

Dokumentasi Penambahan OCR pada Deteksi Plat Nomor

Dokumentasi sebelumnya adalah melakukan training model untuk dapat mengenali plat nomor dan menambahkan bounding box, dokumentasi ini akan menjelaskan bagaimana memindahkan model yang telah dilatih di Google Colab ke PC local, convert model darknet YOLOv4 menjadi tensorflow, menjalankan deteksi plat nomor di PC local, kemudian menambahkan OCR untuk mengenali teks pada bounding box plat nomor.

Daftar Proses	Hal
1. Clone Github Repository Sumber	2
2. Install dependensi yang dibutuhkan	2
3. Download File Weight	3
4. Convert File Weight Menjadi Tensorflow	5
5. Tes Model yang telah Di-convert	6
6. Memperbaiki Error Pada Deteksi Plat Nomor	7
7. Tes Deteksi Dengan Kamera	9
8. Install TesseractOCR untuk Deteksi Karakter	9
9. Tes TesseractOCR pada Sembarang Teks	11
10. Penambahan Tesseract OCR pada Deteksi Plat Nomor	12
11. Kendala yang Dialami pada Segmentasi Karakter	16
12. Alternatif OCR dengan EasyOCR	16
13. Tes Deteksi Plat Nomor dan OCR dengan EasyOCR	19
14. Setting Parameter EasyOCR untuk Segmentasi Karakter	22
15. Menambahkan Parameter batch_size pada EasyOCR	27
16. Memisahkan Nomor TKNB dan Masa Berlaku	27
17. Percobaan Pada Beberapa Gambar Plat Nomor	29
18. Percobaan Beberapa Plat Nomor Melalui Kamera	34
19. Kesimpulan	38

Sumber:

<https://www.youtube.com/watch?v=AAPZLK41rek&t=916s>

<https://github.com/theAIGuysCode/yolov4-custom-functions>

<https://betterprogramming.pub/ocr-in-few-lines-of-code-using-easyocr-24a960b9eca1>

https://github.com/OpenBCI/OpenBCI_Experiment/issues/2

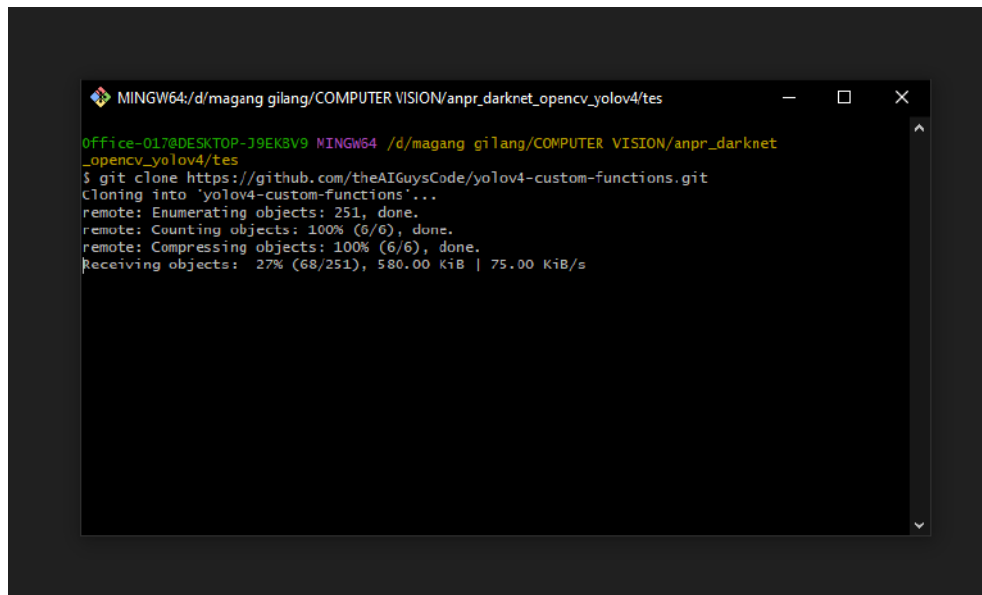
<https://github.com/hunglc007/tensorflow-yolov4-tflite/issues/368>

Syntax-syntax yang perlu dijalankan dapat ditemukan pada link berikut:

<https://github.com/theAIGuysCode/yolov4-custom-functions>

1. Clone Github Repository Sumber

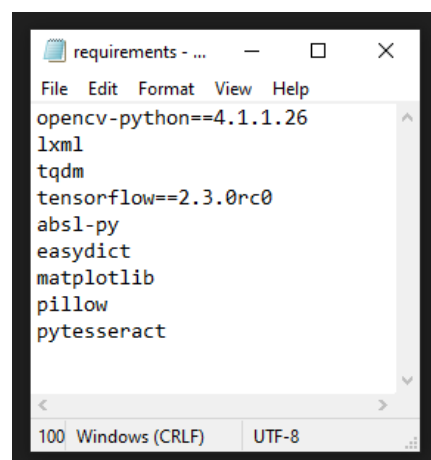
Buka Gitbash atau Powershell kemudian sesuaikan di mana file clone ingin disimpan, ketik “**git clone https://github.com/theAIGuysCode/yolov4-custom-functions.git**”.



```
MINGW64/d/magang gilang/COMPUTER VISION/anpr_darknet_opencv_yolov4/tes
office-017@DESKTOP-39EK8V9 MINGW64 /d/magang gilang/COMPUTER VISION/anpr_darknet_opencv_yolov4/tes
$ git clone https://github.com/theAIGuysCode/yolov4-custom-functions.git
Cloning into 'yolov4-custom-functions'...
remote: Enumerating objects: 251, done.
remote: Counting objects: 100% (6/6), done.
remote: Compressing objects: 100% (6/6), done.
Receiving objects: 27% (68/251), 580.00 KiB | 75.00 KiB/s
```

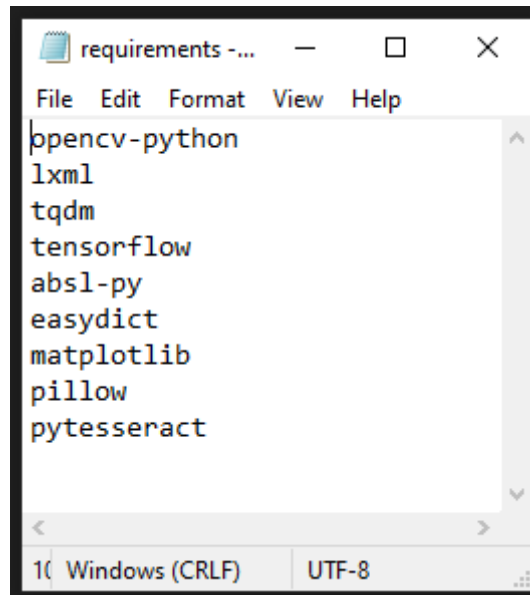
Setelah clone selesai akan ada 2 file requirements.txt, yaitu “requirements.txt” untuk install dependensi dengan CPU dan “requirements-gpu.txt” untuk GPU, dokumentasi kali ini akan menggunakan CPU. Buka file “requirements.txt” dan akan tampil tulisan seperti di bawah ini.

2. Install dependensi yang dibutuhkan



```
requirements - ...
File Edit Format View Help
opencv-python==4.1.1.26
lxml
tqdm
tensorflow==2.3.0rc0
absl-py
easydict
matplotlib
pillow
pytesseract
100 Windows (CRLF) UTF-8
```

Coba install dependensi tersebut dengan ketik “**pip install requirements.txt**” pada gitbash atau powershell, jika terjadi error karena versi opencv-python dan tensorflow yang tidak sesuai, coba hapus versi yang tertulis di file requirements.txt menjadi seperti gambar di bawah.

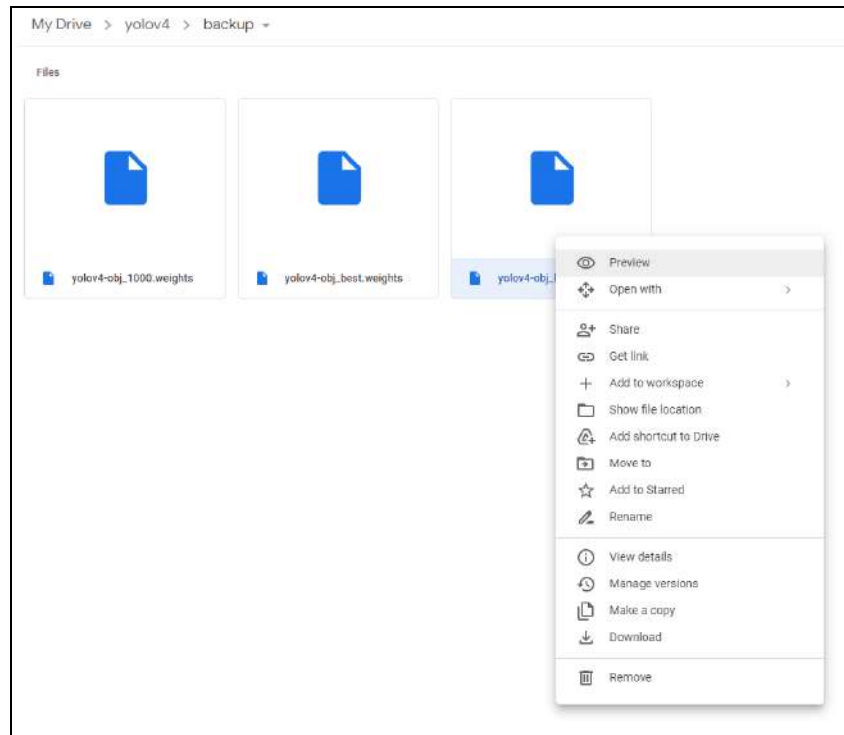


Kemudian ketik “**pip install requirements.txt**” lagi, jika error lagi ada kemungkinan beberapa library memiliki versi yang tidak saling mendukung.

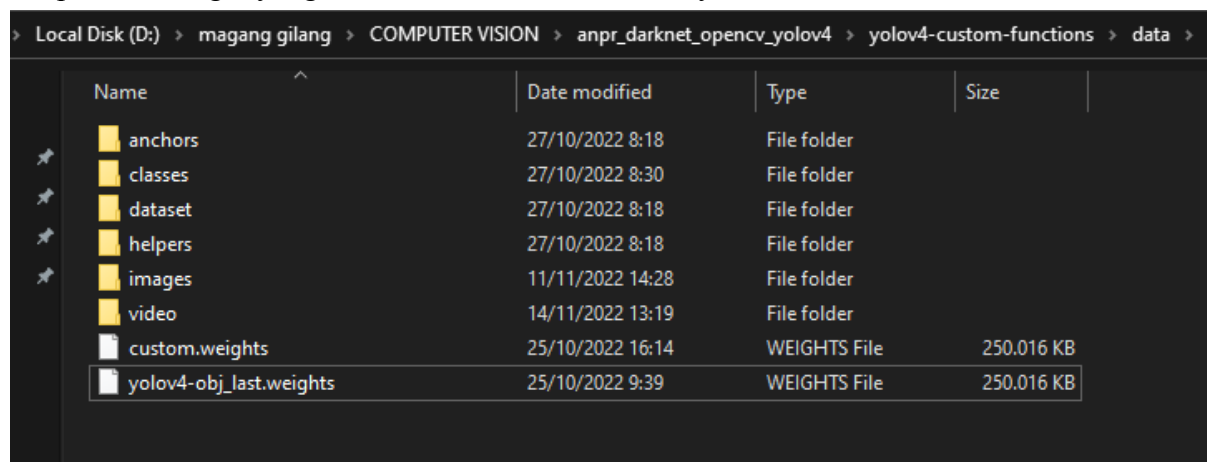
3. Download File Weight

Buka folder path “yolov4/backup” yang telah dibuat pada dokumentasi sebelumnya, download salah satu file weight dan simpan pada folder, jika belum punya file weight dapat di-download pada link:

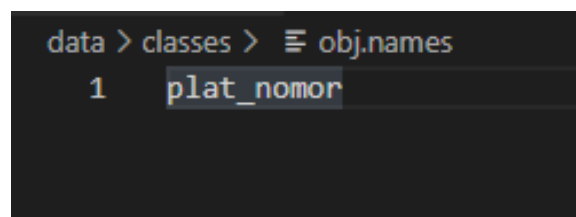
“<https://drive.google.com/file/d/1EUPtbtdF0bjRtNjGv436vDY28EN5DXDH/view>”



Simpan file weight yang telah di-download di folder “yolov4-custom-functions/data/”.



