國立清華大學
National Tsing Hua University

INNOVATION ENGINEERING LAB

IIE Final Project

Object detection in insole recognition using faster R-CNN and Hough transform

Presenter：蔡和諺

目標

- 應用物件偵測(object detection)的技術來辨識鞋墊工廠中射出成型的鞋墊位置，進而利用機械手臂夾取，達成自動化生產的目的。此外，由於物件偵測無法標出被偵測物的方向，因此我們利用霍夫轉換找出鞋墊上的直線特徵，接著透過法向量求取鞋墊擺放的大致方向。

5W1H

What	Why	Who	When	Where
建立一個自動化系統	降地人力成本	公司管理者	想提升企業競爭力時	生產工廠

How

結合深度學習與電腦視覺，達到自動偵測物體的目的

方法

- 物件偵測Object detection

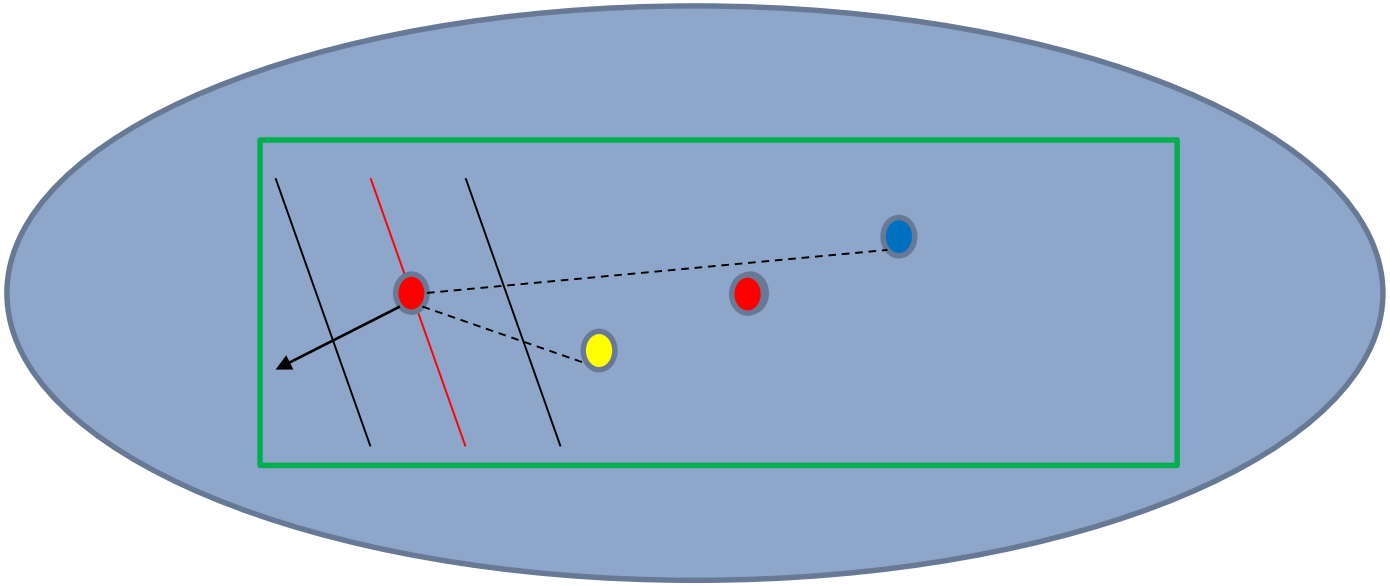
  透過訓練資料集，在新的資料中辨識並定位多個物體。

- 霍夫轉換Hough transform

  透過將圖像進行轉換，找出圖像中可能存在直線的位置。

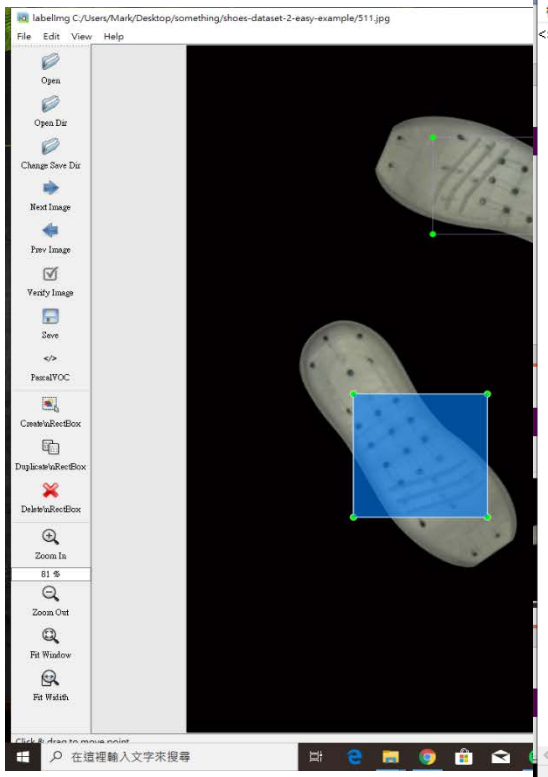
方法

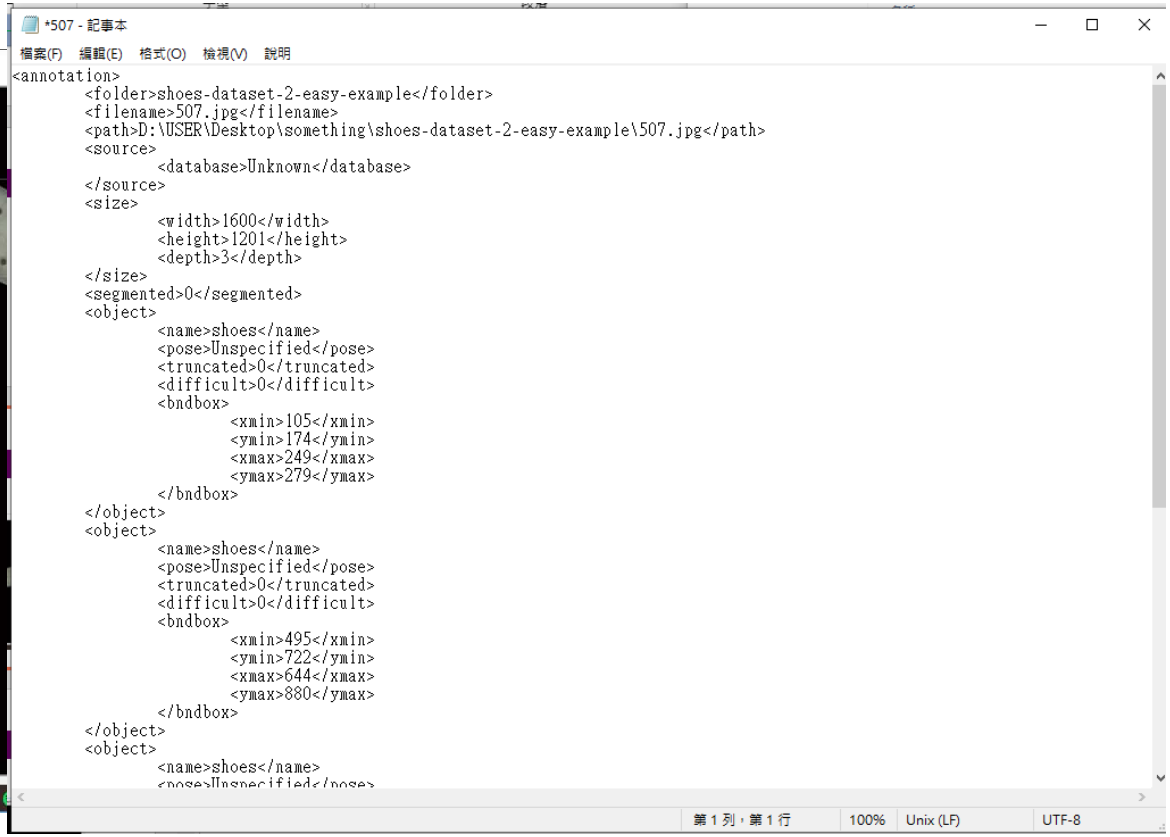
- 概念圖



方法

Label data (labelImg → XML)

