無人機智慧系統開發與實作
System Development and Implementation of Drone Intelligence

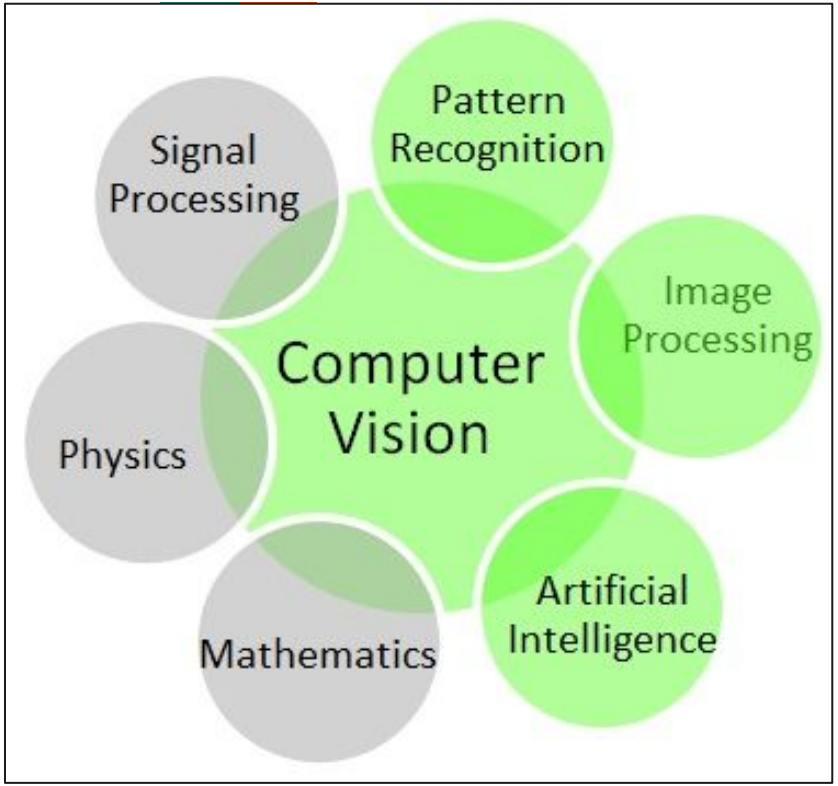
openCV 介紹與導入實作

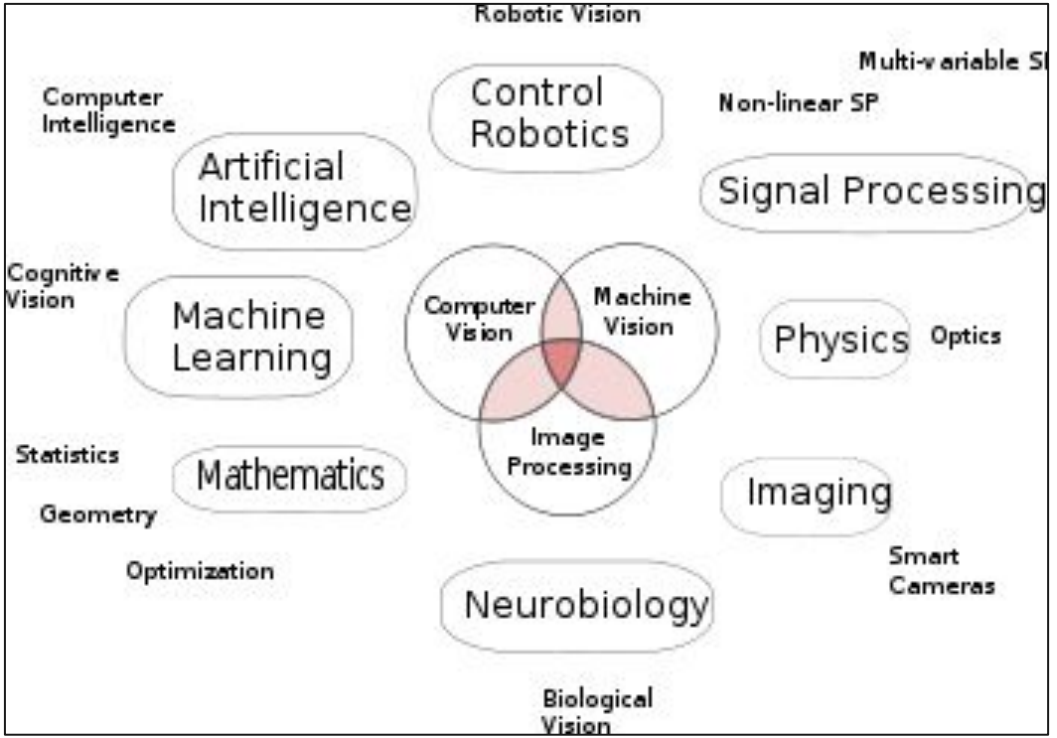
Computer Vision(CV)