Selanjutnya buka Visual Studio Code, buat file bernama “obj.names” berisi nama label yang akan ditampilkan pada bounding box, di sini isi dari file tersebut adalah “plat_nomor”, simpan ke folder “yolov4-custom-functions/data/classes/”.



Kemudian buka file “config.py” di folder “yolov4-custom-functions/core/”, ganti baris kode dengan nama “coco.names” menjadi “obj.names”.

sebelum

```
core > config.py x
1 #!/usr/bin/env python
2 # coding=utf-8
3 from easydict import EasyDict as edict
4
5
6 __C = edict()
7 # Consumers can get config by: from config import cfg
8
9 cfg = __C
10
11 # YOLO options
12 __C.YOLO = edict()
13
14 __C.YOLO.CLASSES = "/data/classes/coco.names"
15 __C.YOLO.ANCHORS = [12,16, 19,36, 40,28, 36,75, 76,55, 72,146, 142,110, 192,243, 459,401]
16 __C.YOLO.ANCHORS_V3 = [10,13, 16,30, 33,23, 30,61, 62,45, 59,119, 116,90, 156,198, 373,326]
17 __C.YOLO.ANCHORS_TINY = [23,27, 37,58, 81,82, 81,82, 135,169, 344,319]
18 __C.YOLO.STRIDES = [8, 16, 32]
19 __C.YOLO.STRIDES_TINY = [16, 32]
20 __C.YOLO.XYSCALE = [1.2, 1.1, 1.05]
21 __C.YOLO.XYSCALE_TINY = [1.05, 1.05]
22 __C.YOLO.ANCHOR_PER_SCALE = 3
23 __C.YOLO.IOU_LOSS_THRESH = 0.5
```

sesudah

```
utils.py M config.py M x detect.py 2 detect_video.py 2 M syntax n:n anpr.txt U
pre > config.py >...
1 #!/usr/bin/env python
2 # coding=utf-8
3 from easydict import EasyDict as edict
4
5
6 __C = edict()
7 # Consumers can get config by: from config import cfg
8
9 cfg = __C
10
11 # YOLO options
12 __C.YOLO = edict()
13
14 __C.YOLO.CLASSES = "/data/classes/obj.names"
15 __C.YOLO.ANCHORS = [12,16, 19,36, 40,28, 36,75, 76,55, 72,146, 142,110, 192,243, 459,401]
16 __C.YOLO.ANCHORS_V3 = [10,13, 16,30, 33,23, 30,61, 62,45, 59,119, 116,90, 156,198, 373,326]
17 __C.YOLO.ANCHORS_TINY = [23,27, 37,58, 81,82, 81,82, 135,169, 344,319]
18 __C.YOLO.STRIDES = [8, 16, 32]
19 __C.YOLO.STRIDES_TINY = [16, 32]
20 __C.YOLO.XYSCALE = [1.2, 1.1, 1.05]
21 __C.YOLO.XYSCALE_TINY = [1.05, 1.05]
22 __C.YOLO.ANCHOR_PER_SCALE = 3
23 __C.YOLO.IOU_LOSS_THRESH = 0.5
```

4. Convert File Weight Menjadi Tensorflow

Buka powershell atau gitbash, kemudian jalankan syntax di bawah ini:

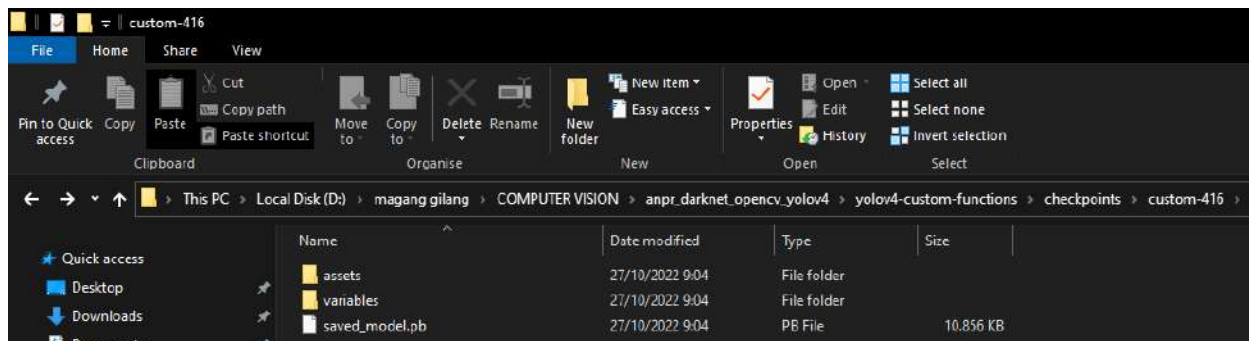
“python save_model.py --weights ./data/custom.weights --output ./checkpoints/custom-416 --input_size 416 --model yolov4”

```
MINGW64:/d/magang/gilang/COMPUTER VISION/anpr_darknet_opencv_yolov4/yolov4-custom-functions
Office-017@DESKTOP-J9EKBV9 MINGW64 /d/magang/gilang/COMPUTER VISION/anpr_darknet_opencv_yolov4/yolov4-custom-functions (master)
$ python save_model.py --weights ./data/custom.weights --output ./checkpoints/custom-416 --input_size 416 --model yolov4
```

Jika file weight berhasil di-convert menjadi tensorflow maka akan muncul seperti di bawah ini:

```
MINGW64/c/Repos/yolov4-custom-functions
tf_op_layer_strided_slice_11 (T [0]) 0 tf_op_layer_shape_11[0][0]
tf_op_layer_strided_slice_12 (T [0]) 0 tf_op_layer_shape_12[0][0]
tf_op_layer_strided_slice_15 (T [(None, None, 1)]) 0 tf_op_layer_ReaDiv_1[0][0]
tf_op_layer_strided_slice_16 (T [(None, None, 1)]) 0 tf_op_layer_ReaDiv_1[0][0]
tf_op_layer_strided_slice_17 (T [(None, None, 1)]) 0 tf_op_layer_ReaDiv_3[0][0]
tf_op_layer_strided_slice_18 (T [(None, None, 1)]) 0 tf_op_layer_ReaDiv_3[0][0]
tf_op_layer_Reshape_14/shape (T [(3,)]) 0 tf_op_layer_strided_slice_11[0][0]
tf_op_layer_strided_slice_12[0][0]
tf_op_layer_concat_17 (TensorFl [(None, None, 4)]) 0 tf_op_layer_strided_slice_15[0][0]
tf_op_layer_strided_slice_16[0][0]
tf_op_layer_strided_slice_17[0][0]
tf_op_layer_strided_slice_18[0][0]
tf_op_layer_Reshape_14 (TensorF [(None, None, None)]) 0 tf_op_layer_GatherV2_1[0][0]
tf_op_layer_Reshape_14/shape[0][0]
tf_op_layer_concat_18 (TensorFl [(None, None, None)]) 0 tf_op_layer_concat_17[0][0]
tf_op_layer_Reshape_14[0][0]
=====
Total params: 64,003,990
Trainable params: 63,937,686
Non-trainable params: 66,304
```

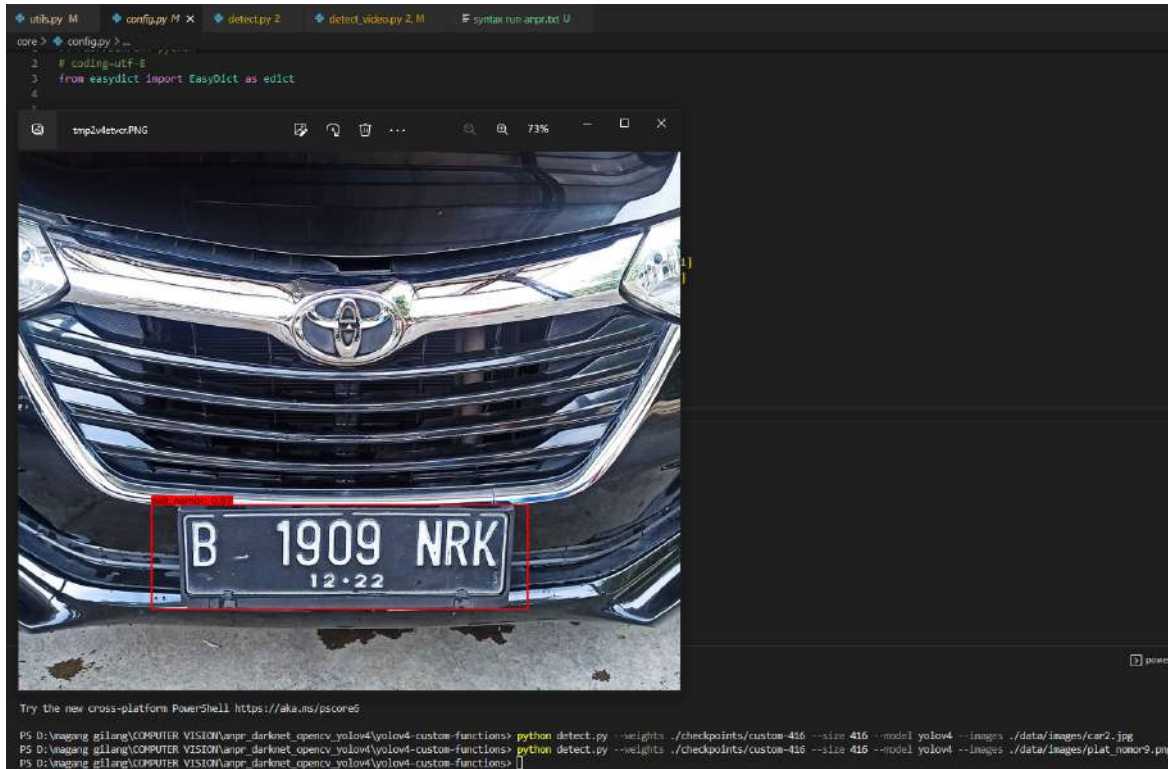
Setelah convert selesai, akan muncul sebuah folder bernama “checkpoints/custom-416”, dalam folder tersebut terdapat file “saved_model.pb” serta folder “assets” dan “variables”.



5. Tes Model yang telah Di-convert

Tes model dapat dijalankan dengan syntax di bawah ini:

“python detect.py --weights ./checkpoints/custom-416 --size 416 --model yolov4 --images ./data/images/plat_nomor9.png”



Deteksi plat nomor pada gambar berhasil dijalankan, setelah itu coba untuk menjalankan deteksi plat nomor langsung dengan kamera. Jalankan syntax di bawah ini:

“`python detect_video.py --weights ./checkpoints/custom-416 --size 416 --model yolov4 --video 0`”
Untuk menutup atau menghentikan program dapat dengan menekan huruf “q” pada keyboard atau tekan “`ctrl+c`”.