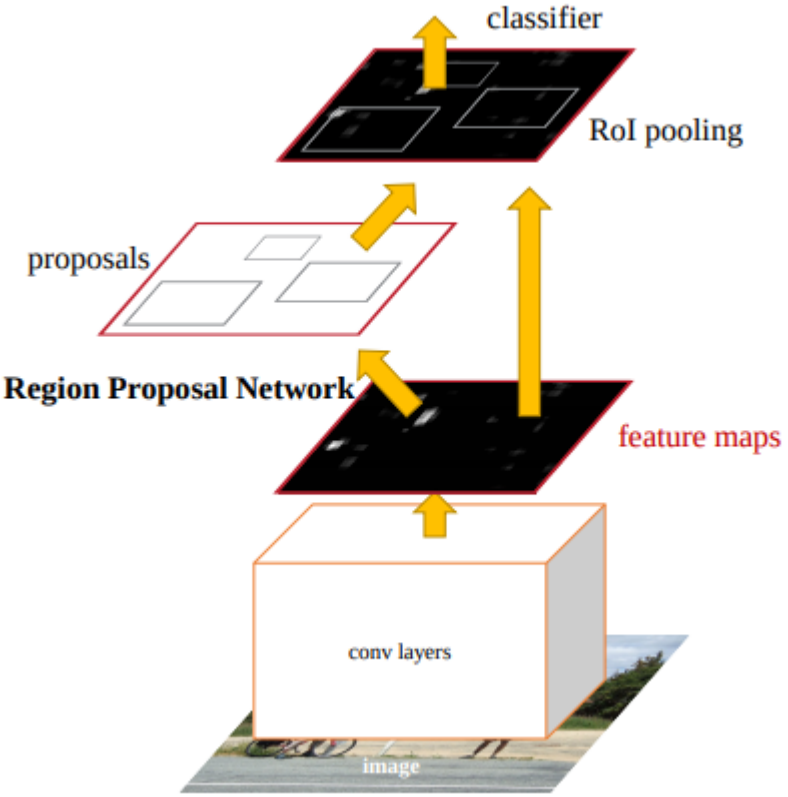




方法

Faster RCNN

- Convolutional layers
- Region proposal networks (RPN)
- ROI pooling
- Classification



方法

Inception-v2 (GooLeNet)

type	patch size/stride or remarks	input size
conv	3×3/2	299×299×3
conv	3×3/1	149×149×32
conv padded	3×3/1	147×147×32
pool	3×3/2	147×147×64
conv	3×3/1	73×73×64
conv	3×3/2	71×71×80
conv	3×3/1	35×35×192
3×Inception	As in figure 5	35×35×288
5×Inception	As in figure 6	17×17×768
2×Inception	As in figure 7	8×8×1280
pool	8 × 8	8 × 8 × 2048
linear	logits	1 × 1 × 2048
softmax	classifier	1 × 1 × 1000

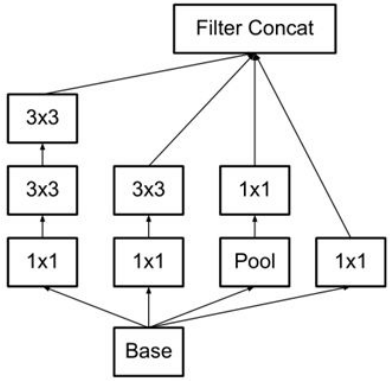


figure5

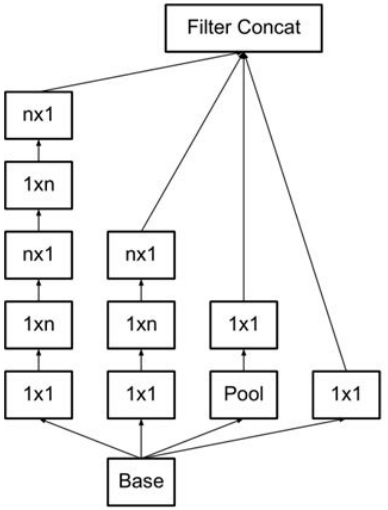


figure6

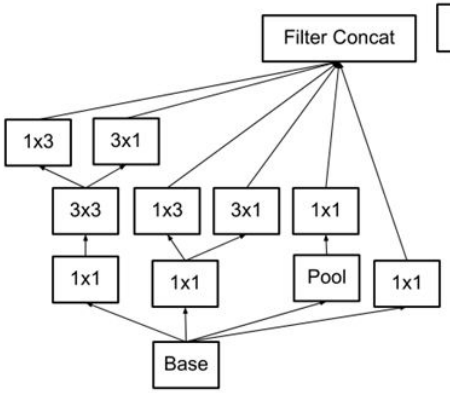


figure7

國立清華大學
National Tsing Hua University

INNOVATION ENGINEERING LAB

方法

Training data (faster_rcnn_inception_v2_pets.config)

```
train_config: {
  batch_size: 1
  optimizer {
    momentum_optimizer: {
      learning_rate: {
        manual_step_learning_rate {
          initial_learning_rate: 0.0002
          schedule {
            step: 900000
            learning_rate: .00002
          }
          schedule {
            step: 1200000
            learning_rate: .000002
          }
        }
      }
      momentum_optimizer_value: 0.9
    }
    use_moving_average: false
  }
```