利用電腦或是攝影機，代替人類的眼睛對目標進行辨識、跟蹤、測量等機器視覺，並更進一步的進行影像處理，使處理後的影像更適合人眼觀察或是儀器檢測。

電腦視覺研究的理論和技術，試圖建立能從影像獲取資訊的人工智慧系統。
電腦視覺也可以說是研究如何使人工系統從影像中「感知」的科學，因為感知可以看作是從感官訊號中提取資訊。

https://en.wikipedia.org/wiki/Computer_vision





https://www.kdnuggets.com/2016/08/seven-steps-understanding-computer-vision.html

openCV

open Computer Vision Library
在BSD(**B**erkeley **S**oftware **D**istribution license)許可下發布的，可以免費用於學術和商業用途。

具c++、python、java等多程式語言API。並支持Windows，Linux，Mac OS，iOS和Android。

用途範圍廣泛:
- interactive art
- inspection
- stitching maps on the web
- advanced robotics

https://opencv.org/
https://zh.wikipedia.org/wiki/BSD%E8%AE%B8%E5%8F%AF%E8%AF%81

安装opencv

```
anaconda:
 conda install --channel https://conda.anaconda.org/menpo opencv3

Ros:
 Already install
```

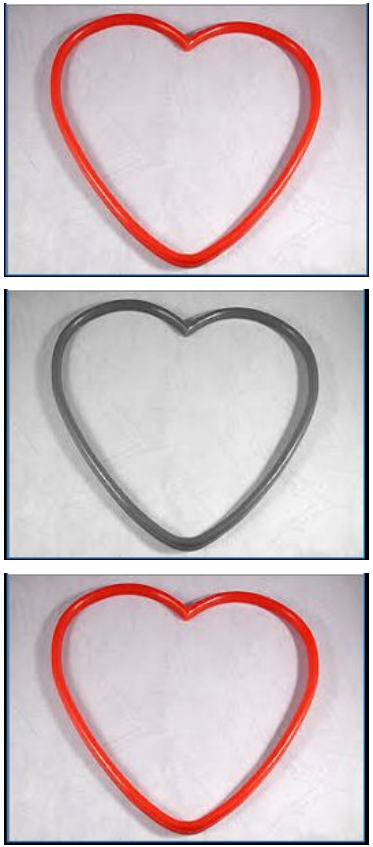
Read image & show, write

import cv2

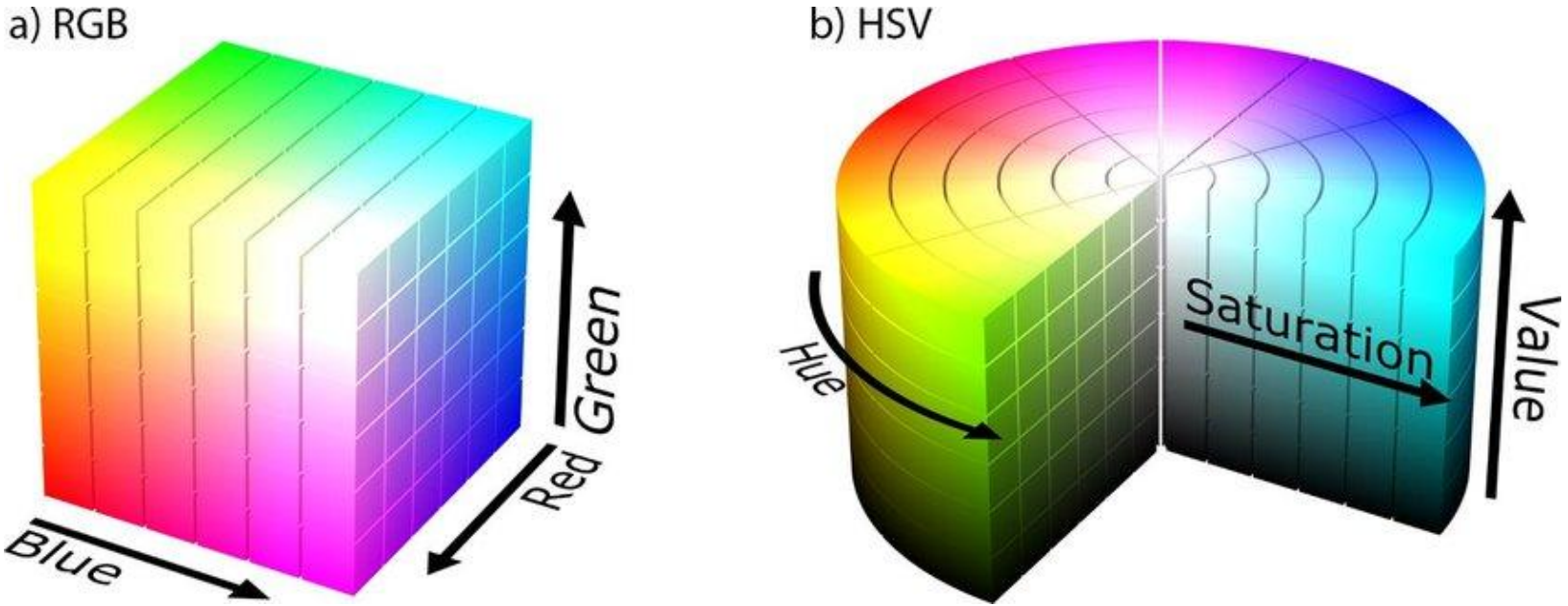
img = cv2.imread(FILE_NAME, Flag)