6. Memperbaiki Error Pada Deteksi Plat Nomor

Pada proses deteksi terdapat beberapa masalah yang membuat bounding box tidak dapat muncul, berikut adalah beberapa error yang terjadi dan solusinya:

6.1. Error Pertama

Error ini berada di file “`yolov4-custom-function/core/Utils.py`”, kode tersebut berada di antara baris 617-622(jika dari clone repository):

sebelum diganti

```
c1, c2 = (coor[0], coor[1]), (coor[2], coor[3])
```

setelah diganti

```
c1, c2 = (int(coor[0]), int(coor[1])), (int(coor[2]), int(coor[3]))
```


6.2 Error Kedua

```
phykurox commented on Jun 23, 2021 • edited +
After saving the model, running detection with 'python detect.py --weights ./checkpoints/yolov4-416 --size 416 --model yolov4 --
images ./data/images/kite.jpg' give me an error of:
cv2.error: OpenCV(4.5.2) :-1: error: (-5:Bad argument) in function 'rectangle' > Overload resolution failed: > - Can't
parse 'pt1'. Sequence item with index 0 has a wrong type > - Can't parse 'pt1'. Sequence item with index 0 has a wrong type
> - Can't parse 'rec'. Expected sequence length 4, got 2 > - Can't parse 'rec'. Expected sequence length 4, got 2
```

Solusinya yaitu mengganti kode pada file “yolov4-custom-function/core/utils.py”, kode tersebut berada di antara baris 257-262(jika dari clone repository):

sebelum diganti

```
cv2.rectangle(image, c1, (np.float32(c3[0]), np.float32(c3[1])), bbox_color, -1) #filled
```

setelah diganti

```
cv2.rectangle(image, c1, (int(np.float32(c3[0])), int(np.float32(c3[1]))), bbox_color, -1)
```

Kode selanjutnya yang perlu diganti adalah:

sebelum diganti

```
cv2.putText(image, bbox_mess, (c1[0], np.float32(c1[1] - 2)), cv2.FONT_HERSHEY_SIMPLEX,
fontScale, (0, 0, 0), bbox_thick // 2, lineType=cv2.LINE_AA)
```

setelah diganti

```
cv2.putText(image, bbox_mess, (c1[0], int(np.float32(c1[1] - 2))), cv2.FONT_HERSHEY_SIMPLEX,
fontScale, (0, 0, 0), bbox_thick // 2, lineType=cv2.LINE_AA)
```

6.3 Error Ketiga

Error ini terjadi saat syntax di bawah dijalankan:

“python detect_video.py --weights ./checkpoints/yolov4-416 --size 416 --model yolov4 --video 0
--output ./detections/results.avi”

Berikut adalah error yang muncul:

```
OpenCV: FFMPEG: tag 0x34504d46/'FMP4' is not supported with codec id 12 and format 'mp4 / MP4 (MPEG-4 Part 14)'
OpenCV: FFMPEG: fallback to use tag 0x7634706d/'mp4v'
```

Kode yang menyebabkan error berada di file “yolov4-custom-function/detect_video.py” antara baris 71-74(jika dari clone repository).

sebelum diganti

```
codec = cv2.VideoWriter_fourcc(*FLAGS.output_format)
```

sesudah diganti

```
codec = cv2.VideoWriter_fourcc('m', 'p', '4', 'v')
```


6.4 Error Keempat

Saat ingin menjalankan kode tiba-tiba terjadi error tidak dapat menjalankan OpenCV, berikut adalah errornya.

```
File "detect_video.py", line 165, in main
    cv2.namedWindow("result", cv2.WINDOW_AUTOSIZE)
cv2.error: OpenCV(4.5.4) D:\a\opencv-python\opencv-python\opencv\modules\highgui\src\window.cpp:1257: error: (-2:Unspecified error) call libgtk2.0-dev and pkg-config, then re-run cmake or configure script in function 'cvNamedWindow'
```

```
File "C:\Users\Office-017\AppData\Local\Programs\Python\Python38\lib\site-packages\imutils\convenience.py", line 65, in <mod
def resize(image, width=None, height=None, inter=cv2.INTER_AREA):
AttributeError: module 'cv2' has no attribute 'INTER_AREA'
```

Berikut adalah syntax yang saya jalankan dan berhasil menghilangkan error tersebut.

```
pip uninstall opencv-python-headless -y

pip install opencv-python --upgrade
```

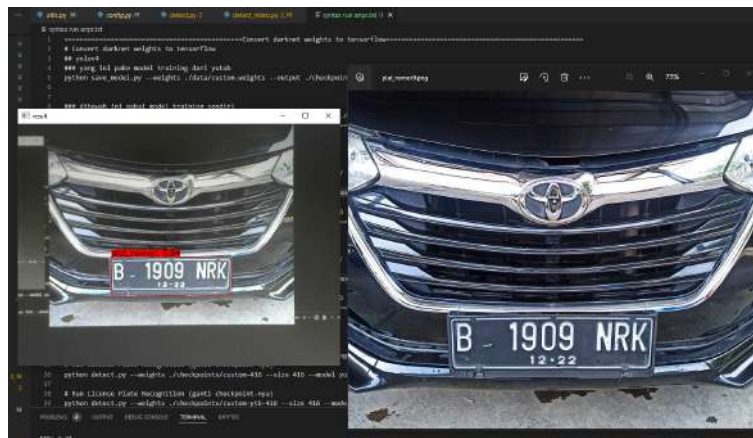
```
pip uninstall opencv-python

then

pip install opencv-python
```

7. Tes Deteksi Dengan Kamera

Deteksi plat nomor dengan kamera dapat dijalankan dengan baik, FPS yang ditangkap dapat lebih tinggi lagi jika menggunakan GPU.



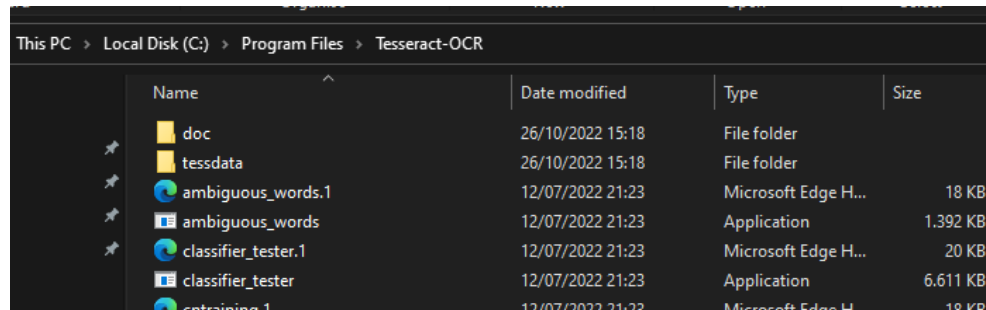
8. Install TesseractOCR untuk Deteksi Karakter

Optical Character Recognition(OCR) berfungsi untuk mengenali karakter pada gambar, dalam dokumentasi ini karakter yang ingin dikenali adalah angka dan huruf di dalam plat nomor, OCR yang digunakan pada github repository di atas adalah TesseractOCR, sebelum dapat

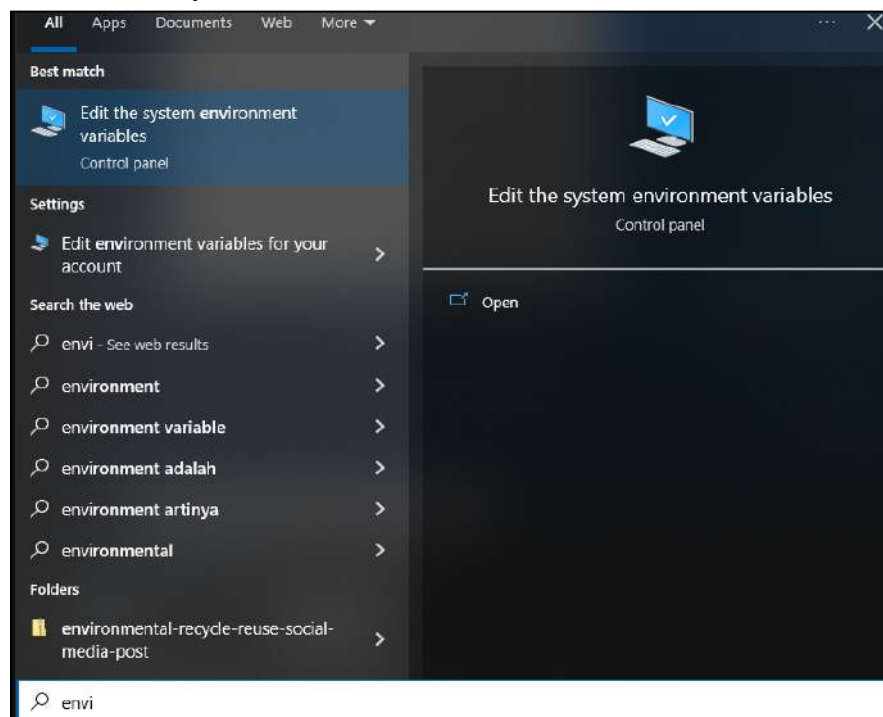
menggunakannya kita harus meng-install TesseractOCR pada PC local kita, tutorial install dapat diakses pada link berikut: <https://www.youtube.com/watch?v=Rb93uLXiTwA>

File download bisa didapatkan di link: <https://github.com/UB-Mannheim/tesseract/wiki>

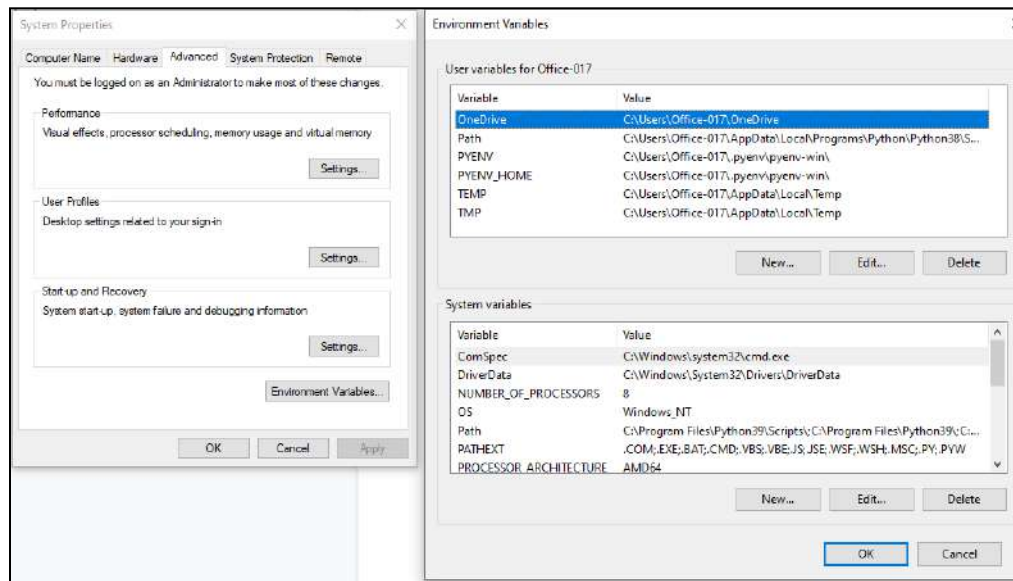
Setelah konfigurasi dan install berhasil dilakukan, pastikan folder “Tesseract-OCR” sudah muncul, salin path folder dari Tesseract-OCR, di sini folder path saya adalah “C:\Program Files\Tesseract-OCR”.