Anaconda Prompt - python train.py --logtostderr --train_dir=training/ --pipeline_config_path=training/faster_rcnn_ince

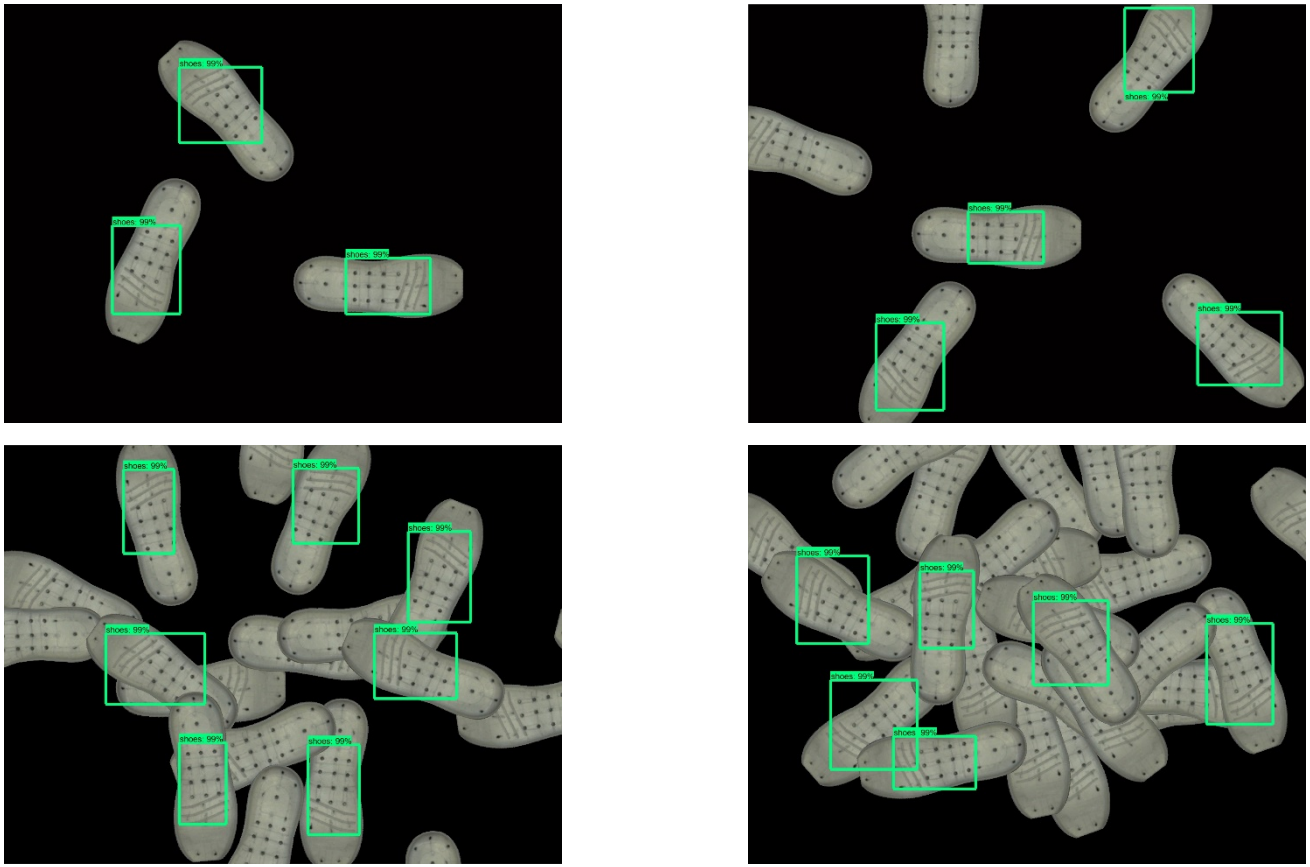
```
INFO:tensorflow:global step 6589: loss = 0.0682 (9.994 sec/step)
INFO:tensorflow:global step 6590: loss = 0.0265 (9.792 sec/step)
INFO:tensorflow:global step 6590: loss = 0.0265 (9.792 sec/step)
INFO:tensorflow:global step 6591: loss = 0.0222 (9.872 sec/step)
INFO:tensorflow:global step 6591: loss = 0.0222 (9.872 sec/step)
INFO:tensorflow:global step 6592: loss = 0.0484 (11.761 sec/step)
INFO:tensorflow:global step 6592: loss = 0.0484 (11.761 sec/step)
INFO:tensorflow:Recording summary at step 6592.
INFO:tensorflow:Recording summary at step 6592.
INFO:tensorflow:global step 6593: loss = 0.0265 (10.354 sec/step)
INFO:tensorflow:global step 6593: loss = 0.0265 (10.354 sec/step)
INFO:tensorflow:global step 6594: loss = 0.0747 (10.225 sec/step)
INFO:tensorflow:global step 6594: loss = 0.0747 (10.225 sec/step)
INFO:tensorflow:global step 6595: loss = 0.0271 (10.104 sec/step)
INFO:tensorflow:global step 6595: loss = 0.0271 (10.104 sec/step)
INFO:tensorflow:global step 6596: loss = 0.0309 (9.904 sec/step)
INFO:tensorflow:global step 6596: loss = 0.0309 (9.904 sec/step)
INFO:tensorflow:global step 6597: loss = 0.0198 (10.085 sec/step)
INFO:tensorflow:global step 6597: loss = 0.0198 (10.085 sec/step)
INFO:tensorflow:global step 6598: loss = 0.0894 (9.851 sec/step)
INFO:tensorflow:global step 6598: loss = 0.0894 (9.851 sec/step)
INFO:tensorflow:global step 6599: loss = 0.0253 (9.987 sec/step)
INFO:tensorflow:global step 6599: loss = 0.0253 (9.987 sec/step)
INFO:tensorflow:global step 6600: loss = 0.0206 (10.188 sec/step)
INFO:tensorflow:global step 6600: loss = 0.0206 (10.188 sec/step)
INFO:tensorflow:global step 6601: loss = 0.0238 (10.186 sec/step)
INFO:tensorflow:global step 6601: loss = 0.0238 (10.186 sec/step)
INFO:tensorflow:global step 6602: loss = 0.0298 (9.916 sec/step)
INFO:tensorflow:global step 6602: loss = 0.0298 (9.916 sec/step)
```