Flag:
cv2.IMREAD_COLOR
cv2.IMREAD_GRAYSCALE
cv2.IMREAD_UNCHANGED

cv2.imwrite(WRITE_FILE_NAME, image)

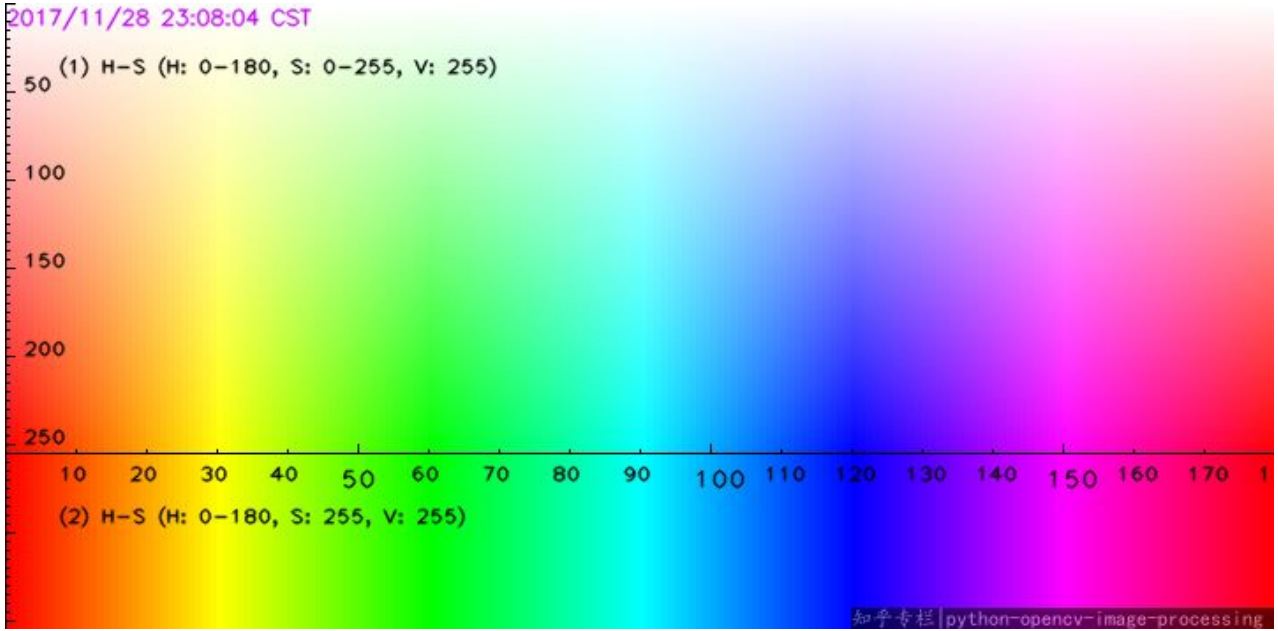


RGB vs HSV



Popov, V., Ostarek, M., & Tenison, C. (2018). Practices and pitfalls in inferring neural representations. *NeuroImage*, *174*, 340-351.

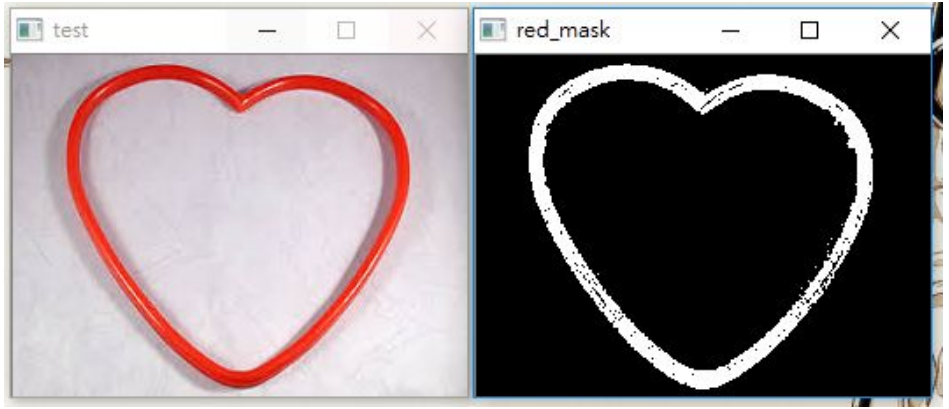
Detect Red



https://stackoverflow.com/questions/47483951/how-to-define-a-threshold-value-to-detect-only-green-colour-objects-in-an-image/47483966#47483966

