Project #1: Build a tag inspection system for a fulfillment center

sollution:-

Approch-1

step 1 :- Frist I select this image

step 2 :- Then , i extract all purchase orders items by cv2..
Load image, grayscale, median blur, sharpen image
Threshold and morph close

#CODE

```
import numpy as np
```

```
import os, json, cv2, random
#from google.colab.patches import cv2_imshow

#!pip3 install opencv-python
# Install pytorch
# Install easyocr
#!pip install easyocr

import easyocr
import cv2
from matplotlib import pyplot as plt
import numpy as np

# Load image, grayscale, median blur, sharpen image
image = cv2.imread('/home/ravipartab/Downloads/photo_gauge/Imapct
Fulfillment/20220502_130423.jpg')


gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
blur = cv2.medianBlur(gray, 5)
sharpen_kernel = np.array([[-1,-1,-1], [-1,9,-1], [-1,-1,-1]])
sharpen = cv2.filter2D(blur, -1, sharpen_kernel)

# Threshold and morph close
thresh = cv2.threshold(sharpen, 160, 255, cv2.THRESH_BINARY_INV)[1]
kernel = cv2.getStructuringElement(cv2.MORPH_RECT, (3,3))
close = cv2.morphologyEx(thresh, cv2.MORPH_CLOSE, kernel,
iterations=2)
kernel1 = np.ones((8,8))
imgDial = cv2.dilate(close,kernel1,iterations=3)
imgThre = cv2.erode(imgDial,kernel1,iterations=2)
imgThre=cv2.bitwise_not(imgThre)
imgDial1 = cv2.dilate(imgThre,kernel1,iterations=5)
```

step 3 :-# Find contours and filter using threshold area,and extract all items.And save all extracted images(10 images)

#CODE

```
# Find contours and filter using threshold area
cnts = cv2.findContours(imgDial1, cv2.RETR_EXTERNAL,
cv2.CHAIN_APPROX_SIMPLE)
cnts = cnts[0] if len(cnts) == 2 else cnts[1]

min_area = 50000
max_area = 110000
image_number = 0
for c in cnts:
    area = cv2.contourArea(c)
    if area > min_area and area < max_area:
        x,y,w,h = cv2.boundingRect(c)
        if h/w < 1.5 and w/h < 1.5:
          #print(h/w)
          ROI = image[y:y+h, x:x+w]
          #cv2.imshow(ROI)
          cv2.imwrite('insp-new{}.png'.format(image_number), ROI)
          cv2.rectangle(image, (x, y), (x + w, y + h), (36,255,12), 2)
          image_number += 1
          print(image_number)
        else:
          pass
```

output:-

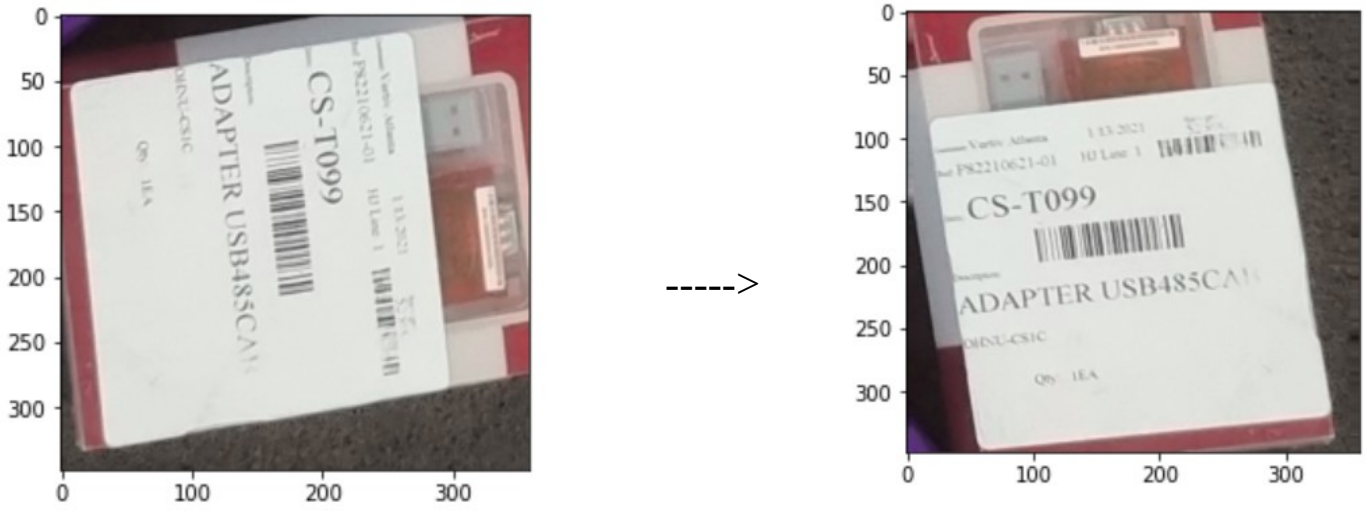
step 4:- prepare data for ocr,(rotate)