國立清華大學
National Tsing Hua University

INNOVATION ENGINEERING LAB

方法

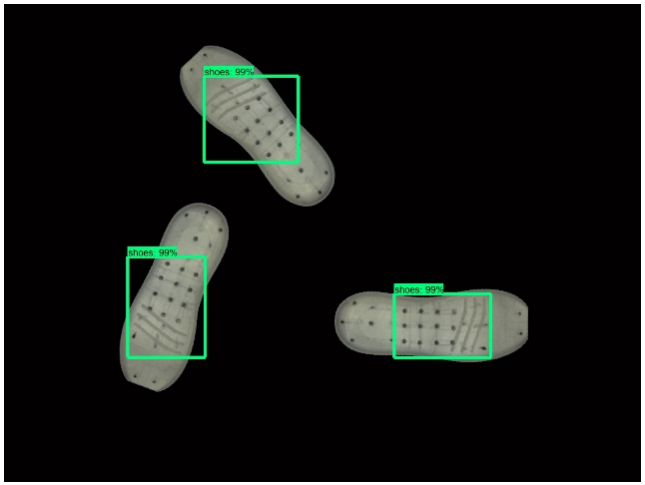
Result of object detection (Object_detection_image.py)

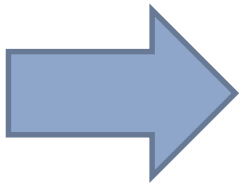


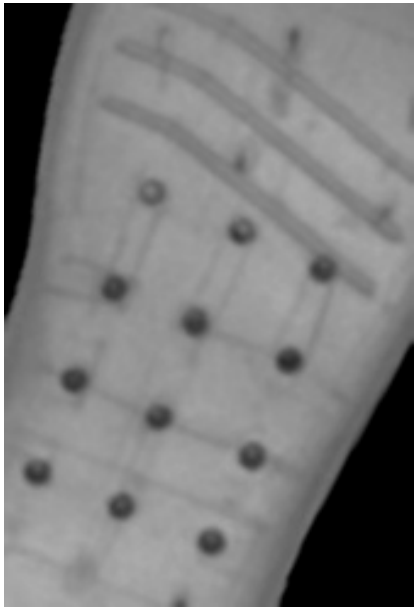
方法

Crop image

```
crop_img = image[int(round(1201*test[0]))+6:int(round(1201*test[2]))-6,
                 int(round(1600*test[1]))+6:int(round(1600*test[3]))-6]
```