Detect Red

```python
import cv2
import numpy as np

img = cv2.imread('red.jpg',  cv2.IMREAD_COLOR)
hsv_img = cv2.cvtColor(img, cv2.COLOR_BGR2HSV)
# red hsv range and mask on hsv_img
lower_red_0 = np.array([0, 70, 0])
upper_red_0 = np.array([5, 255, 255])
lower_red_1 = np.array([175, 70, 0])
upper_red_1 = np.array([180, 255, 255])
red_mask0 = cv2.inRange(hsv_img, lower_red_0, upper_red_0)
red_mask1 = cv2.inRange(hsv_img, lower_red_1, upper_red_1)
red_mask = cv2.bitwise_or(red_mask0, red_mask1)

# show red mask
cv2.imshow("red_mask", red_mask)
cv2.imshow("test", img)
cv2.waitKey(0)
```



find contour

find contour:
cv2.findContours(IMAGE, MODE, METHOD)
IMAGE: 單通道圖(灰度圖)
MODE:
CV_RETR_EXTERNA: retrieves only the extreme outer contours.
CV_RETR_LIST: retrieves all of the contours without establishing any hierarchical relationships.
CV_RETR_CCOMP: retrieves all of the contours and organizes them into a two-level hierarchy.
CV_RETR_TREE retrieves all of the contours and reconstructs a full hierarchy of nested contours.
METHOD:
CV_CHAIN_APPROX_NONE: stores absolutely all the contour points.
CV_CHAIN_APPROX_SIMPLE : compresses horizontal, vertical, and diagonal segments and leaves only their end points.

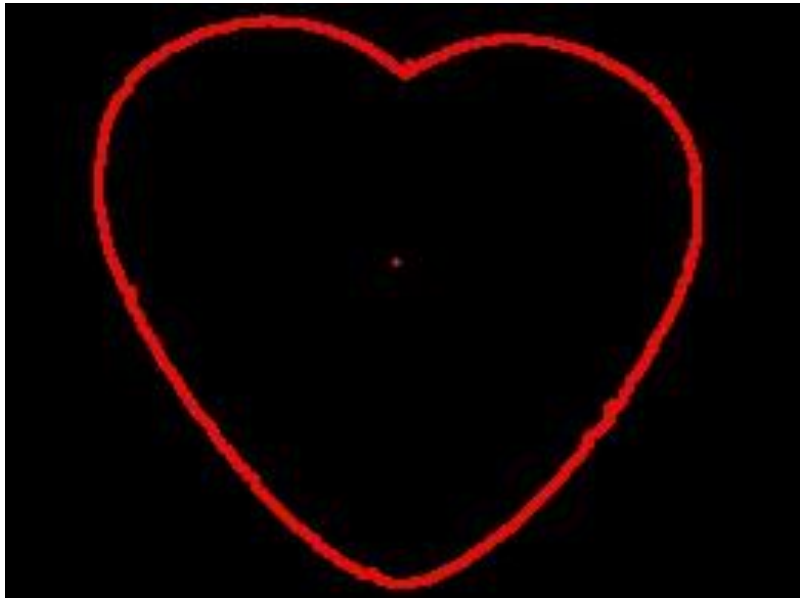
https://docs.opencv.org/2.4/modules/imgproc/doc/structural_analysis_and_shape_descriptors.html?highlight=findcontours#findcontours

find contour

```python
import cv2
import numpy as np

def findMask(img):
  lower_red_0 = np.array([0, 70, 0])
  upper_red_0 = np.array([5, 255, 255])
  lower_red_1 = np.array([175, 70, 0])
  upper_red_1 = np.array([180, 255, 255])
  red_mask0 = cv2.inRange(hsv_img, lower_red_0, upper_red_0)
  red_mask1 = cv2.inRange(hsv_img, lower_red_1, upper_red_1)
  red_mask = cv2.bitwise_or(red_mask0, red_mask1)
  return red_mask

img = cv2.imread('red.jpg')
print(img.shape)
hsv_img = cv2.cvtColor(img, cv2.COLOR_BGR2HSV)
# red hsv range and mask on hsv_img
red_mask = findMask(hsv_img)
print(red_mask.shape)
(contour_contours, contour_h) = cv2.findContours(red_mask, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_NONE)
contour_i = red_mask.copy()
contour_i = cv2.cvtColor(contour_i, cv2.COLOR_GRAY2BGR)
cv2.drawContours(contour_i, contour_contours, -1, (0, 0, 255), 2)
# show red mask
cv2.imshow("red_mask", red_mask)
cv2.imshow("test", contour_i)
cv2.imwrite("red_mask.jpg", red_mask)
cv2.imwrite("test.jpg", contour_i)
cv2.waitKey(0)
```

find centroid

重心: average of add contour point (x, y)