#CODE

```
IMAGE_PATH = '/home/ravipartab/Downloads/insp-new0.png'

image = cv2.imread(IMAGE_PATH)

# convert the image to grayscale and flip the foreground
# and background to ensure foreground is now "white" and
# the background is "black"
gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)



gray = cv2.bitwise_not(gray)

# threshold the image, setting all foreground pixels to
# 255 and all background pixels to 0
thresh = cv2.threshold(gray, 0, 255,cv2.THRESH_BINARY |
cv2.THRESH_OTSU)[1]

# grab the (x, y) coordinates of all pixel values that
```

```
# are greater than zero, then use these coordinates to
# compute a rotated bounding box that contains all
# coordinates
coords = np.column_stack(np.where(thresh > 0))
angle = cv2.minAreaRect(coords)[-1]
print(angle)


# rotate the image to deskew it
(h, w) = image.shape[:2]
center = (w // 2, h // 2)
M = cv2.getRotationMatrix2D(center, angle, 1.0)
rotated = cv2.warpAffine(image, M, (w, h),
flags=cv2.INTER_CUBIC, borderMode=cv2.BORDER_REPLICATE)


# draw the correction angle on the image so we can validate it
#cv2.putText(rotated, "Angle: {:.2f} degrees".format(angle),(10, 30),
cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 0, 255), 2)
# show the output image
#print("[INFO] angle: {:.3f}".format(angle))
#cv2.imshow("Input", image)
#cv2.imshow("Rotated", rotated)
#cv2.waitKey(0)
```

output

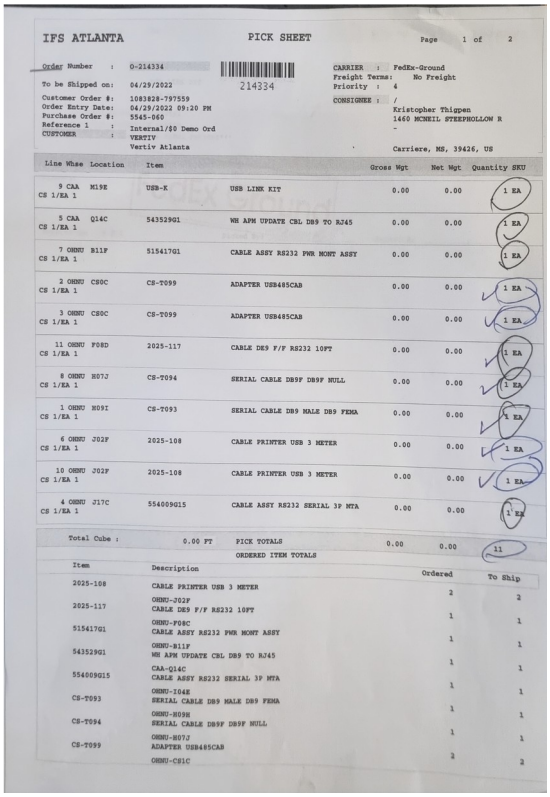


step 5—Perform ocr

```
# CODE

reader = easyocr.Reader(['en'])
result = reader.readtext(rotated)
#result



font = cv2.FONT_HERSHEY_SIMPLEX



spacer = 100
l1=[]
for detection in result:
    top_left = tuple(detection[0][0])
    bottom_right = tuple(detection[0][2])
    text = detection[1]
    print(text)

    l1.append(text)

    rotated = cv2.rectangle(rotated,(int(top_left[0]),int(top_left[1])),
(int(bottom_right[0]),int(bottom_right[1])),(0,255,0),3)
    rotated = cv2.putText(rotated,text,(20,spacer), font, 0.5,
(0,255,0),2,cv2.LINE_AA)
    spacer+=50

plt.imshow(rotated)
plt.show()
#cv2.imwrite('ocr_img0.png', img)
```