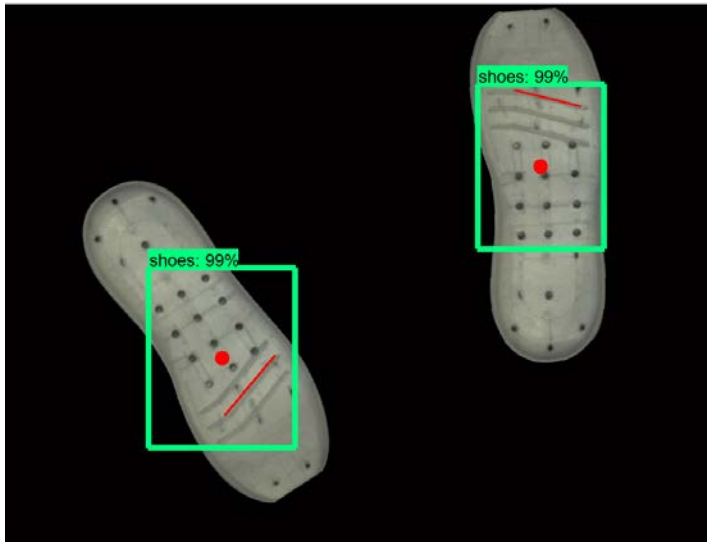


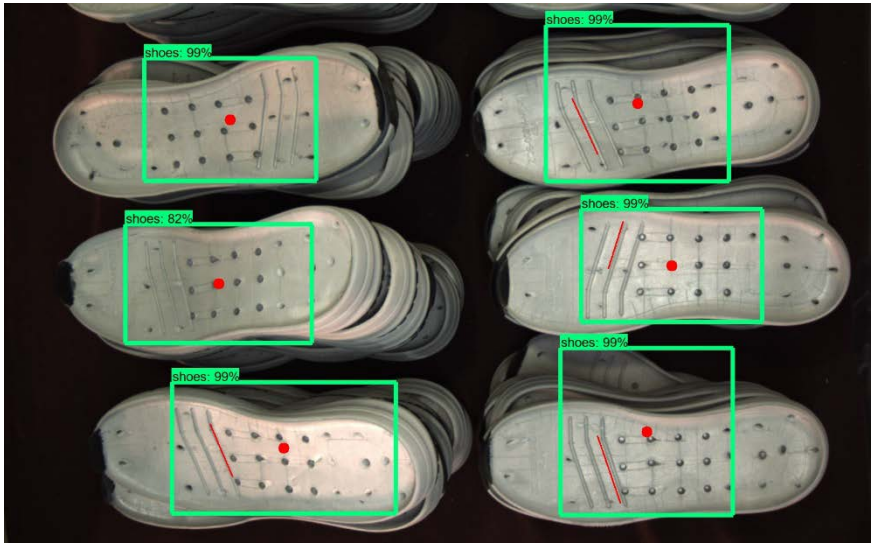


方法

Hough transform

```
minLineLength = 80
maxLineGap = 5
lines = cv2.HoughLinesP(crop_img, 2, np.pi/180, 1, np.array([]), minLineLength,maxLineGap)
# print(lines)

if lines is not None:
    for x1,y1,x2,y2 in lines[0]:
        cv2.line(image,(int(round(1600*test[1]))+x1,int(round(1200*test[0]))+y1),
                       (int(round(1600*test[1]))+x2,int(round(1200*test[0]))+y2),(0,0,255),2)
```





