LAMPIRAN

Kode program *image processing* kendaraan

```
from imutils.video import VideoStream
from imutils.video import FPS
import numpy as np
import argparse
import imutils
import time
import cv2

ap = argparse.ArgumentParser()
ap.add_argument("-p", "--prototxt", required=True,
   help="path to Caffe 'deploy' prototxt file")
ap.add_argument("-m", "--model", required=True,
   help="path to Caffe pre-trained model")
ap.add_argument("-c", "--confidence", type=float, default=0.2,
   help="minimum probability to filter weak detections")
ap.add_argument("-v", "--video_source", type=int, default=0,
   help="video source (default = 0, external usually = 1)")
args = vars(ap.parse_args())

CLASSES = ["background", "aeroplane", "bicycle", "bird", "boat",
   "bottle", "bus", "car", "cat", "chair", "cow", "diningtable",
   "dog", "horse", "motorbike", "person", "pottedplant", "sheep",
   "sofa", "train", "tvmonitor"]
COLORS = np.random.uniform(0, 255, size=(len(CLASSES), 3))
print("[INFO] loading model...")
net = cv2.dnn.readNetFromCaffe(args["prototxt"], args["model"])
print("[INFO] starting video stream...")
```

```
vs = VideoStream(src=args["video_source"]).start()
time.sleep(2.0)
fps = FPS().start()

while True:
   frame = vs.read()
   frame = imutils.resize(frame, width=400)
   (h, w) = frame.shape[:2]
   blob = cv2.dnn.blobFromImage(cv2.resize(frame, (300, 300)),  0.007843, (300, 300),
127.5)
   net.setInput(blob)
   detections = net.forward()
   for i in np.arange(0, detections.shape[2]):
      confidence = detections[0, 0, i, 2]
      if confidence > args["confidence"]:
         idx = int(detections[0, 0, i, 1])
         box = detections[0, 0, i, 3:7] * np.array([w, h, w, h])
         (startX, startY, endX, endY) = box.astype("int")
         label = "{}: {:.2f}%".format(CLASSES[idx],
            confidence * 100)
         cv2.rectangle(frame, (startX, startY), (endX, endY),
            COLORS[idx], 2)
         y = startY - 15 if startY - 15 > 15 else startY + 15
         cv2.putText(frame, label, (startX, y),
            cv2.FONT_HERSHEY_SIMPLEX, 0.5, COLORS[idx], 2)

   cv2.imshow("Frame", frame)
   key = cv2.waitKey(1) & 0xFF
   if key == ord("q"):
```

```
        break
    fps.update()
fps.stop()
print("[INFO] elapsed time: {:.2f}".format(fps.elapsed()))
print("[INFO] approx. FPS: {:.2f}".format(fps.fps()))
cv2.destroyAllWindows()
vs.stop()
```

Kode program pembacaan plat kendaraan

```
img = cv2.imread('D://plat1.jpeg',cv2.IMREAD_COLOR)

img = cv2.resize(img, (620,480) )

gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)

destination_image = cv2.bilateralFilter(source_image, diameter of pixel, sigmaColor, sigmaSpace)

gray = cv2.bilateralFilter(gray, 13, 15, 15)

edged = cv2.Canny(gray, 30, 200)

contours = cv2.findContours(edged.copy(), cv2.RETR_TREE, cv2.CHAIN_APPROX_SIMPLE)

contours = imutils.grab_contours(contours)

contours = sorted(contours, key = cv2.contourArea, reverse = True)[:10]

for c in contours:

   peri = cv2.arcLength(c, True)

   approx = cv2.approxPolyDP(c, 0.018 * peri, True)

   if len(approx) == 4:

      screenCnt = approx

      break

if screenCnt is None:

   detected = 0

   print ("No contour detected")

else:

    detected = 1

if detected == 1:

   cv2.drawContours(img, [screenCnt], -1, (0, 0, 255), 3)

mask = np.zeros(gray.shape,np.uint8)

new_image = cv2.drawContours(mask,[screenCnt],0,255,-1,)

new_image = cv2.bitwise_and(img,img,mask=mask)

(x, y) = np.where(mask == 255)
```

```
(topx, topy) = (np.min(x), np.min(y))
(bottomx, bottomy) = (np.max(x), np.max(y))
Cropped = gray[topx:bottomx+1, topy:bottomy+1]
text = pytesseract.image_to_string(Cropped, config='--psm 11')
cv2.imshow("Frame", frame)
key = cv2.waitKey(1) & 0xFF
   if key == ord("q"):
      break
cv2.destroyAllWindows()
```