Step 6:-Extact item number in list

Step 7:- Make list of all items from pick-sheet store in a list



IFS ATLANTA — PICK SHEET — Page 1 of 2

Order Number : O-214334 — 214334

To be Shipped on: 04/29/2022
Customer Order #: 1083828-797559
Order Entry Date: 04/29/2022 09:20 PM
Purchase Order #: 5545-060
Reference 1 : Internal/80 Demo Ord
CUSTOMER : VERTIV
Vertiv Atlanta

CARRIER : FedEx-Ground
Freight Terms: No Freight
Priority : 4
CONSIGNEE : /
Kristopher Thigpen
1460 MCNEIL STEEPHOLLOW R
-
Carriere, MS, 39426, US

Line	Whse	Location	Item		Gross Wgt	Net Wgt	Quantity	SKU
9	CAA	M19E	USB-K	USB LINK KIT	0.00	0.00	1 EA	
CS 1/EA 1								
5	CAA	Q14C	543529G1	WH APM UPDATE CBL DB9 TO RJ45	0.00	0.00	1 EA	
CS 1/EA 1								
7	OHNU	B11F	515417G1	CABLE ASSY RS232 PWR MONT ASSY	0.00	0.00	1 EA	
CS 1/EA 1								
2	OHNU	C80C	CS-T099	ADAPTER USB485CAB	0.00	0.00	1 EA	
CS 1/EA 1								
3	OHNU	C80C	CS-T099	ADAPTER USB485CAB	0.00	0.00	1 EA	
CS 1/EA 1								
11	OHNU	F08D	2025-117	CABLE DE9 F/F RS232 10FT	0.00	0.00	1 EA	
CS 1/EA 1								
8	OHNU	H07J	CS-T094	SERIAL CABLE DB9F DB9F NULL	0.00	0.00	1 EA	
CS 1/EA 1								
1	OHNU	H09I	CS-T093	SERIAL CABLE DB9 MALE DB9 FEMA	0.00	0.00	1 EA	
CS 1/EA 1								
6	OHNU	J02F	2025-108	CABLE PRINTER USB 3 METER	0.00	0.00	1 EA	
CS 1/EA 1								
10	OHNU	J02F	2025-108	CABLE PRINTER USB 3 METER	0.00	0.00	1 EA	
CS 1/EA 1								
4	OHNU	J17C	554009G15	CABLE ASSY RS232 SERIAL 3P MTA	0.00	0.00	1 EA	
CS 1/EA 1								
Total Cube :			0.00 FT	PICK TOTALS	0.00	0.00	11	

ORDERED ITEM TOTALS

Item	Description	Ordered	To Ship
2025-108	CABLE PRINTER USB 3 METER	2	2
2025-117	OHNU-J02F CABLE DE9 F/F RS232 10FT	1	1
515417G1	OHNU-F08C CABLE ASSY RS232 PWR MONT ASSY	1	1
543529G1	OHNU-B11F WH APM UPDATE CBL DB9 TO RJ45	1	1
554009G15	CAA-Q14C CABLE ASSY RS232 SERIAL 3P MTA	1	1
CS-T093	OHNU-I04E SERIAL CABLE DB9 MALE DB9 FEMA	1	1
CS-T094	OHNU-H09H SERIAL CABLE DB9F DB9F NULL	1	1
CS-T099	OHNU-H07J ADAPTER USB485CAB	3	3
	OHNU-C81C		

Step 8:- validate items in pick sheet and extracted items

```
#CODE

#From pick sheet
l2=['USB-k','543529G1','515417G1','CS-T099','CS-T099','2025-117','CS-T094','CS-T093','2025-108','2025-108','554009G15']

l3 = []
for i in la:
    for l in i:
        l3.append(l)

result = []
for element in l2:
    if element in l3:
        result.append(element)

#result
#len(result)

if image_number == len(result):
    print('all parts are as per the sheet, ')
else:
    print('all parts are  NOT as per the sheet, Please validate again')
```

