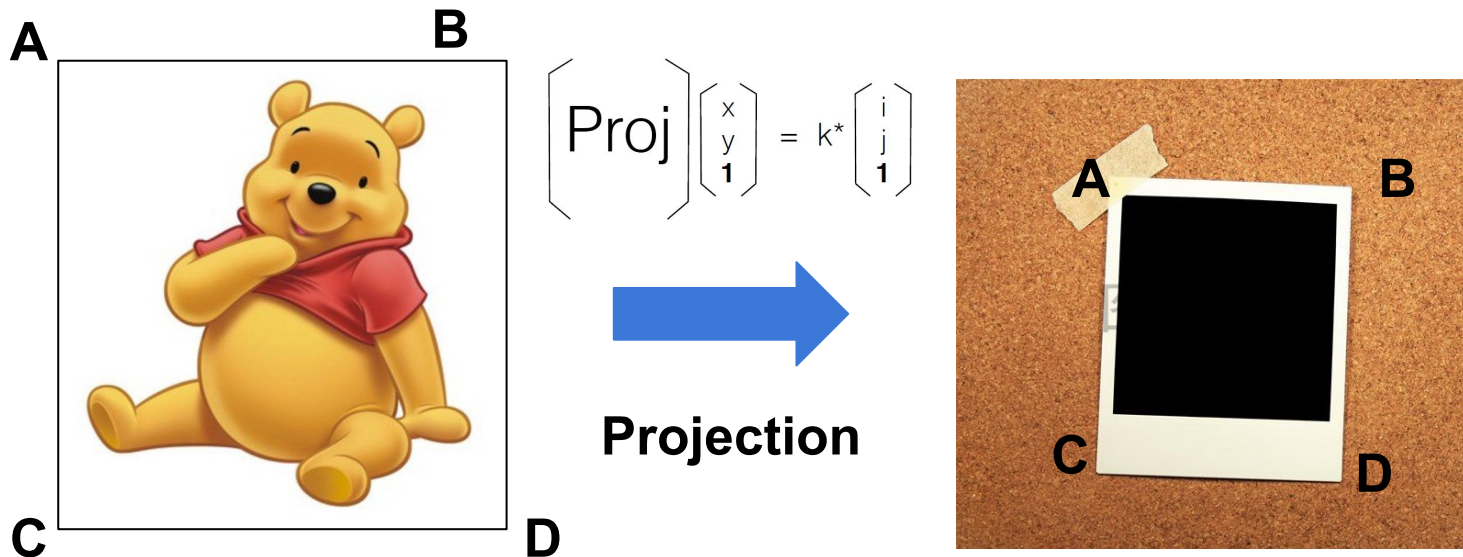


Projection



透視變換 (Perspective Transformation) 是將成像投影到一個新的視平面 (Viewing Plane)，也稱作投影映射 (Projective Mapping)

## 2. Warping (50%)

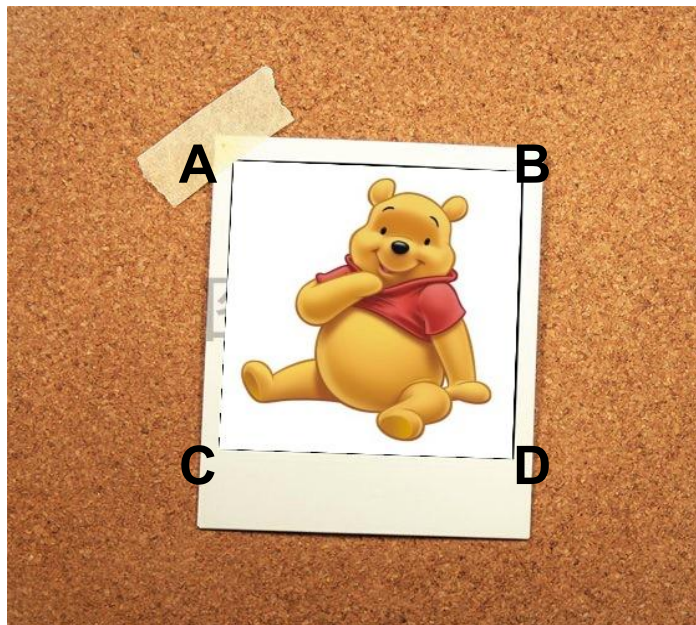
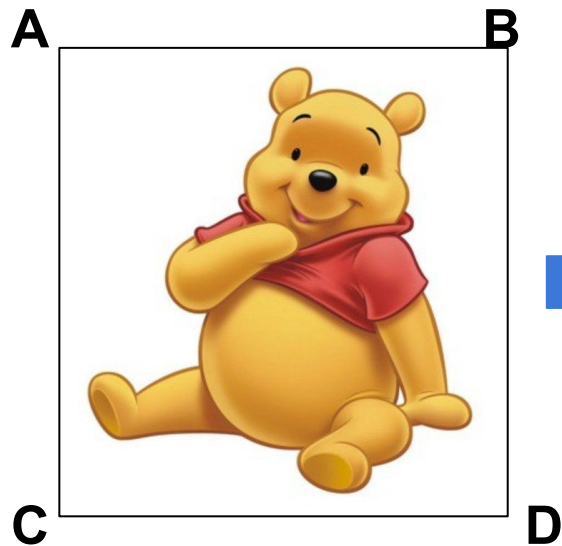


`cv2.getPerspectiveTransform(cap_corner, img_corner)`

- `cap_corner` , `img_corner` 為四個點的陣列, 順序需要兩兩相對
- 返回一個3x3的matrix



## 2. Warping (50%)



不能使用 `cv2.warpPerspective(src, M, dsize)`

- 返回轉換後的圖, 再將轉換圖貼上去
- 自己刻, 用bilinear interpolation將圖填上去

## 2. Warping (50%)

- 將webcam得到的即時影像warp到看板上
1. 得到兩張圖中對應的四個點
  2. 利用cv2.getPerspectiveTransform得到轉換關係
  3. 透過bilinear interpolation將圖適當的填上