Selanjutnya buka “Edit the system environment variables”

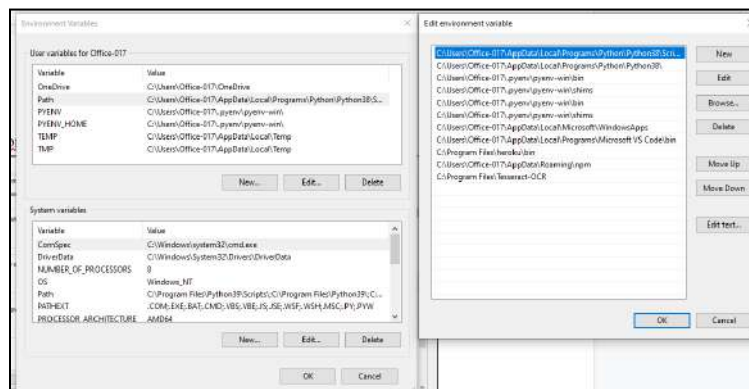


Kemudian klik “Environment Variables” dan akan muncul window baru



Pada bagian “User variables for Office-017”(nama user bisa berbeda, tergantung PCnya):
klik variable “Path” -> klik “Edit” -> klik “New” -> paste path “C:\Program Files\Tesseract-OCR” -> klik
“OK”

Pastikan folder path TesseractOCR sudah masuk, jika belum ada bisa klik New -> paste folder path
TesseractOCR.



Ada kemungkinan TesseractOCR masih belum bisa dijalankan pada command prompt, oleh karena itu tambahkan juga folder path TesseractOCR pada path di “System variables”, proses sama seperti sebelumnya.

9. Tes TesseractOCR pada Sembarang Teks

Buka command prompt -> siapkan gambar dengan teks digital (bukan tulisan tangan) -> jalankan syntax “tesseract namagambar.format ‘namahasil’ ”

Berikut adalah syntax dengan nama file gambar dan file txt hasil yang saya coba

```
Microsoft Windows [Version 10.0.19043.2130]
(c) Microsoft Corporation. All rights reserved.

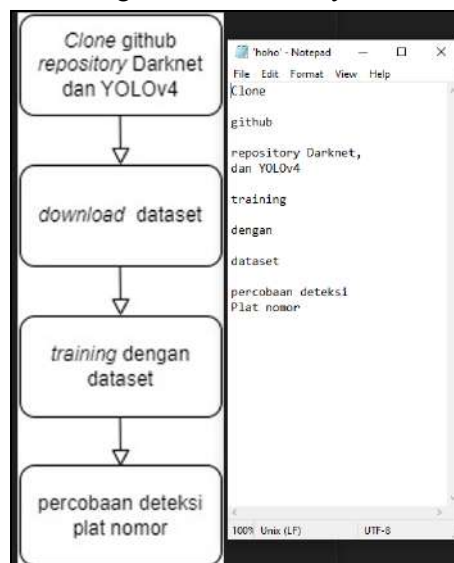
C:\Users\Office-017>tesseract
Usage:
  tesseract --help | --help-extra | --version
  tesseract --list-langs
  tesseract imagename outputbase [options...] [configfile...]

OCR options:
  -l LANG[+LANG]      Specify language(s) used for OCR.
NOTE: These options must occur before any configfile.

Single options:
  --help              Show this help message.
  --help-extra        Show extra help for advanced users.
  --version            Show version information.
  --list-langs        List available languages for tesseract engine.

C:\Users\Office-017>cd Downloads
C:\Users\Office-017\Downloads>tesseract hoho.jpg 'hoho'
Estimating resolution as 117
C:\Users\Office-017\Downloads>_
```

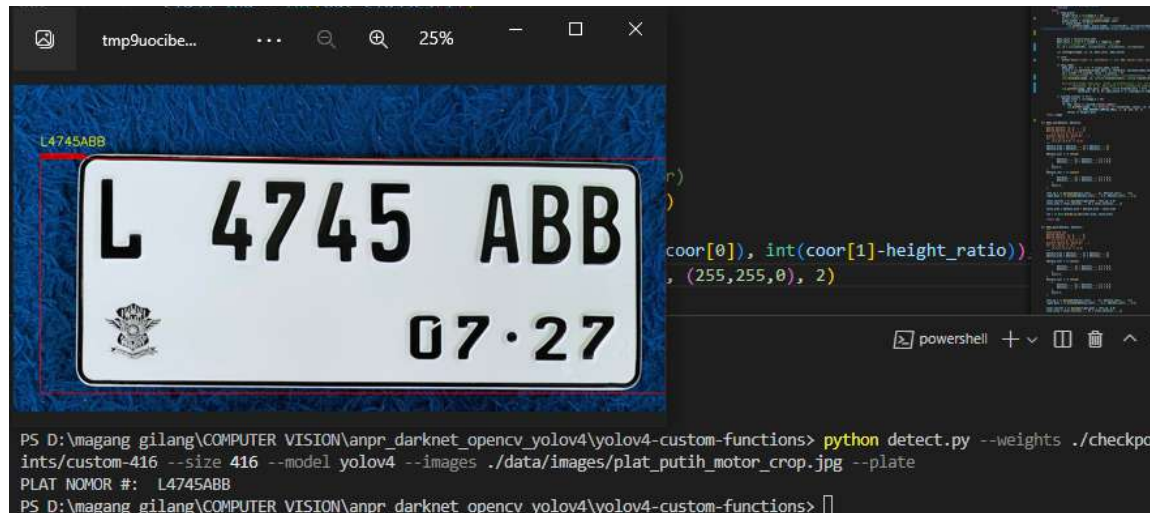
gambar dan hasilnya



TesseractOCR dapat mengenali teks pada gambar, Tesseract akan bekerja lebih baik pada teks dengan font yang sering digunakan pada dokumen-dokumen seperti font Times New Roman, untuk font yang belum dikenali perlu dilakukan proses training font.

10. Penambahan Tesseract OCR pada Deteksi Plat Nomor

Kode untuk menambahkan OCR pada deteksi plat nomor sudah tersedia dari repository github yang kita clone sebelumnya, menjalankan OCR pada repository tersebut sangat mudah, dengan menambahkan flag “**--plate**” maka OCR dapat berjalan, berikut adalah contohnya:



Pada deteksi plat nomor, gambar perlu dilakukan beberapa proses dengan OpenCV agar kontur dapat ditemukan dan TesseractOCR hanya mendeteksi kontur terpilih, gambar-gambar di bawah ini akan menunjukkan prosesnya. Pemrosesan gambar tersebut berada di file “utils.py” pada function “**recognize_plate(img, coords)**”.

```
#percobaan recognize plate tesseract, proses dimodifikasi
def recognize_plate(img, coords):

    # separate coordinates from box
    xmin, ymin, xmax, ymax = coords
    # get the subimage that makes up the bounded region and tak
    box = img[int(ymin)-5:int(ymax)+5, int(xmin)-5:int(xmax)+5]
    #box = img[int(ymin):int(ymax), int(xmin):int(xmax)]

    # grayscale region within bounding box
    gray = cv2.cvtColor(box, cv2.COLOR_RGB2GRAY)
```

Function “**recognize_plate(img, coords)**” mengambil koordinat bounding box yang telah dibuat, menambahkan 5 pixel tambahan pada tinggi dan lebar bounding box, kemudian menormalisasi gambar dengan convert RGB ke GRAY. Sumber:

<https://mti.binus.ac.id/2017/07/03/optical-character-recognition-ocr/>

kode lama

```
# resize image to three times as large as original for better readability
gray = cv2.resize(gray, None, fx = 3, fy = 3, interpolation = cv2.INTER_CUBIC)
```

kode baru

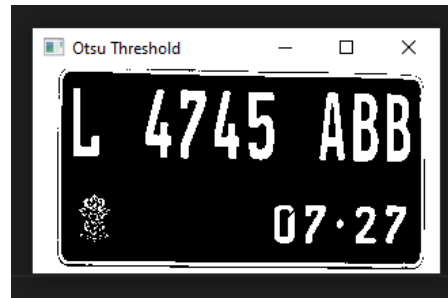
```
#ubah ukuran gambar menjadi 300x150
gray = cv2.resize(gray, (300, 150))
#cv2.imshow("Gray", gray)
#cv2.waitKey(0)
```

Mengubah ukuran gambar agar TesseractOCR lebih mudah mengenali teks, kode yang lama mengubah gambar menjadi 3x lebih besar dari ukuran sebenarnya, namun hal ini akan menyulitkan OCR untuk mengenali gambar jika yang dihasilkan masih terlalu kecil ataupun terlalu besar.

Selanjutnya adalah mengubah gambar dengan filter otsu threshold, berikut adalah kodenya

```
ret, thresh = cv2.threshold(gray, 0, 255, cv2.THRESH_OTSU | cv2.THRESH_BINARY_INV)
cv2.imshow("Otsu Threshold", thresh)
cv2.waitKey(0)
```

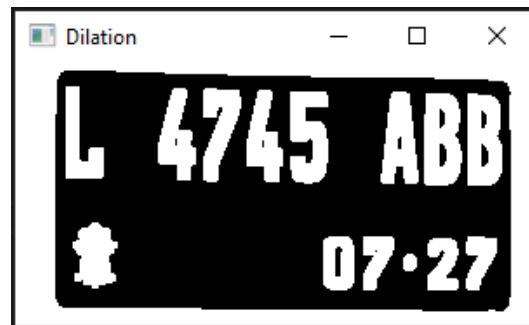
hasilnya



Proses berikutnya adalah mendapatkan struktur dari karakter, serta menerapkan dilation untuk menambah ukuran pada karakter, berikut adalah kodenya:

```
rect_kern = cv2.getStructuringElement(cv2.MORPH_RECT, (4,4))
dilation = cv2.dilate(thresh, rect_kern, iterations = 1)
cv2.imshow("Dilation", dilation)
cv2.waitKey(0)
```

hasilnya



Sumber dilation: <https://skillplus.web.id/dilation-dan-erosion/>

Selanjutnya adalah mencari kontur dari gambar lalu mengurutkan kontur yang ditemukan menjadi kiri ke kanan.

```
try:
    contours, hierarchy = cv2.findContours(dilation, cv2.RETR_TREE, cv2.CHAIN_APPROX_SIMPLE)
except:
    ret_img, contours, hierarchy = cv2.findContours(dilation, cv2.RETR_TREE, cv2.CHAIN_APPROX_SIMPLE)

# sort contours left-to-right
sorted_contours = sorted(contours, key=lambda ctr: cv2.boundingRect(ctr)[0])
```