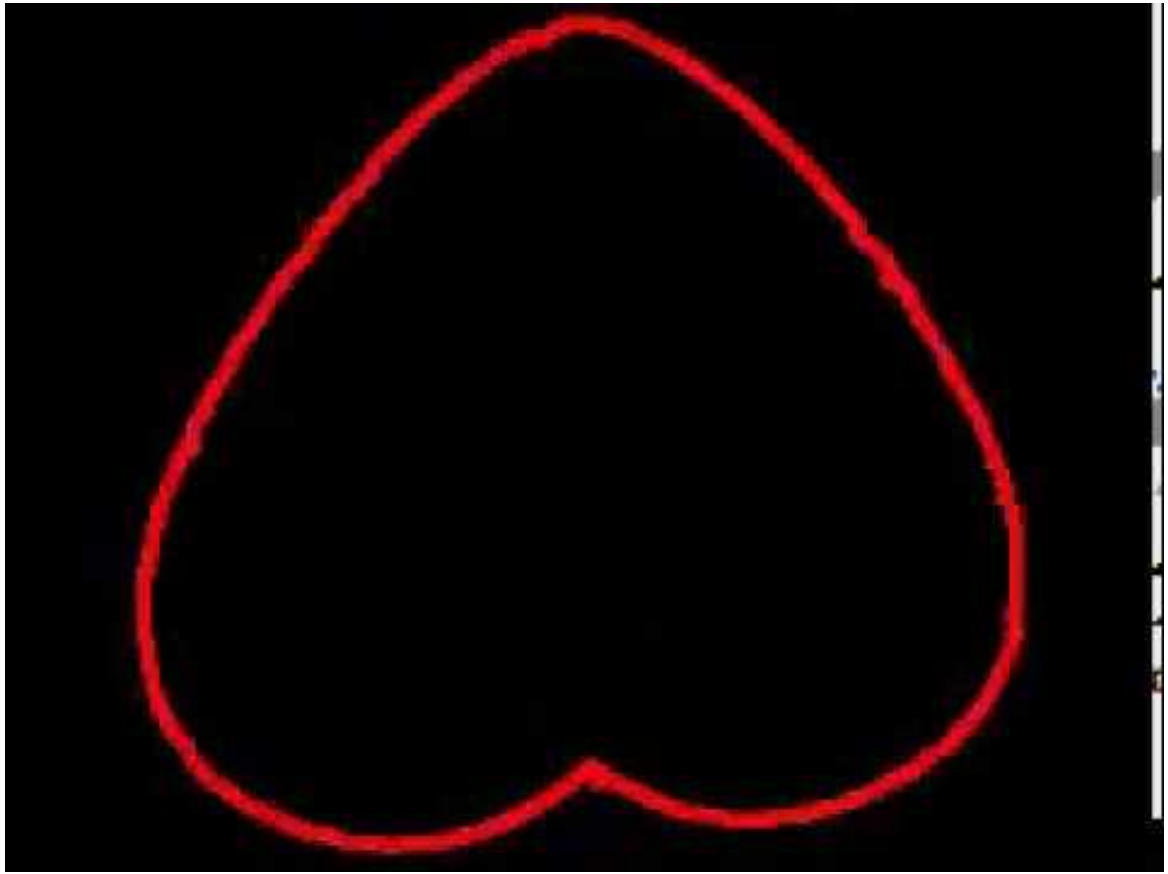


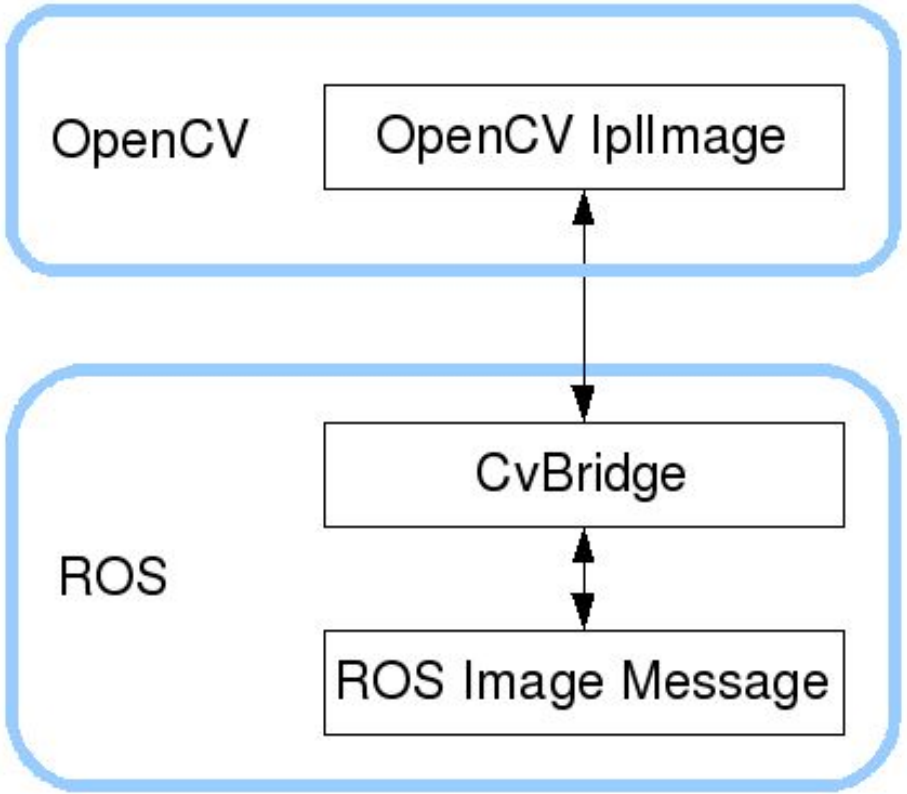
```
cv2.drawContours(contour_i, contour_contours, -1, (0, 0, 255), 2)
# show red mask
avg_x = []
avg_y = []
for cnt in contour_contours:
  for c in cnt:
    avg_x.append(c[0][0])
    avg_y.append(c[0][1])
print(np.mean(avg_x))
print(np.mean(avg_y))
cv2.circle(contour_i, (int(np.mean(avg_x)), int(np.mean(avg_y))), 1, (0,0,255), -1)
cv2.imshow("test_center", contour_i)
cv2.imwrite("test_center.jpg", contour_i)
cv2.waitKey(0)
```

