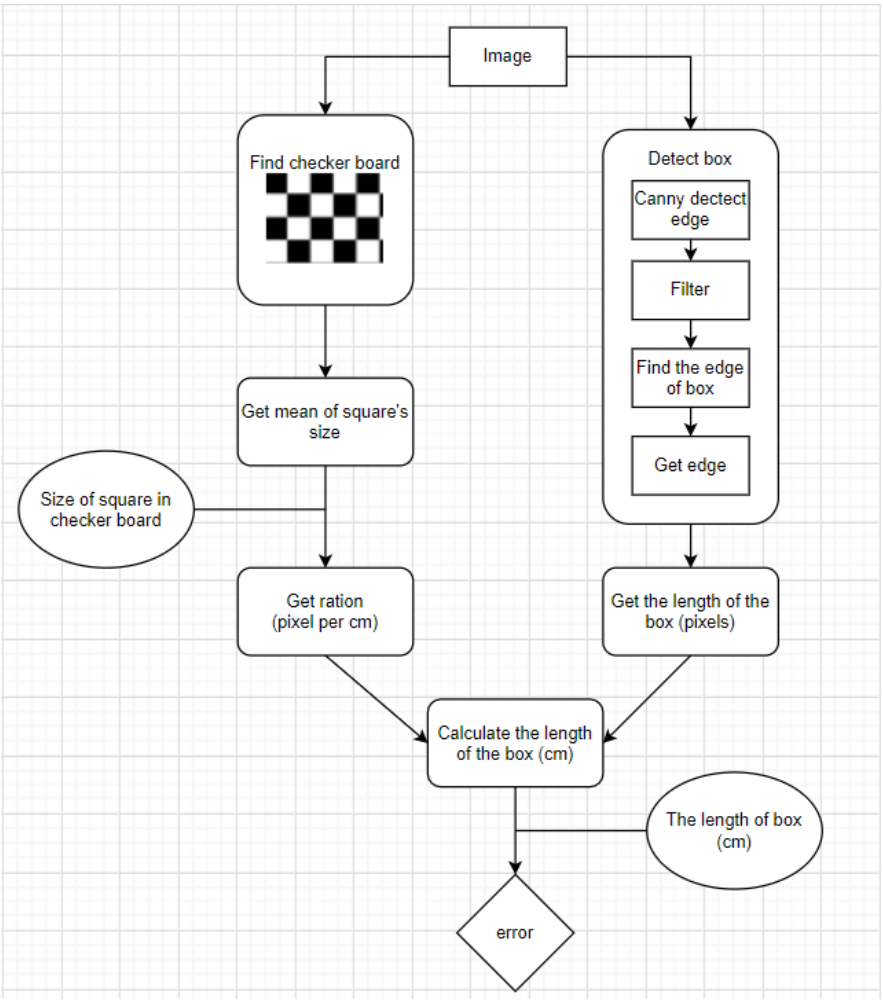
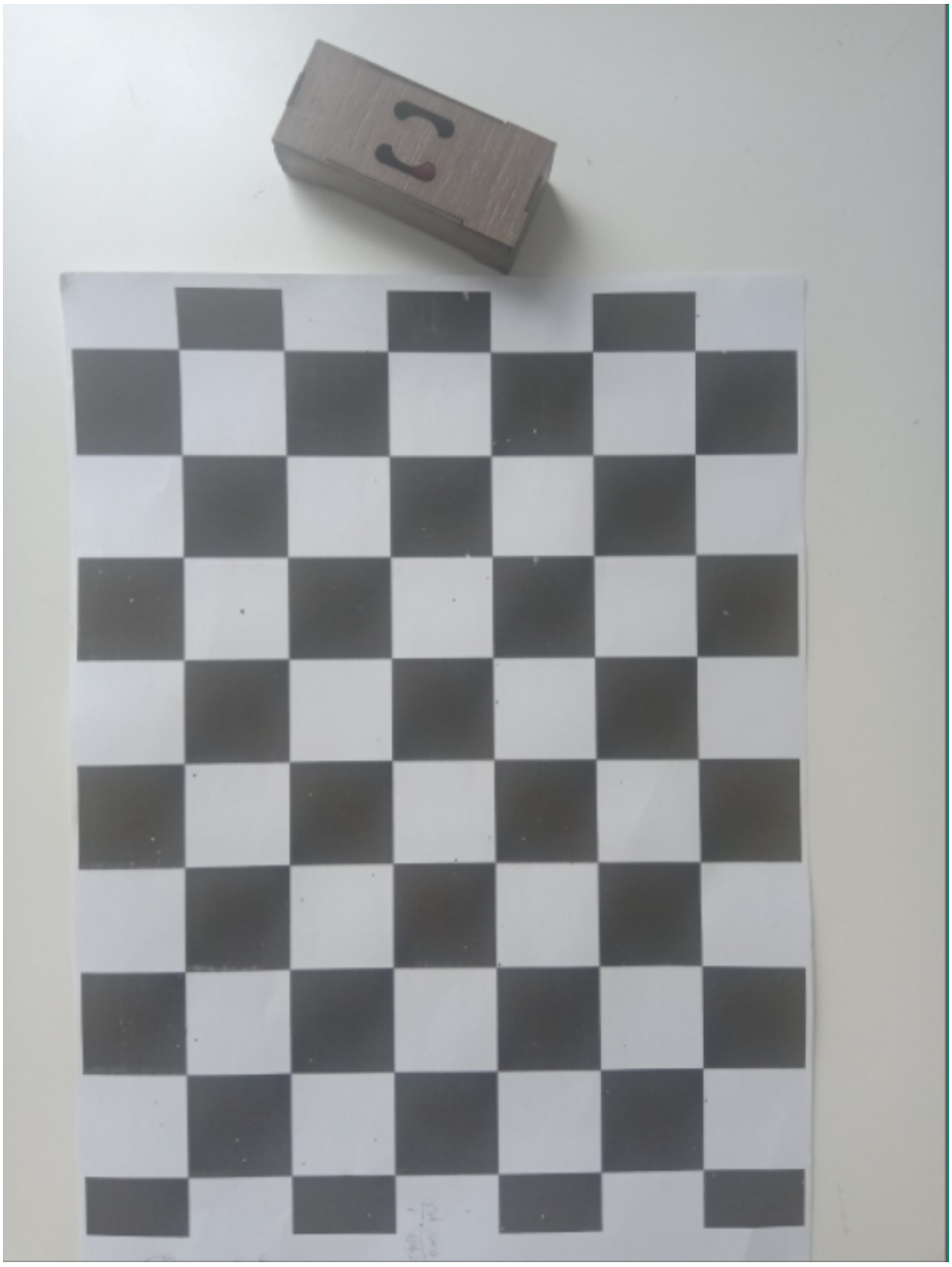
Môn học: Xử lí ảnh và thị giác Robot
Bài 2: Calibration