Proses selanjutnya adalah mengatur pilihan ukuran karakter, hal ini dilakukan untuk menghindari deteksi karakter yang tidak diinginkan dengan ukuran tertentu, contohnya adalah logo POLRI di

kiri bawah, untuk mengatur karakter ukuran karakter yang akan dideteksi dapat diatur pada kode di bawah di baris 81 hingga 95 (mungkin dapat berbeda baris jika dari clone repository).

```

78 for cnt in sorted_contours:
79     x,y,w,h = cv2.boundingRect(cnt)
80     height, width = im2.shape
81     # if height of box is not tall enough relative to total height then skip
82     if height / float(h) > 6: continue
83
84     ratio = h / float(w)
85     # if height to width ratio is less than 1.5 skip
86     if ratio < 1.5: continue
87
88     # if width is not wide enough relative to total width then skip
89     if width / float(w) > 15: continue
90
91     area = h * w
92     # if area is less than 100 pixels skip
93     if area < 100: continue
94     # draw the rectangle
95     rect = cv2.rectangle(im2, (x,y), (x+w, y+h), (0,255,0),2)
96     # grab character region of image
97     roi = thresh[y-5:y+h+5, x-5:x+w+5]
98     # perform bitwise not to flip image to black text on white background
99     roi = cv2.bitwise_not(roi)
100     #cv2.imshow("ROI", roi)
101     #cv2.waitKey(0)
102     # perform another blur on character region
103     roi = cv2.medianBlur(roi, 5)
104     try:
105         text = pytesseract.image_to_string(roi, config='--c tessedit_char_whitelist=0123456789ABCDEF GHIJKLMNOPQRSTUVWXYZ --psm 8 --oem 3')
106         #text = pytesseract.image_to_string(roi, config='--psm 8 --oem 3')
107         # clean tesseraact text by removing any unwanted blank spaces
108         clean_text = re.sub('[\W_]+', '', text)
109         #clean_text = re.sub(text)
110         plate_num += clean_text
111         #plate_num += text
112     except:
113         text = None

```

Dari seluruh kode yang dijalankan, berikut ini adalah hasil deteksi karakter pada plat nomor, kotak hitam tiap karakter menandakan karakter tersebut terdeteksi, OCR akan bekerja dengan mencocokkan karakter pada tiap kotak hitam dengan teks yang dikenali, namun tetap masih ada kemungkinan OCR salah mendeteksi karakter.

hasil deteksi karakter



hasil teks yang terdeteksi



11. Kendala yang Dialami pada Segmentasi Karakter

Terdapat error saat menjalankan deteksi plat nomor dengan TesseractOCR melalui kamera, tepatnya error pada manipulasi gambar dengan OpenCV (dari file “utils.py”), sampai dokumentasi ini diketik masih belum mendapatkan solusinya.

dapat berjalan beberapa detik



terkadang langsung mati dengan error seperti di bawah



12. Alternatif OCR dengan EasyOCR

TesseractOCR merupakan OCR yang ringan untuk dijalankan namun perlu konfigurasi lebih lanjut untuk mendapatkan hasil yang lebih baik, terdapat OCR yang dapat berjalan dengan cukup baik namun memakan waktu pemrosesan yang lebih lama yaitu EasyOCR. Referensi deteksi plat nomor dengan EasyOCR dapat dilihat pada link berikut:

https://www.youtube.com/watch?v=0-4p_QgrdbE (pada jam 1:13:00)

Install pytorch terlebih dahulu, buka <https://pytorch.org/> kemudian sesuaikan instalasi dengan spesifikasi dan dependensi pada PC local, di bawah ini adalah konfigurasi milik saya, PyTorch Stable, Windows, Pip, Python, dan CPU. Akan muncul syntax di bawah, copy dan jalankan di command prompt.

PyTorch Build	Stable (1.13.0)	Preview (Nightly)		
Your OS	Linux	Mac	Windows	
Package	Conda	Pip	LibTorch	Source
Language	Python	C++ / Java		
Compute Platform	CUDA 11.6	CUDA 11.7	ROCM 5.2	CPU

Run this Command: `pip install torch torchvision torchaudio`

NOTE: PyTorch LTS has been deprecated. For more information, see [this blog](#).

proses instalasi dengan command prompt

```
Microsoft Windows [Version 10.0.19043.2130]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Office-017>pip3 install torch torchvision torchaudio
WARNING: Ignoring invalid distribution -pencv-python-headless (c:\users\office-017\appdata\local\programs\python\python38\lib\site-packages)
WARNING: Ignoring invalid distribution -pencv-python-headless (c:\users\office-017\appdata\local\programs\python\python38\lib\site-packages)
Requirement already satisfied: torch in c:\users\office-017\appdata\local\programs\python\python38\lib\site-packages (1.13.0)
Requirement already satisfied: torchvision in c:\users\office-017\appdata\local\programs\python\python38\lib\site-packages (0.14.0)
Requirement already satisfied: torchaudio in c:\users\office-017\appdata\local\programs\python\python38\lib\site-packages (0.13.0)
Requirement already satisfied: typing-extensions in c:\users\office-017\appdata\local\programs\python\python38\lib\site-packages (from torch) (3.7.4.3)
Requirement already satisfied: requests in c:\users\office-017\appdata\local\programs\python\python38\lib\site-packages (from torchvision) (2.28.1)
Requirement already satisfied: pillow<8.3.0, >=5.3.0 in c:\users\office-017\appdata\local\programs\python\python38\lib\site-packages (from torchvision) (9.2.0)
Requirement already satisfied: numpy in c:\users\office-017\appdata\local\programs\python\python38\lib\site-packages (from torchvision) (1.23.4)
Requirement already satisfied: urllib3<1.27, >=1.21.1 in c:\users\office-017\appdata\local\programs\python\python38\lib\site-packages (from requests->torchvision) (1.26.12)
Requirement already satisfied: idna<4, >=2.5 in c:\users\office-017\appdata\local\programs\python\python38\lib\site-packages (from requests->torchvision) (3.4)
Requirement already satisfied: certifi<2017.4.17 in c:\users\office-017\appdata\local\programs\python\python38\lib\site-packages (from requests->torchvision) (2022.9.24)
Requirement already satisfied: charset-normalizer<3, >=2 in c:\users\office-017\appdata\local\programs\python\python38\lib\site-packages (from requests->torchvision) (2.1.1)
```

Install EasyOCR dengan syntax “**pip install easyocr**” (pada jam 1:12:30 di link referensi)

```

C:\Users\Office-017>pip install easyocr
WARNING: Ignoring invalid distribution -pencv-python-headless (c:\users\office-017\appdata\local\programs\python\python38\lib\site-packages)
WARNING: Ignoring invalid distribution -pencv-python-headless (c:\users\office-017\appdata\local\programs\python\python38\lib\site-packages)
Requirement already satisfied: easyocr in c:\users\office-017\appdata\local\programs\python\python38\lib\site-packages (1.6.2)
Requirement already satisfied: scipy in c:\users\office-017\appdata\local\programs\python\python38\lib\site-packages (from easyocr) (1.4.1)
Requirement already satisfied: Pillow in c:\users\office-017\appdata\local\programs\python\python38\lib\site-packages (from easyocr) (9.2.0)
Requirement already satisfied: PyYAML in c:\users\office-017\appdata\local\programs\python\python38\lib\site-packages (from easyocr) (5.4.1)
Requirement already satisfied: pyclicker in c:\users\office-017\appdata\local\programs\python\python38\lib\site-packages (from easyocr) (1.3.0.post3)
Requirement already satisfied: torch in c:\users\office-017\appdata\local\programs\python\python38\lib\site-packages (from easyocr) (1.13.0)
Requirement already satisfied: Shapely in c:\users\office-017\appdata\local\programs\python\python38\lib\site-packages (from easyocr) (1.8.5.post1)
Requirement already satisfied: numpy in c:\users\office-017\appdata\local\programs\python\python38\lib\site-packages (from easyocr) (1.23.4)
Requirement already satisfied: torchvision>=0.5 in c:\users\office-017\appdata\local\programs\python\python38\lib\site-packages (from easyocr) (0.14.0)
Requirement already satisfied: ninja in c:\users\office-017\appdata\local\programs\python\python38\lib\site-packages (from easyocr) (1.10.2.4)
Collecting opencv-python-headless<4.5.4.60
  Using cached opencv-python-headless-4.5.4.60-cp38-cp38-win_and64.whl (35.0 MB)
Requirement already satisfied: python-bidi in c:\users\office-017\appdata\local\programs\python\python38\lib\site-packages (from easyocr) (0.4.2)

```

Sekarang coba menjalankan EasyOCR, berikut adalah kode yang saya jalankan di file baru:

```
#open camera with opencv and easyocr
import cv2
import numpy as np
import easyocr
reader = easyocr.Reader(['en']) # need to run only once to load model into memory

cap = cv2.VideoCapture(0)
# Check if camera opened successfully
if (cap.isOpened() == False):
    print("Error opening video stream or file")
while (cap.isOpened()):
    ret, frame = cap.read()
    if ret == True:
        cv2.imshow('frame', frame)
        result = reader.readtext(frame)
        for detection in result:
            top_left = tuple([int(val) for val in detection[0][0]])
            bottom_right = tuple([int(val) for val in detection[0][2]])
            text = detection[1]
            font = cv2.FONT_HERSHEY_SIMPLEX
            frame = cv2.putText(frame, text, top_left, font, 1, (0,255,0), 2, cv2.LINE_AA)

        #jika array result tidak kosong print result, jika kosong print "tidak ada plat"
        if result:
            for hasil in result:
                print(hasil[1])
                #print(result[0][1])
            else:
                print("tidak ada plat")
                if cv2.waitKey(25) & 0xFF == ord('q'):
                    break
        else:
            break
    cap.release()
    cv2.destroyAllWindows()
```

Di bawah ini adalah hasilnya, EasyOCR masih mendeteksi karakter apapun yang masuk ke frame, jika tidak ada karakter lain di luar objek yang ingin dideteksi maka hasilnya dapat lebih baik.



13. Tes Deteksi Plat Nomor dan OCR dengan EasyOCR

Sebelum menjalankan EasyOCR pastikan sudah melakukan import EasyOCR pada file “utils.py” dengan menambahkan kode “import easyocr” di baris-baris atas. Kode bawaan yang di-clone dari github repository menggunakan TesseractOCR sehingga banyak konfigurasi yang diperlukan oleh TesseractOCR namun ada sebagian konfigurasi yang mengalami error ketika TesseractOCR diganti dengan EasyOCR, berikut adalah beberapa baris kode yang harus dimatikan atau diganti.

Kode dari gambar di bawah berada di dalam function “**draw_bbox()**”, kode yang dihilangkan atau dimatikan berada di baris 377-379(jika dari clone repository dapat berbeda baris), baris tersebut berfungsi untuk menampilkan hasil deteksi yang diletakkan di atas bounding box, namun karena konfigurasi yang berbeda dengan EasyOCR maka terjadi error, kode tersebut dapat diganti dengan konfigurasi yang benar atau dimatikan, jika dimatikan maka hasil deteksi muncul di terminal namun tidak muncul pada window “result”.