video capture

cv2.VideoWriter_fourcc('X',"V",'I','D')
cv2.VideoWriter(NAME, fourcc, FPS,(width, height))
Writer.write(img)

```python
import cv2
import numpy as np

def findMask(img):
  lower_red_0 = np.array([0, 70, 0])
  upper_red_0 = np.array([5, 255, 255])
  lower_red_1 = np.array([175, 70, 0])
  upper_red_1 = np.array([180, 255, 255])
  red_mask0 = cv2.inRange(hsv_img, lower_red_0, upper_red_0)
  red_mask1 = cv2.inRange(hsv_img, lower_red_1, upper_red_1)
  red_mask = cv2.bitwise_or(red_mask0, red_mask1)
  return red_mask

img = cv2.imread('red.jpg')
fourcc = cv2.VideoWriter_fourcc('X',"V",'I','D')
print((img.shape[1], img.shape[0]))
out = cv2.VideoWriter('test.avi', fourcc, 20.0,(img.shape[1], img.shape[0]))

print(img.shape)
hsv_img = cv2.cvtColor(img, cv2.COLOR_BGR2HSV)
# red hsv range and mask on hsv_img
red_mask = findMask(hsv_img)
print(red_mask.shape)
(contour_i, contour_contours, contour_h) = cv2.findContours(red_mask, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_NONE)
#contour_i = red_mask.copy()
contour_i = cv2.cvtColor(contour_i, cv2.COLOR_GRAY2BGR)
cv2.drawContours(contour_i, contour_contours, -1, (0, 0, 255), 2)
# show red mask
frame = 600
while frame > 0:
  print(frame)
  contour_i = cv2.flip(contour_i,0)
  out.write(contour_i)
  cv2.imshow("red_mask", red_mask)
  cv2.imshow("test", contour_i)
  #cv2.imwrite("red_mask.jpg", red_mask)
  #cv2.imwrite("test.jpg", contour_i)
  cv2.waitKey(1)
  frame -= 1

out.release()
```