Thuật toán



Ảnh gốc

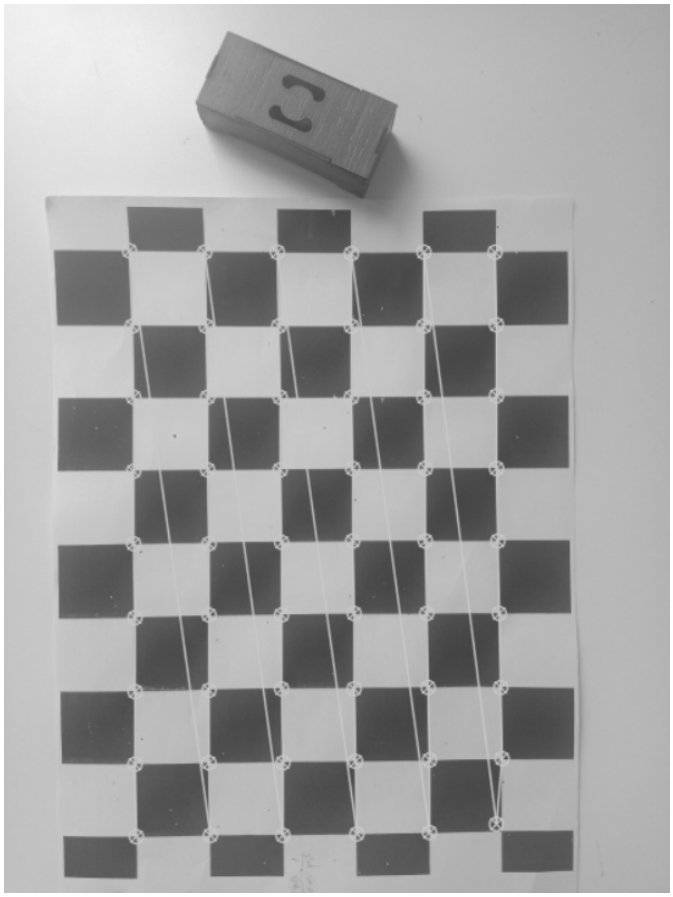


Bước 1: Find checkerboard

```python
def find_corners(img):
    img = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)

    ret, corners = cv2.findChessboardCorners(img, (9,6), None)

    if ret:
        cv2.drawChessboardCorners(img, (9,6), corners, ret)
        corners = np.resize(corners,(9, 6, 2))

    return img, corners
```