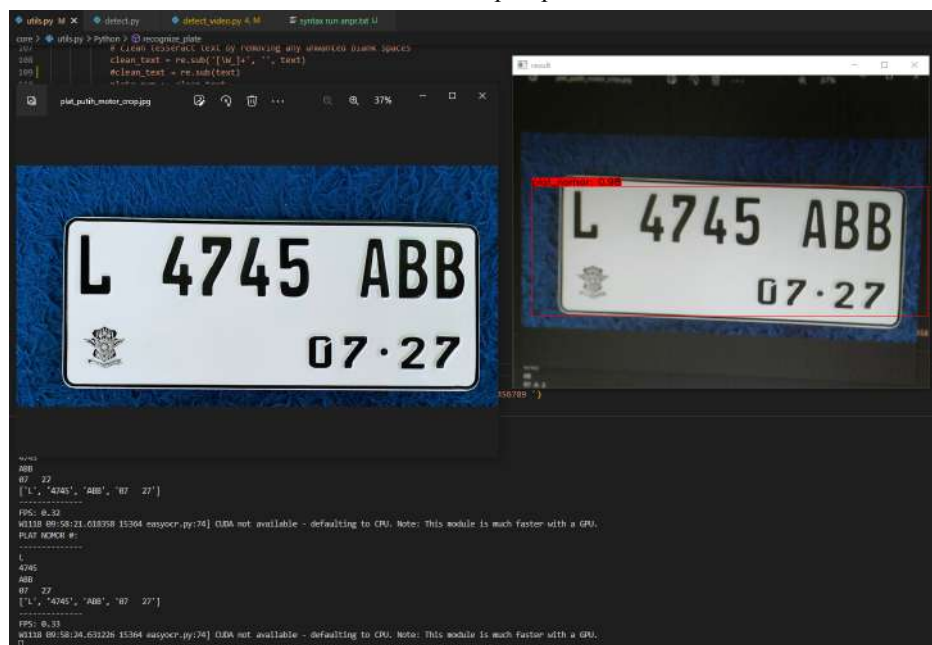
```

350 def draw_bbox(image, bboxes, info = False, counted_classes = None, show_label=True, allowed_classes=list(read_class_names(cfg.YOLO.CLASSES).values()), read_plate = False):
351     classes = read_class_names(cfg.YOLO.CLASSES)
352     num_classes = len(classes)
353     image_h, image_w, _ = image.shape
354     hsv_tuples = [(1.0 * x / num_classes, 1., 1.) for x in range(num_classes)]
355     colors = list(map(lambda x: colorsys.hsv_to_rgb(*x), hsv_tuples))
356     colors = list(map(lambda x: (int(x[0] * 255), int(x[1] * 255), int(x[2] * 255)), colors))
357
358     random.seed(0)
359     random.shuffle(colors)
360     random.seed(None)
361
362     out_boxes, out_scores, out_classes, num_boxes = bboxes
363     for i in range(num_boxes):
364         if int(out_classes[i]) < 0 or int(out_classes[i]) > num_classes: continue
365         coor = out_boxes[i]
366
367         fontScale = 0.5
368         score = out_scores[i]
369         class_ind = int(out_classes[i])
370         class_name = classes[class_ind]
371         if class_name not in allowed_classes:
372             continue
373         else:
374             if read_plate:
375                 height_ratio = int(image_h / 75)
376                 plate_number = recognize_plate(image, coor)
377                 # if plate number != None:
378                 #     cv2.putText(image, plate_number, (int(coor[0]), int(coor[1]-height_ratio)),
379                 #                 cv2.FONT_HERSHEY_SIMPLEX, 1.25, (255,255,0), 2)

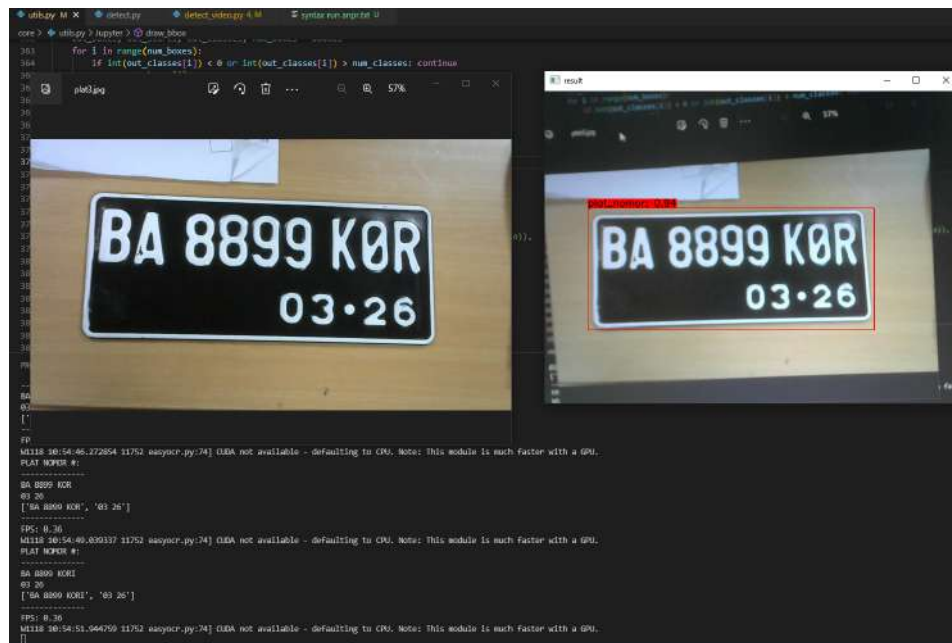
```

Berikut adalah hasilnya setelah dijalankan, dan tidak ada error yang muncul

hasil deteksi plat putih



hasil deteksi plat hitam



Dari hasil deteksi dengan gambar plat putih dan hitam, hasil deteksi pada plat putih lebih baik, tetap ada kesalahan deteksi karakter namun tidak sebanyak pada plat hitam, EasyOCR berjalan lebih lambat daripada TesseractOCR dan juga belum dapat membaca karakter yang disegmentasi dengan OpenCV, EasyOCR dapat menjadi alternatif jika proses segmentasi karakter dengan OpenCV masih terjadi error.

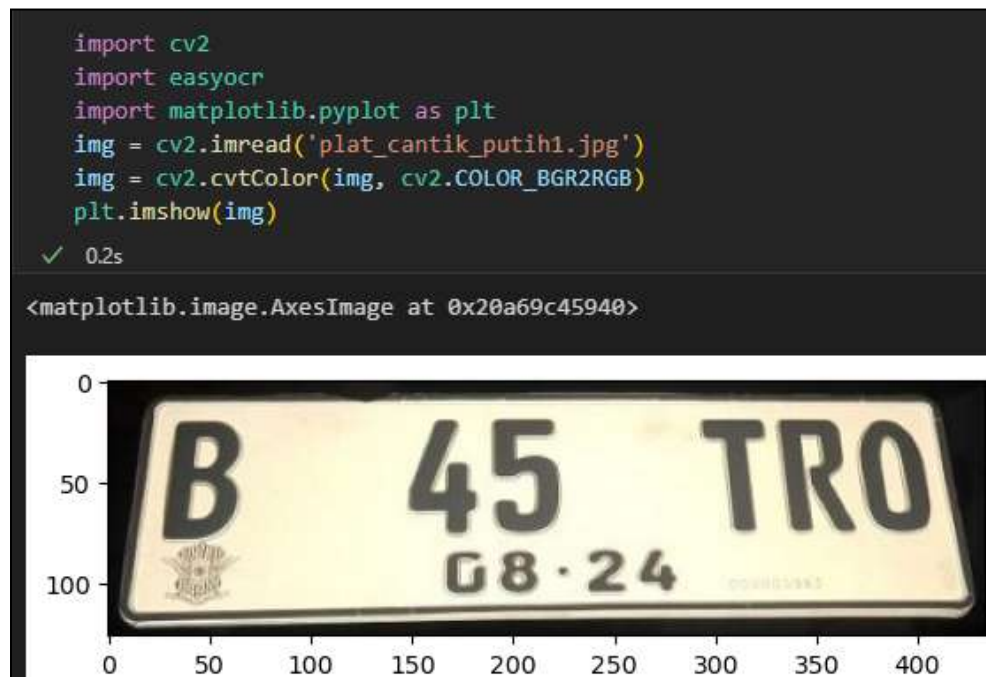
EasyOCR sangat mudah digunakan, namun jika jarak antar karakter dalam satu baris terlalu jauh maka EasyOCR secara otomatis akan membuat bounding box terpisah, oleh karena itu kita perlu menggabungkan bounding box yang terpisah namun pada baris yang sama menjadi satu bounding box utuh, proses akan dijabarkan di bawah.

14. Setting Parameter EasyOCR untuk Segmentasi Karakter

Berikut adalah gambar yang digunakan dalam percobaan dan hasil yang ditampilkan.

Referensi parameter EasyOCR: <https://www.jaided.ai/easyocr/documentation/>

gambar



hasil deteksi

```
#tes easyocr dengan gambar
import easyocr
box = cv2.imread('plat_cantik_putih1.jpg')
text = easyocr.Reader(['en']).readtext(box, allowlist='ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789 ')
#reader = easyocr.Reader(['en'], allowlist = "0123456789") # need to run only once to load mod
#result = reader.readtext('plat_cantik_putih.jpg')

✓ 2.6s

CUDA not available - defaulting to CPU. Note: This module is much faster with a GPU.

for hasil in text:
    print(hasil[1].upper())

✓ 0.3s

B
45
TRO
6 8
24
```

Ada beberapa karakter yang dipisah karena jarak dengan karakter sebelumnya terlalu jauh, kita akan coba tambahkan parameter “width_ths=” untuk menggabungkan karakter dengan baris yang sama.

format hasil dengan “width_ths”

```
#tes easyocr dengan gambar
import easyocr
box = cv2.imread('plat_cantik_putih1.jpg')
text = easyocr.Reader(['en']).readtext(box, allowlist='ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789 ', width_ths=2)
#reader = easyocr.Reader(['en'], allowlist = "0123456789") # need to run only once to load model into memory
#result = reader.readtext('plat_cantik_putih.jpg')

✓ 2.5s

CUDA not available - defaulting to CPU. Note: This module is much faster with a GPU.

for hasil in text:
    print(hasil[1].upper())

✓ 0.4s

B 45 TRO
68 24
```

Karakter sudah tersegmentasi sesuai barisnya, sekarang coba terapkan pada kode di deteksi plat nomor dengan kamera, rubah isi function “recognize_plate(img, coords)” pada file “utils.py”.

Buka file “utils.py”, cari function “recognize_plate(img, coords)” yang menggunakan EasyOCR, berikut adalah milik saya sebelum ditambahkan parameter “width_ths=”.

```
# function to recognize license plate numbers using easyocr from coordinates
def recognize_plate(img, coords):
    #take coordinates from box
    xmin, ymin, xmax, ymax = coords

    #detection license plate from coordinates
    box = img[int(ymin):int(ymax), int(xmin):int(xmax)]

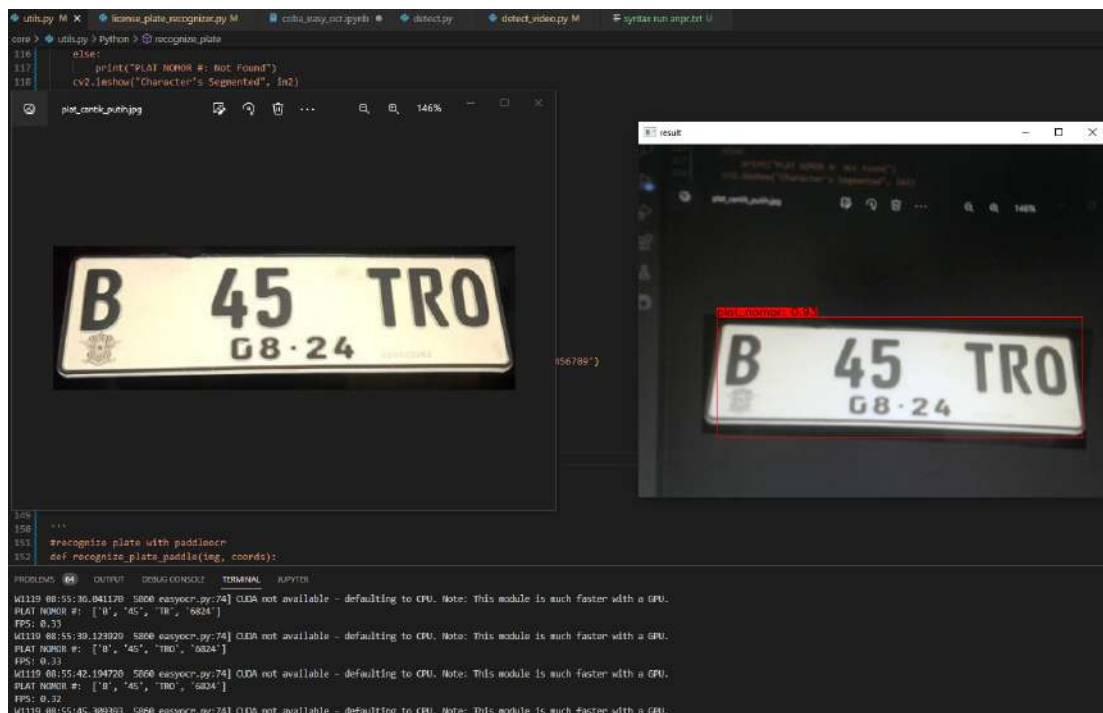
    box = cv2.resize(box, (600, 300))
    #cv2.imshow("resize box", box)
    #cv2.waitKey(0)
    text = easyocr.Reader(['en']).readtext(box, allowlist='ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789')

    #print the result
    hasil_final = []

    for i in text:
        hasil_final.append(i[1])
    print("PLAT NOMOR #: ", hasil_final)

    return text
```

Berikut adalah hasil deteksi tanpa penambahan parameter “width_ths=”.