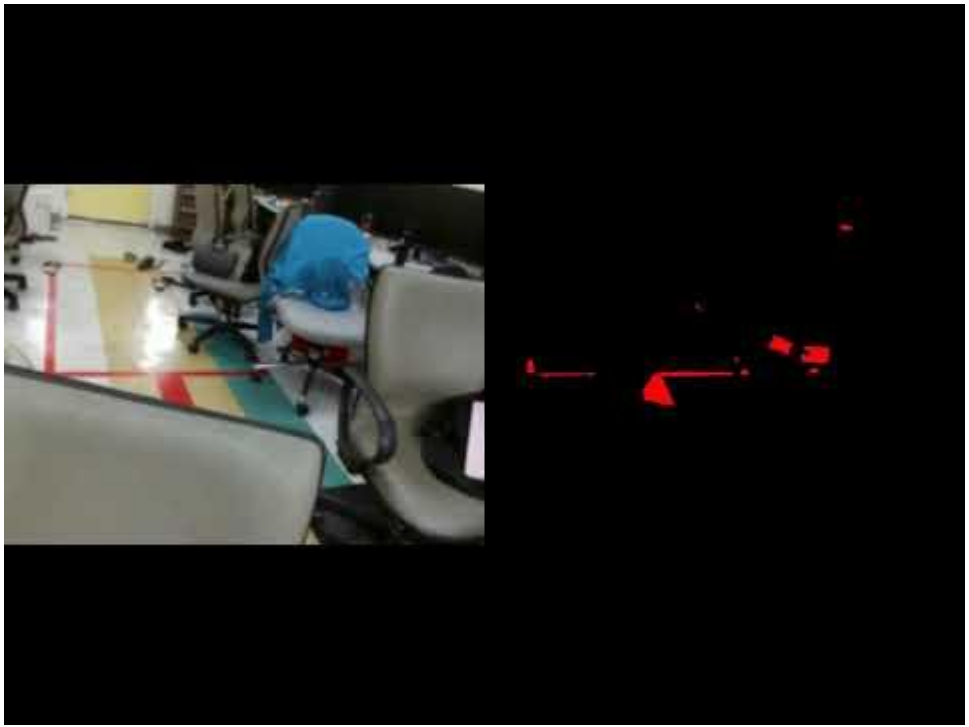


http://wiki.ros.org/vision_opencv

Using openCV in Ros - contours

```
def main():

    fourcc = cv2.VideoWriter_fourcc('X','V','I','D')
    out = cv2.VideoWriter('test_contour.avi', fourcc, 30.0, (1920, 720))
    rospy.init_node('h264_listener')
    rospy.Subscriber("/tello/image_raw/h264", H264Packet, callback)
    pub = rospy.Publisher('/selfDefined', test, queue_size = 1)
    container = av.open(stream)
    rospy.loginfo('main: opened')
    frame_skip = 300
    for frame in container.decode(video=0):
        if 0 < frame_skip:
          frame_skip -= 1
          continue
        start_time = time.time()
        image = cv2.cvtColor(np.array(frame.to_image()), cv2.COLOR_RGB2BGR)
        blurred_img = cv2.GaussianBlur(image, (13, 13), 0)
        hsv_img = cv2.cvtColor(blurred_img.copy(), cv2.COLOR_BGR2HSV)
        red_mask = findMask(hsv_img)
        (c_i, c_c, c_h) = cv2.findContours(red_mask, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_NONE)
        show_image = cv2.cvtColor(c_i, cv2.COLOR_GRAY2BGR)
        cv2.drawContours(show_image, c_c, -1, (0,0,255), -1)

        #pub.publish(test([center[0],center[1]]))
        out.write(np.concatenate((blurred_img, show_image), axis=1))
        cv2.imshow('result', np.concatenate((blurred_img, show_image), axis=1))
        cv2.waitKey(1)
        if frame.time_base < 1.0/60:
          time_base = 1.0/60
        else:
          time_base = frame.time_base
        frame_skip = int((time.time() - start_time)/time_base)
```

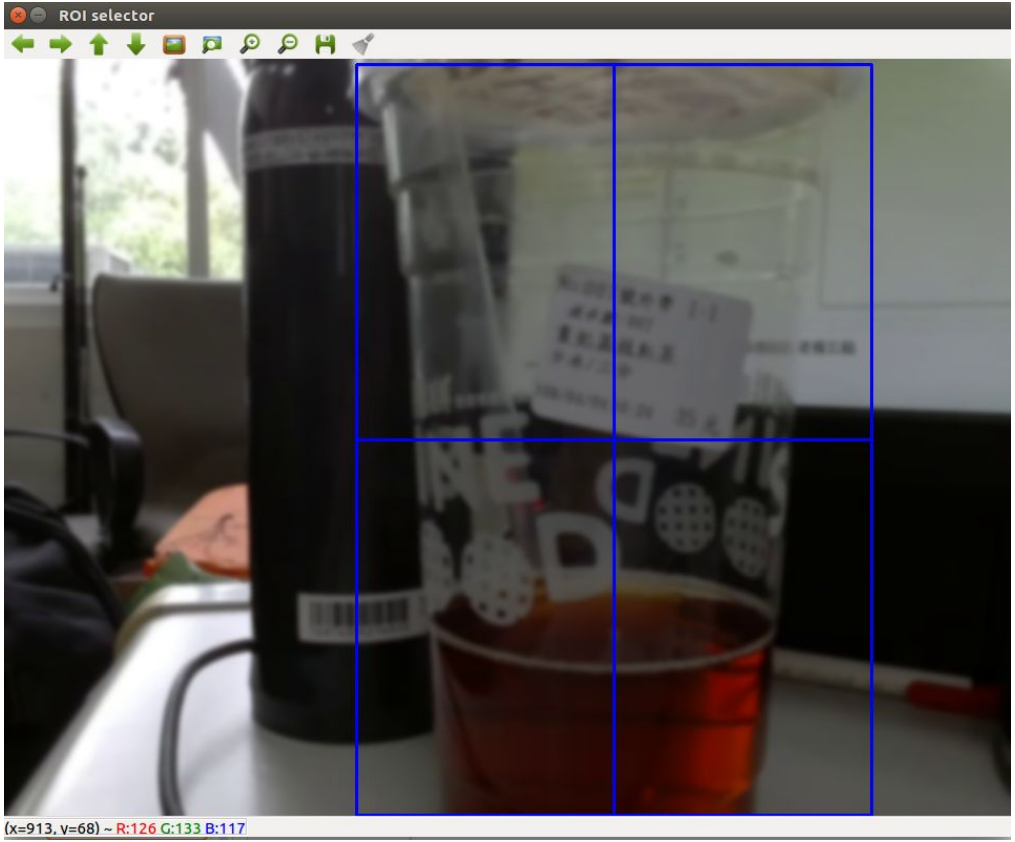
test



ROI(Region of Interst)

cv2.selectROI(image)
return tuple
1. Left corner point
2. width, height

Only select the region, but UAV is moving…
Need tracker to track interst thing



Tracker

tracker: Boosting CSRT GOTURN KCF TLD ...

```
tracker = cv2.TrackerTLD_create()
tracker.init(image, bbox)
retval, bbox = tracker.update(image)
```

```
p1 = (int(bbox[0]), int(bbox[1]))
p2 = (int(bbox[0] + bbox[2]), int(bbox[1] + bbox[3]))
center = (int((p2[0]+p1[0])/2),int((p2[1]+p1[1])/2))

show_image = blurred_img.copy()
cv2.rectangle(show_image, p1, p2, (0,0,255), 2, 1)
cv2.circle(show_image, center,3,(0,0,255),-1)
```

```
pub = rospy.Publisher('/selfDefined', test, queue_size = 1)
pub.publish(test([center[0],center[1]]))
```

```
kslab@kslab-ESC500-G4:~$ rosmsg show tello_driver/test
int32[2] l1
```

https://docs.opencv.org/3.4/d9/df8/group__tracking.html

test

Using openCV in Ros - control

self_pub = rospy.Subscriber('/selfDefined', test, callback)
cmd_pub = rospy.Publisher('/tello/cmd_vel', Twist, queue_size = 10)