國立清華大學 National Tsing Hua University

INNOVATION ENGINEERING LAB

方法

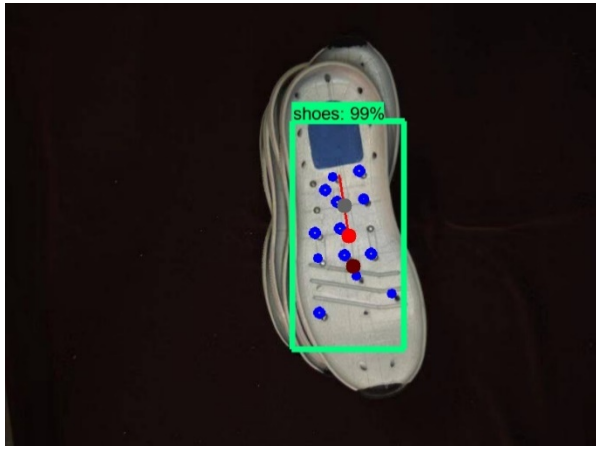
```
if lines is not None:
    for x1,y1,x2,y2 in lines[0]:
        cv2.line(image,(int(round(1600*test[1]))+x1,int(round(1200*test[0]))+y1),(int(round(1600*test[1]))+x2,int(round(1200*test[0]))+y2),(0,0,255),2)

        cv2.line(image,(int(round((1600*test[1]+1600*test[3])*0.5))-((int(round(1200*test[0]))+y2)-(int(round(1200*test[0]))+y1)),int(round((1200*test[0]+1200*test[2])*0.5))-((int(round(1600*test[1]))+x2)-(int(round(1600*test
                (int(round((1600*test[1]+1600*test[3])*0.5)),int(round((1200*test[0]+1200*test[2])*0.5))),(0,0,255),2)

        #cv2.line(crop_img, (x1,y1), (x2,y2), (0,0,255), 2)
        vector = (x2-x1,y2-y1)
mid_x = int(round(1600*test[1]+(x1+x2)*0.5))
mid_y = int(round(1200*test[0]+(y1+y2)*0.5))
a_x = int(round((1600*test[1]+1600*test[3])*0.5))-   int(round(0.1*((int(round(1200*test[0]))+y2)-(int(round(1200*test[0]))+y1))))
a_y = int(round((1200*test[0]+1200*test[2])*0.5))-   int(round(0.1*((int(round(1600*test[1]))+x2)-(int(round(1600*test[1]))+x1))))
b_x = int(round((1600*test[1]+1600*test[3])*0.5))+   int(round(0.1*((int(round(1200*test[0]))+y2)-(int(round(1200*test[0]))+y1))))
b_y = int(round((1200*test[0]+1200*test[2])*0.5))+   int(round(0.1*((int(round(1600*test[1]))+x2)-(int(round(1600*test[1]))+x1))))
mid = (mid_x, mid_y)
a = (a_x, a_y)
b = (b_x, b_y)
da = (mid_x - a_x)**2 + (mid_y - a_y)**2
db = (mid_x - b_x)**2 + (mid_y - b_y)**2
print(da, db)
if da > db:
    cv2.circle(image, (int(round((1600*test[1]+1600*test[3])*0.5))-   int(round(0.5*((int(round(1200*test[0]))+y2)-(int(round(1200*test[0]))+y1)))), int(round((1200*test[0]+1200*test[2])*0.5))-   int(round(0.5*((int(round(160
    cv2.circle(image, (int(round((1600*test[1]+1600*test[3])*0.5))+   int(round(0.5*((int(round(1200*test[0]))+y2)-(int(round(1200*test[0]))+y1)))), int(round((1200*test[0]+1200*test[2])*0.5))+   int(round(0.5*((int(round(160
else:
    cv2.circle(image, (int(round((1600*test[1]+1600*test[3])*0.5))-   int(round(0.5*((int(round(1200*test[0]))+y2)-(int(round(1200*test[0]))+y1)))), int(round((1200*test[0]+1200*test[2])*0.5))-   int(round(0.5*((int(round(160
    cv2.circle(image, (int(round((1600*test[1]+1600*test[3])*0.5))+   int(round(0.5*((int(round(1200*test[0]))+y2)-(int(round(1200*test[0]))+y1)))), int(round((1200*test[0]+1200*test[2])*0.5))+   int(round(0.5*((int(round(160
```

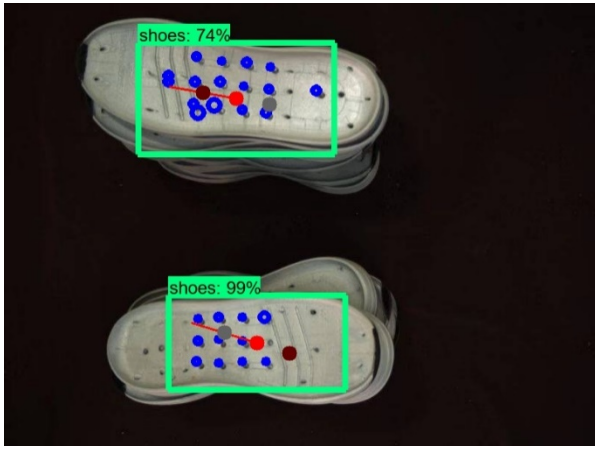
```
, int(round((1200*test[0]+1200*test[2])*0.5))-   int(round(0.5*((int(round(1600*test[1]))+x2)-(int(round(1600*test[1]))+x1))))), 10,(100,100,100),-1)
, int(round((1200*test[0]+1200*test[2])*0.5))+   int(round(0.5*((int(round(1600*test[1]))+x2)-(int(round(1600*test[1]))+x1))))), 10,(0,0,100),-1)

, int(round((1200*test[0]+1200*test[2])*0.5))-   int(round(0.5*((int(round(1600*test[1]))+x2)-(int(round(1600*test[1]))+x1))))), 10,(0,0,100),-1)
, int(round((1200*test[0]+1200*test[2])*0.5))+   int(round(0.5*((int(round(1600*test[1]))+x2)-(int(round(1600*test[1]))+x1))))), 10,(100,100,100),-1)
```