Sekarang kita coba tambahkan parameter “width_ths=”, di bawah ini adalah milik saya, besaran nilai parameter tersebut dapat diatur sesuai keinginan, jika dengan nilai kecil karakter belum tergabung coba ganti nilai parameter dengan nilai yang lebih besar.

```
# function to recognize license plate numbers using easyocr from coordinates
def recognize_plate(img, coords):
    #take coordinates from box
    xmin, ymin, xmax, ymax = coords

    #detection license plate from coordinates
    box = img[int(ymin):int(ymax), int(xmin):int(xmax)]

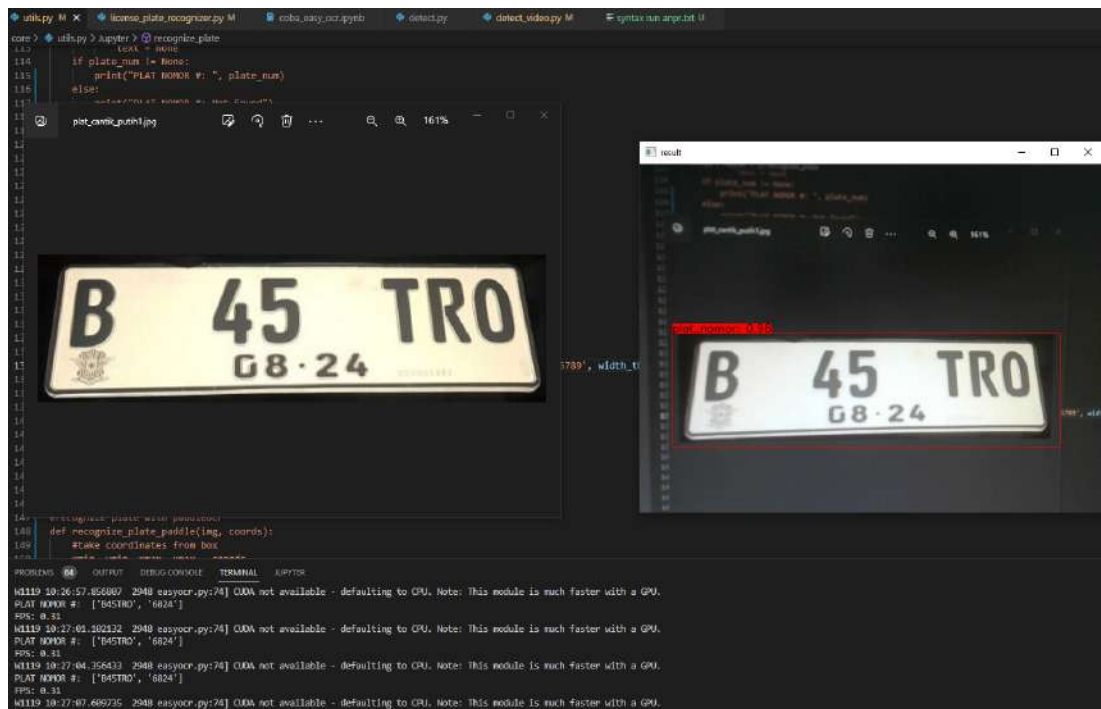
    box = cv2.resize(box, (600, 300))
    #cv2.imshow("resize box", box)
    #cv2.waitKey(0)
    text = easyocr.Reader(['en']).readtext(box, allowlist='ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789', width_ths = 5)

    #print the result
    hasil_final = []

    for i in text:
        hasil_final.append(i[1])
    print("PLAT NOMOR #: ", hasil_final)

    return text
```

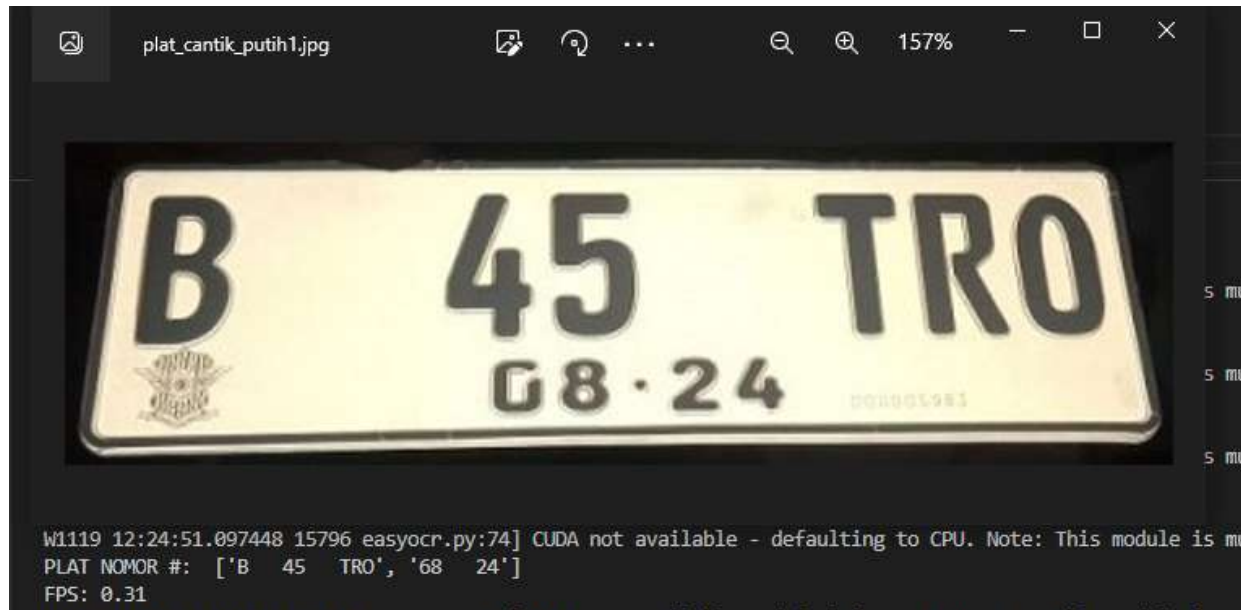
hasil penambahan parameter



Karakter plat nomor sudah tergabung, karakter nomor (yang besar) dengan bulan tahun berlaku (yang kecil di bawah) juga dapat dipisahkan.

Catatan: Setelah dicoba ternyata ada beberapa karakter tanda baca atau spasi yang terbaca sebagai karakter lain, jika itu terjadi saat di tes coba tambahkan karakter spasi pada “allowlist”, namun jika jarak antar karakter terlalu jauh maka spasi yang terbaca juga jauh.

Berikut adalah hasil jika “allowlist” masih ada spasi



Setelah mencoba beberapa parameter yang tersedia pada EasyOCR, ternyata ada parameter yang dapat mempersingkat penulisan kode untuk menampilkan output, nama parameter tersebut adalah “detail=”, parameter tersebut membuat output berupa hasil deteksi tanpa koordinat dan akurasi, sehingga untuk menampilkan hasil deteksi tidak perlu menggunakan looping, berikut ini adalah perbandingannya:

tanpa parameter “detail=”

```
# function to recognize license plate numbers using easyocr from coordinates
def recognize_plate(img, coords):
    #take coordinates from box
    xmin, ymin, xmax, ymax = coords

    #detection license plate from coordinates
    box = img[int(ymin):int(ymax), int(xmin):int(xmax)]

    box = cv2.resize(box, (600, 300))
    #cv2.imshow("resize box", box)
    #cv2.waitKey(0)
    text = easyocr.Reader(['en']).readtext(box, allowlist='ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789', width_ths = 5)

    #print the result
    hasil_final = []

    for i in text:
        hasil_final.append(i[1])
    print("PLAT NOMOR #: ", hasil_final)

    return text
```

Menggunakan looping, kode menjadi lebih panjang, bandingkan dengan gambar di bawah

dengan parameter “detail=”

```
127 # function to recognize license plate numbers using easyocr from coordinates
128 def recognize_plate(img, coords):
129     #take coordinates from box
130     xmin, ymin, xmax, ymax = coords
131     #detection license plate from coordinates
132     box = img[int(ymin):int(ymax), int(xmin):int(xmax)]
133     box = cv2.resize(box, (600, 300))
134     #cv2.imshow("resize box", box)
135     #cv2.waitKey(0)
136     text = easyocr.Reader(['en'], gpu=False).readtext(box, allowlist='ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789 ', width_ths=5, detail=0)
137
138     print("PLAT NOMOR #: ", text)
139
140
141     return text
142
143
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL JUPYTER

W1121 09:34:55.608503 8464 easyocr.py:70] Using CPU. Note: This module is much faster with a GPU.
PLAT NOMOR #: ['3 1386 Q0', '19224']
FPS: 0.33
W1121 09:34:58.648377 8464 easyocr.py:70] Using CPU. Note: This module is much faster with a GPU.
PLAT NOMOR #: ['0 1306 Q0', '10 24']
FPS: 0.34
W1121 09:35:01.642373 8464 easyocr.py:70] Using CPU. Note: This module is much faster with a GPU.
PLAT NOMOR #: ['8 1306 Q0', '10 24']
FPS: 0.33
FPS: 2.40
W1121 09:35:05.042283 8464 easyocr.py:70] Using CPU. Note: This module is much faster with a GPU.
PLAT NOMOR #: ['8 1386 Q0', '10 24']
FPS: 0.32
W1121 09:35:08.174908 8464 easyocr.py:70] Using CPU. Note: This module is much faster with a GPU.
PLAT NOMOR #: ['0 1306 Q0', '10 24']
FPS: 0.32

Tanpa looping namun dapat menampilkan hasil yang sama, maka dari itu saya akan gunakan parameter “detail=”.

Dari output yang muncul terdapat 2 index dalam array, agar lebih mudah untuk dibaca kita bisa memisahkan index array ke-0 dan ke-1, index ke-0 adalah nomor dari TNKB, sedangkan index ke-1 adalah masa berlakunya, di bawah ini adalah cara yang dapat dilakukan untuk memisahkannya.

15. Menambahkan Parameter batch_size pada EasyOCR

Parameter “batch_size” berfungsi untuk meningkatkan performa pengenalan karakter sehingga waktu yang diperlukan dapat lebih singkat, secara default parameter “batch_size=1”, kita dapat mengganti dengan angka yang lebih besar, menaikkan “batch_size” lebih besar akan memakan lebih banyak memory. Tambahkan “batch_size=” pada function “recognize_plate(img, coords)”, bersebelahan dengan parameter “width_ths=” dan “detail=”.