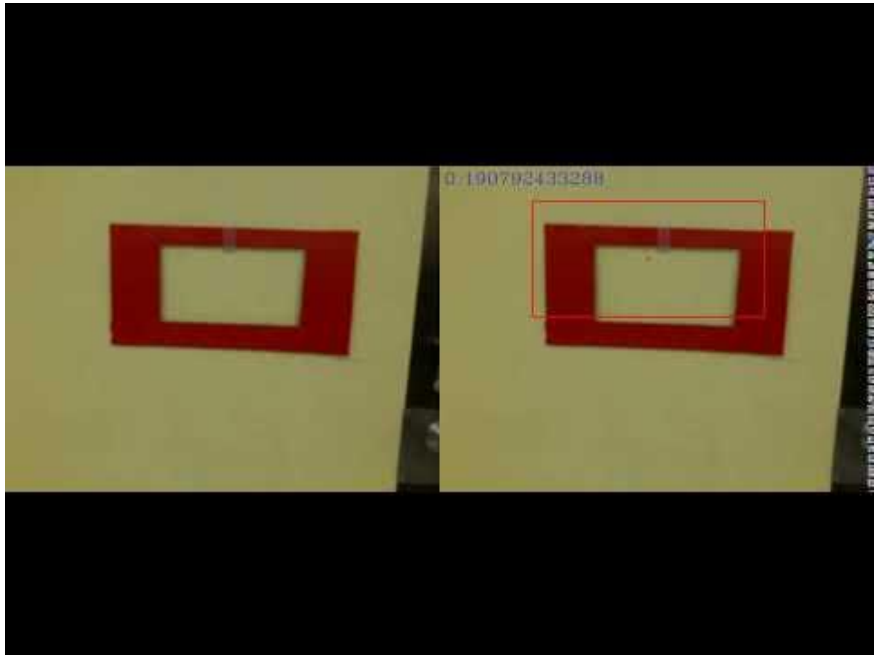
vision_center = (480, 180)
compare ROI_center and vision_center to get UAV move direction and then move forward

```
dx = target[0] - center[0]
dy = target[1] - center[1]

msg.linear.y = -dx / abs(dx) * 0.1
msg.linear.z = -dy / abs(dy) * 0.2
```

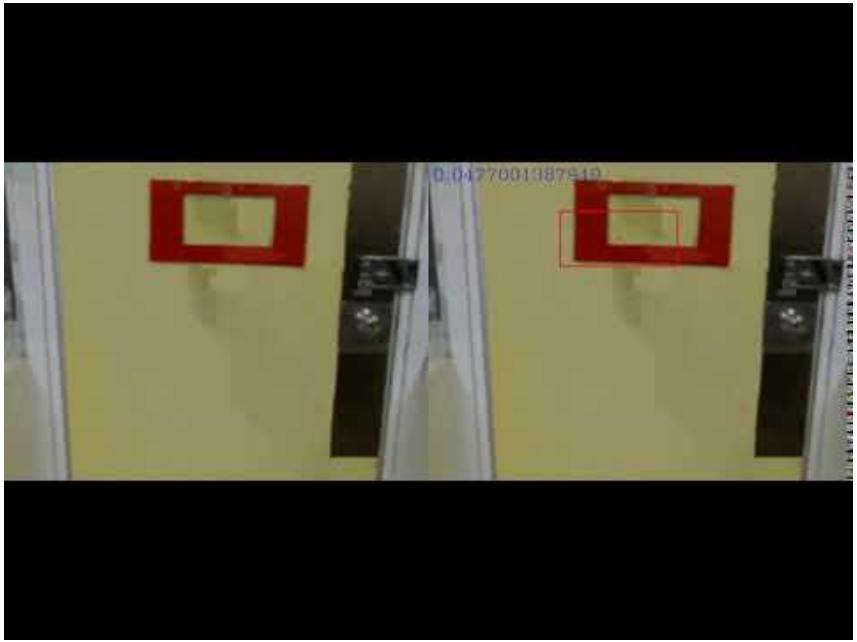
Using openCV in Ros - control

```
if check == False:
  if abs(dx) < 48 and abs(dy) < 48:
    check = True
  else:
    check = False
else:
  if abs(dx) >= 50 or abs(dy) >= 50:
    check = False

if check == True or (dx == 0 or dy == 0):
  msg = Twist()
  msg.linear.x = 0.2
  cmd_pub.publish(msg)
  rate.sleep()
else:
  msg = Twist()
  msg.linear.y = -dx / abs(dx) * 0.1
  msg.linear.z = -dy / abs(dy) * 0.2

  cmd_pub.publish(msg)
  rate.sleep()
  #sleep(1)
```

DEMO1



DEMO2



DEMO3

DEMO4