approch 2 :-

Step 1 to 3 same as approch 1

Step 4:- Tried with pyzbar to read barcode from image

Code

```
#!pip install pyzbar

from matplotlib import pyplot as plt

# Importing library
import cv2
from pyzbar.pyzbar import decode

# Make one method to decode the barcode
def BarcodeReader(image):

    # read the image in numpy array using cv2
    img = cv2.imread(image)

    # Decode the barcode image
    detectedBarcodes = decode(img)

    # If not detected then print the message
    if not detectedBarcodes:
        print("Barcode Not Detected or your barcode is blank/corrupted!")
    else:

          # Traverse through all the detected barcodes in image
        for barcode in detectedBarcodes:
```

```
        # Locate the barcode position in image
        (x, y, w, h) = barcode.rect

        # Put the rectangle in image using
        # cv2 to heighlight the barcode
        cv2.rectangle(img, (x-10, y-10),
                      (x + w+10, y + h+10),
                      (255, 0, 0), 2)

        if barcode.data!="":

        # Print the barcode data
            print('data',barcode.data)
            #print('type',barcode.type)

            return(barcode.data)



    plt.imshow(img)
    plt.show()
    #Display the image
    #cv2.imshow("Image", img)
    #cv2.waitKey(0)
    #cv2.destroyAllWindows()



image="/home/ravipartab/Downloads/photo_gauge/Imapct
Fulfillment/20220502_130050.jpg"

a=BarcodeReader(image)
```

```
import codecs

print(type(a)) # <class 'bytes'>
strData = codecs.decode(a, 'UTF-8')

strData
```

Step5:-validate items in pick sheet and extracted items

Project #2: Build an optical system for measuring length of pipes

sollution:-

Approch-1

step 1 :- Annotate data through labelme tool

step 2:- Tried Image Stitching with OpenCV,so that i can calculate total lenth of pipe in a LOT.

##CODE

```
import numpy as np
import pandas as pd
import cv2
from IPython.display import Image
import os
from google.colab.patches import cv2_imshow
import uuid   # Unique identifier


main_folder = '/content/drive/MyDrive/x'
my_folders = os.listdir(main_folder)
```

```
print(my_folders)

for folder in my_folders:
  path = main_folder +'/'+folder
  images=[]
  myList=os.listdir(path)
  print(f'total no of image detected {len(myList)}')
  for imgN in myList:
    curImg=cv2.imread(f'{path}/{imgN}')
    images.append(curImg)
  #print(len(images))
  stitcher = cv2.Stitcher.create()
  (status,result) = stitcher.stitch(images)
  if (status == cv2.STITCHER_OK):
    print('Panorama Generated')
    cv2_imshow(result)



    IMAGES_PATH='/content/drive/MyDrive/panorama'
    # Naming out image path
    imgname = os.path.join(IMAGES_PATH,folder+str(uuid.uuid1())+'.png')

    # Writes out image to file
    cv2.imwrite(imgname, result)
    print(imgname)
    print('Panorama saved')

    cv2.waitKey(1)
  else:
    print('panorama generation unsuccessful')

cv2.waitKey(0)
```

step 3:- Tried Image Stitching with OpenCV,so that i can calculate total lenth of pipe in a LOT.

Step 4: train detectron2

```
]: import torch, detectron2
   !nvcc --version
   TORCH_VERSION = ".".join(torch.__version__.split(".")[:2])
   CUDA_VERSION = torch.__version__.split("+")[-1]
   print("torch: ", TORCH_VERSION, "; cuda: ", CUDA_VERSION)
   print("detectron2:", detectron2.__version__)
```

```
nvcc: NVIDIA (R) Cuda compiler driver
Copyright (c) 2005-2020 NVIDIA Corporation
Built on Mon_Oct_12_20:09:46_PDT_2020
Cuda compilation tools, release 11.1, V11.1.105
Build cuda_11.1.TC455_06.29190527_0
torch:  1.12 ; cuda:  cu113
detectron2: 0.6
```

```
]: import detectron2
   from detectron2.utils.logger import setup_logger
   setup_logger()
   import numpy as np
   import os, json, cv2, random
   from google.colab.patches import cv2_imshow
   from detectron2 import model_zoo
   from detectron2.engine import DefaultPredictor
   from detectron2.config import get_cfg
   from detectron2.utils.visualizer import Visualizer
   from detectron2.data import MetadataCatalog, DatasetCatalog
```