- Đầu vào là ảnh màu
- Đầu ra là ảnh đã detect các góc và corners là vị trí của các góc detect được

Kết quả:



Bước 2: Get mean of square's size

```python
def find_ratio(corners, real):
    rows, cols = corners.shape[:2]
    distances = []
    for i in range(rows-1):
        for j in range(cols-1):
            dis1 = find_pixel_distance(corners[i,j], corners[i,j+1])
            dis2 = find_pixel_distance(corners[i,j], corners[i+1,j])
            if(dis1<100):
                distances.append(dis1)
            if dis2<100:
                distances.append(dis2)
    distances = np.round(distances)
    ave = np.ceil(np.sum(distances)/len(distances))
    # ave = distances[2]
    print("Average pixel: ", ave)
    return ave/real
```

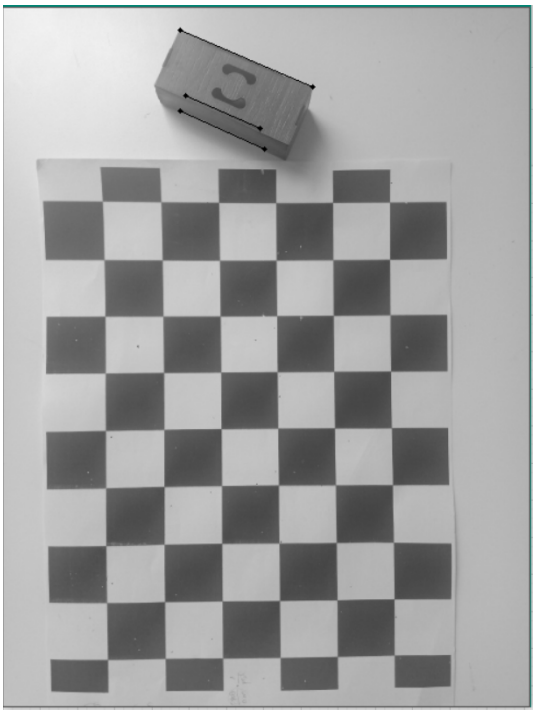
- Đầu vào là vị trí của các góc và giá trị đo được thực tế giữa hai góc kề nhau (một cạnh của hình vuông)
- Tìm giá trị trung bình giữa khoảng cách hai góc liền kề
- Đầu ra là tỉ lệ pixel/cm

Bước 3: Detect box:

```python
def find_edge(img):

    edges = cv2.Canny(img,50,150,apertureSize = 3)

    lines = cv2.HoughLinesP(edges,1,np.pi/360,50,minLineLength=30,maxLineGap=10)

    lengths = []
    for line in lines:
        x1,y1,x2,y2 = line[0]
        pt1 = (x1, y1)
        pt2 = (x2, y2)
        dis = find_pixel_distance(pt1, pt2)
        deg = find_pixel_angle(pt1, pt2)
        # print("Distance: ", dis)
        # print("Deg: ", deg)
        if deg<-10 and deg>-80:
            lengths.append(dis)
            # print("Distance: ", dis)
            # print("Deg: ", deg)
            cv2.line(img,(x1,y1),(x2,y2),(0,255,0),1)
            cv2.circle(img, pt1, radius=0, color=(0, 0, 255), thickness=3)
            cv2.circle(img, pt2, radius=0, color=(0, 0, 255), thickness=3)

    return img, lengths
```

- Đầu vào là hình ảnh xám
- Trả về hình ảnh các biên tách được và mảng chứa độ dài các đường biên

Kết quả



Bước 4: Tính độ dài của box dựa trên tỉ lệ tính toán ở trên và so sánh với thực tế

```
m_dis = lengths[0]/ratio

r_dis = 7 # Real length of the ruler in cm
print("Real distance: {} cm".format(r_dis))
print("Measurement distance: {} cm".format(m_dis))
```

- Độ dài (cm) của box bằng độ dài (pixel) / tỉ lệ (pixel/cm)
- Độ dài thực tế của box là 7cm

Từ đó ta có được sai số

```
Average pixel:  53.0
Ratio:  17.966101694915253
Real distance: 7 cm
Measurement distance: 7.3471698113207555 cm
```

Average pixel : Số pixel trung bình tính được trên một cạnh của ô trong checkerboard
Ratio: Tỉ lệ pixel/cm
Real distance: độ dài thực của hộp
Measurement distance: độ dài tính toán của hộp
Sai số ~0.34 cm