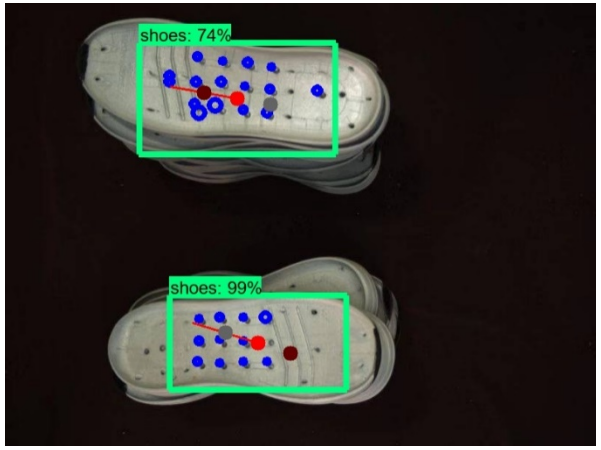
國立清華大學
National Tsing Hua University

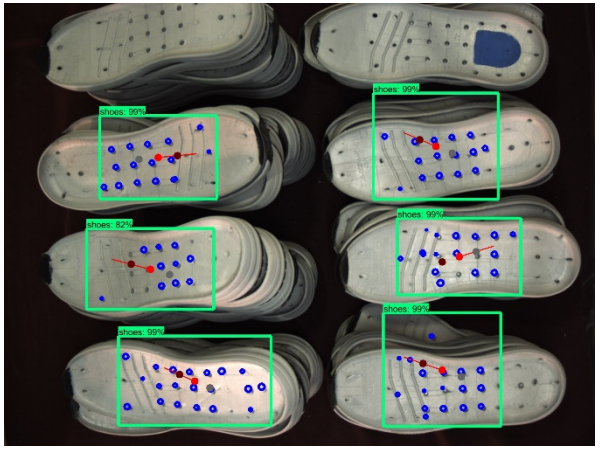
INNOVATION ENGINEERING LAB

結果









結論

這次報告的結果在Faster R-CNN的部分相當成功，由於剛開始標註圖片時，只標註了具有完整特徵的區域，因此在測試結果時未有完整特徵的鞋墊雖有可能被偵測，但其信心分數會較低，可以用此來篩選最有可能是鞋墊的位置。

在真實情況相較訓練的資料及更為複雜，因此在真實的圖片上經常會出現無法框出bounding box的問題，在尋找直線特徵上也較易受到鄰近的鞋墊干擾。整體來說，在物件偵測這方面已經達到良好的結果了，但在找出物件方向上，這次提出的方法仍然有許多進步的空間。

結論

最新的技術已經可以將標註圖片的bounding box加上旋轉角度的參數，接著在訓練時將角度的變化加入損失函數的計算，如此以來也能生成旋轉過的bounding box並回傳角度參數。

國立清華大學 National Tsing Hua University

INNOVATION ENGINEERING LAB

參考資料

文獻

Ren, S., He, K., Girshick, R., & Sun, J. (2015). Faster r-cnn: Towards real-time object detection with region proposal networks. In Advances in neural information processing systems (pp. 91-99).

網站

https://medium.com/@9821343/object-detection-api%E7%89%A9%E4%BB%B6%E8%AD%98%E5%88%A5%E5%88%86%E9%A1%9E%E5%99%A8%E5%AF%A6%E4%BD%9C-%E6%8E%A1%E7%94%A8tensorflow-cpu-%E5%9C%A8windows-10%E4%B8%8A-83d73065f27f

國立清華大學
National Tsing Hua University

INNOVATION
ENGINEERING
LAB

Thank You for Your Listening