Step : 5

prepare data

```
import detectron2
from detectron2.utils.logger import setup_logger
setup_logger()
import numpy as mp
import os, json, cv2, random
from google.colab.patches import cv2_imshow
from detectron2 import model_zoo
from detectron2.engine import DefaultPredictor
from detectron2. config import get_cfg
from detectron2.utils.visualizer import Visualizer
from detectron2.data import MetadataCatalog, DatasetCatalog
```

```
from detectron2.data. datasets import register_coco_instances
register_coco_instances("customtrain", {}, "/content/drive/MyDrive/data_detectron/json/via_region_data.json", "/con
```

```
sample_metadata = MetadataCatalog.get("customtrain")
dataset_dicts = DatasetCatalog.get( "customtrain")
```

```
WARNING [07/18 05:27:10 d2.data.datasets.coco]:
Category ids in annotations are not in [1, #categories]! We'll apply a mapping for you.

[07/18 05:27:10 d2.data.datasets.coco]: Loaded 1 images in COCO format from /content/drive/MyDrive/data_detectron/
json/via_region_data.json
WARNING [07/18 05:27:10 d2.data.datasets.coco]: Filtered out 3 instances without valid segmentation. There might b
e issues in your dataset generation process.  Please check https://detectron2.readthedocs.io/en/latest/tutorials/d
```

step 6: predict

```
from detectron2. engine import DefaultTrainer
cfg = get_cfg()
cfg.merge_from_file(model_zoo.get_config_file("COCO-InstanceSegmentation/mask_rcnn_R_50_FPN_3x.yaml"))
cfg.DATASETS.TRAIN = ("customtrain",)
cfg.DATASETS.TEST = ()
cfg.DATALOADER.NUM_WORKERS = 2
cfg.MODEL.WEIGHTS = model_zoo.get_checkpoint_url("COCO-InstanceSegmentation/mask_rcnn_R_50_FPN_3x.yaml") #
cfg.SOLVER.IMS_PER_BATCH = 1
cfg.SOLVER.BASE_LR = 0.0025
cfg.SOLVER.MAX_ITER = 600
cfg.MODEL.ROI_HEADS.BATCH_SIZE_PER_IMAGE = 128
cfg.MODEL.ROI_HEADS.NUM_CLASSES = 14

os.makedirs(cfg.OUTPUT_DIR, exist_ok=True)
trainer = DefaultTrainer(cfg)
trainer.resume_or_load(resume=True)
trainer.train()

cfg.MODEL.WEIGHTS = os.path.join(cfg.OUTPUT_DIR, "model_final.pth")
cfg.MODEL.ROI_HEADS.SCORE_THRESH_TEST = 0.5
# set the testing threshold for this model
cfg.DATASETS.TEST = ("customtrain",)
predictor = DefaultPredictor(cfg)
```

step 7: visualize result

```
from detectron2.utils.visualizer import ColorMode

for d in random.sample(dataset_dicts, 1):
  im = cv2.imread(d["file_name"])
  outputs = predictor(im)

  v = Visualizer(im[:, :, ::-1],
                 metadata=sample_metadata,
                 scale=0.8,
                 instance_mode=ColorMode.IMAGE_BW # remove the colors of unsegmented pixels
  )
  v = v. draw_instance_predictions(outputs["instances"].to("cpu"))
  cv2_imshow(v.get_image()[:, :, ::-1])
```

A) OUTPUT OF print(instances) Instances(num_instances=4, image_height=360, image_width=640, fields=[pred_boxes, scores, pred_classes, pred_masks])

Explanation: this output says me there are 4 boxes detected.
B) OUTPUT OF print(instances.pred_boxes) Boxes(tensor([[289.3555, 17.8171, 451.1482, 347.6050], [382.5501, 14.9712, 635.7133, 231.8446], [467.1654, 66.3414, 611.7201, 226.0997], [22.4782, 3.7928, 428.1484, 254.6716]]))
Explanation: this output says me, the coordinates of the boxes detected. In particular, the first box (instances.pred_boxes[0]) has the top_left point with coordinates (x,y)=(289.3555, 17.8171), and the bottom_right point with coordinates (x,y)=(451.1482, 347.6050)
C) OUTPUT OF print(instances.pred_boxes[0]) Boxes(tensor([[289.3555, 17.8171, 451.1482, 347.6050]])) Explanation: with this command, I just print the coordinates of the first box (instances.pred_boxes[0])

CODE

```
l1=[]
for i in (outputs["instances"].pred_boxes):
  l1.append((i[2]-i[0]).cpu().numpy().item())
  #print((i[2]-i[0]).cpu().numpy())

max1 = np.max(l1)
l2=[]
for i in l1:
  #print((i/max1)*50)
  l2.append((i/max1)*50)


sum(l2)


#add length of each pipe
```

###THE END ####