```
127 # function to recognize license plate numbers using easyocr from coordinates
128 #function recognize_plate di bawah ini menggunakan EasyOCR, dapat berjalan tanpa error
129
130 def recognize_plate(img, coords):
131     #take coordinates from box
132     xmin, ymin, xmax, ymax = coords
133     #detection license plate from coordinates
134     box = img[int(ymin):int(ymax), int(xmin):int(xmax)]
135     box = cv2.resize(box, (600, 300))
136     #cv2.imshow("resize box", box)
137     #cv2.waitKey(0)
138     text = easyocr.Reader(['en'], gpu=False).readtext(box, allowlist='ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789 ', width_ths=5, detail=0, batch_size=5)
139
140
141     text_hitung = len(text)
142
```

16. Memisahkan Nomor TKNB dan Masa Berlaku

Kode ditambahkan pada function “recognize_plate(img, coords)” pada file “utils.py”, langkah pertama adalah menghitung panjang dari array variabel “text”, jika panjangnya 0 maka tidak ada karakter plat yang terdeteksi, jika panjangnya 1 maka tampilkan isi dari variabel “text” indeks ke-1, jika panjangnya 2 maka tampilkan isi dari variabel “text” indeks ke-1 dan ke-2.

penambahan kode

```

125 |
126 |
127 | # function to recognize license plate numbers using easyocr from coordinates
128 | def recognize_plate(img, coords):
129 |     # make coordinates from box
130 |     xmin, ymin, xmax, ymax = coords
131 |     # detection license plate from coordinates
132 |     box = img[int(ymin):int(ymax), int(xmin):int(xmax)]
133 |     box = cv2.resize(box, (600, 100))
134 |     # cv2.imshow("resize box", box)
135 |     # cv2.waitKey(0)
136 |     text = easyocr.Reader(['en'], gpu=False).readText(box, allowlist="ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789 ", width_ths=5, detail=0)
137 |
138 |     text_hitung = len(text)
139 |     # print(text_hitung)
140 |     # print("PLAT NOMOR #:", text)
141 |
142 |     if text_hitung == 0:
143 |         print("TNKB #: Not Found")
144 |         print("masa berlaku #: Not Found")
145 |
146 |     elif text_hitung == 1:
147 |         print("TNKB #:", text[0])
148 |         print("masa berlaku #: Not Found")
149 |
150 |     elif text_hitung == 2:
151 |         print("TNKB #:", text[0])
152 |         print("masa berlaku #:", text[1])
153 |
154 |     return text
155 |
156 |

```

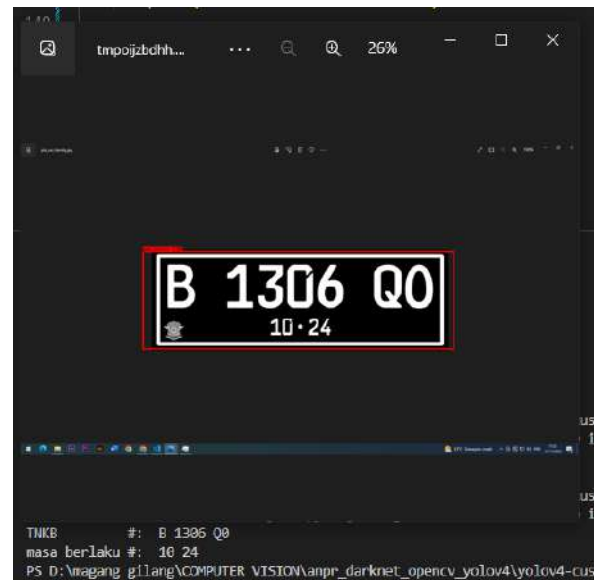
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL JUPYTER

```

W1122 08:35:40.001206 2720 easyocr.py:70] Using CPU. Note: This module is much faster with a GPU.
2
PLAT NOMOR #: ['2345 67', '02 27']
TNKB #: 2345 67
masa berlaku #: 02 27
FPS: 0.32
W1122 08:35:52.238157 2720 easyocr.py:70] Using CPU. Note: This module is much faster with a GPU.
2
PLAT NOMOR #: ['2345 67', '02 27']
TNKB #: 2345 67
masa berlaku #: 02 27
FPS: 0.32
W1122 08:35:55.424818 2720 easyocr.py:70] Using CPU. Note: This module is much faster with a GPU.
2
PLAT NOMOR #: ['2345 67', '02 27']
TNKB #: 2345 67

```

hasil pemisahan TNKB dengan masa berlaku



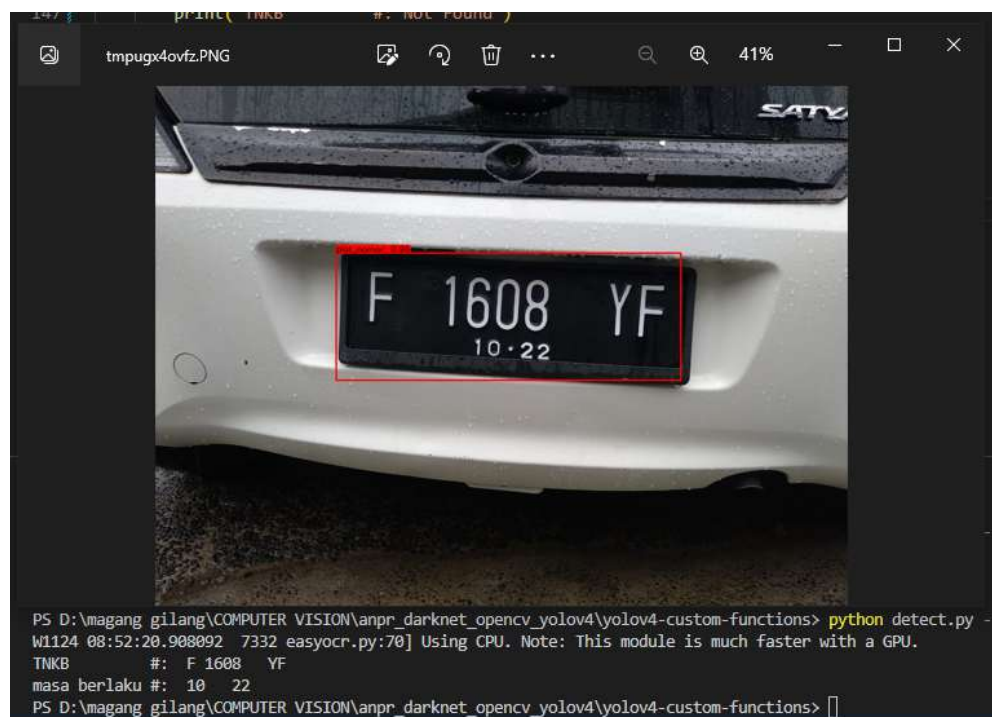
17. Percobaan Pada Beberapa Gambar Plat Nomor

Untuk menjalankan program pada gambar jalankan syntax di bawah ini:

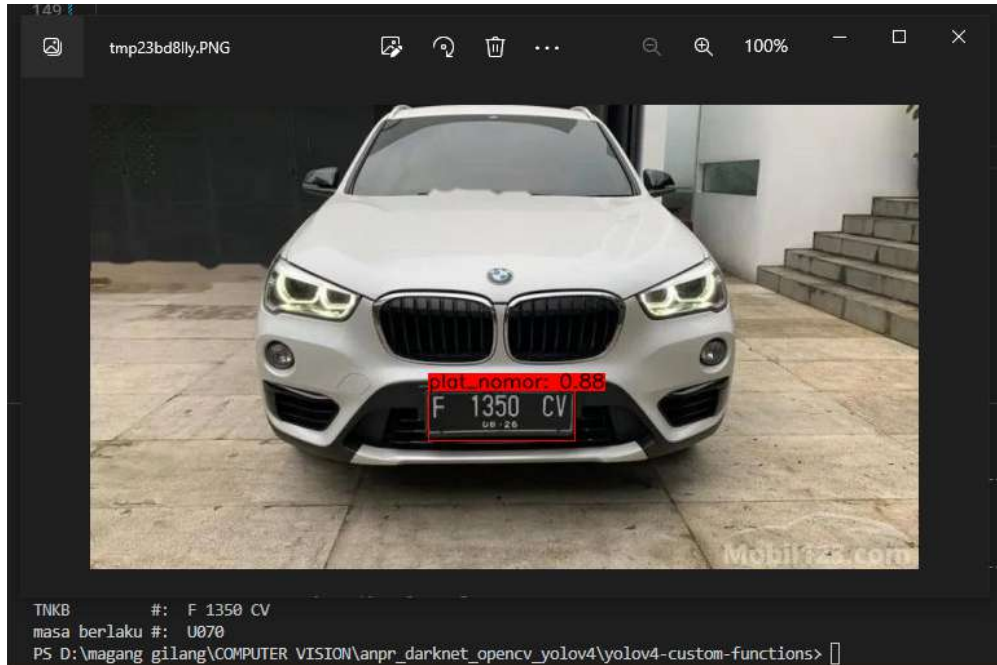
```
“python detect.py --weights ./checkpoints/custom-416 --size 416 --model yolov4 --images  
./data/images/car2.jpg --plate”
```

Sesuaikan path folder gambar yang ingin digunakan, gambar yang saya gunakan berada di path folder “yolov4-custom-functions/data/images”

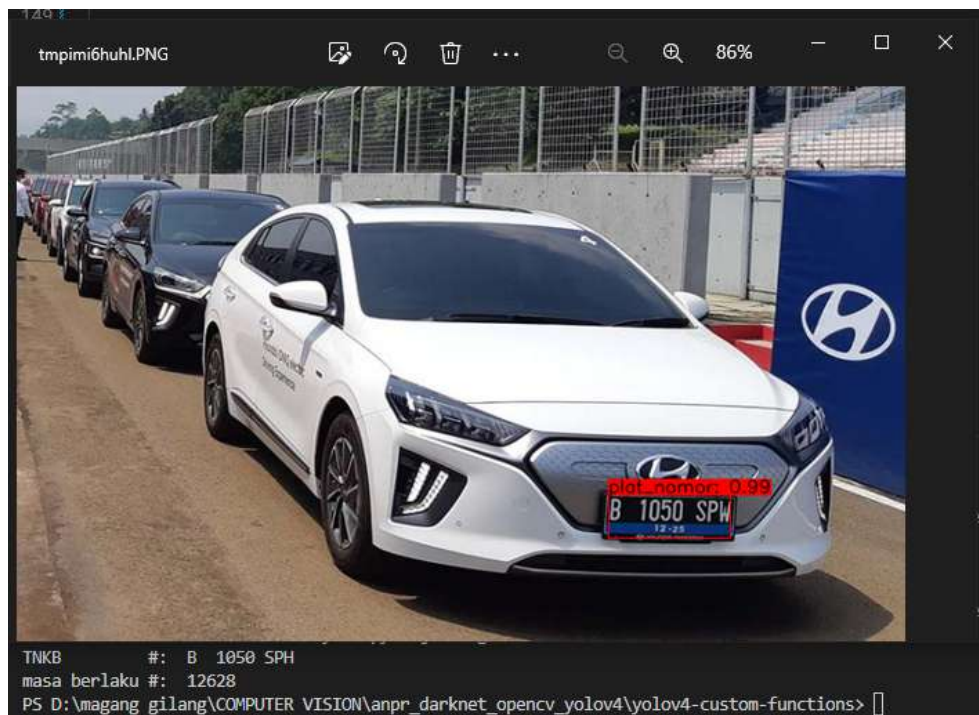
Berikut adalah hasil dari beberapa gambar yang menjadi percobaan:



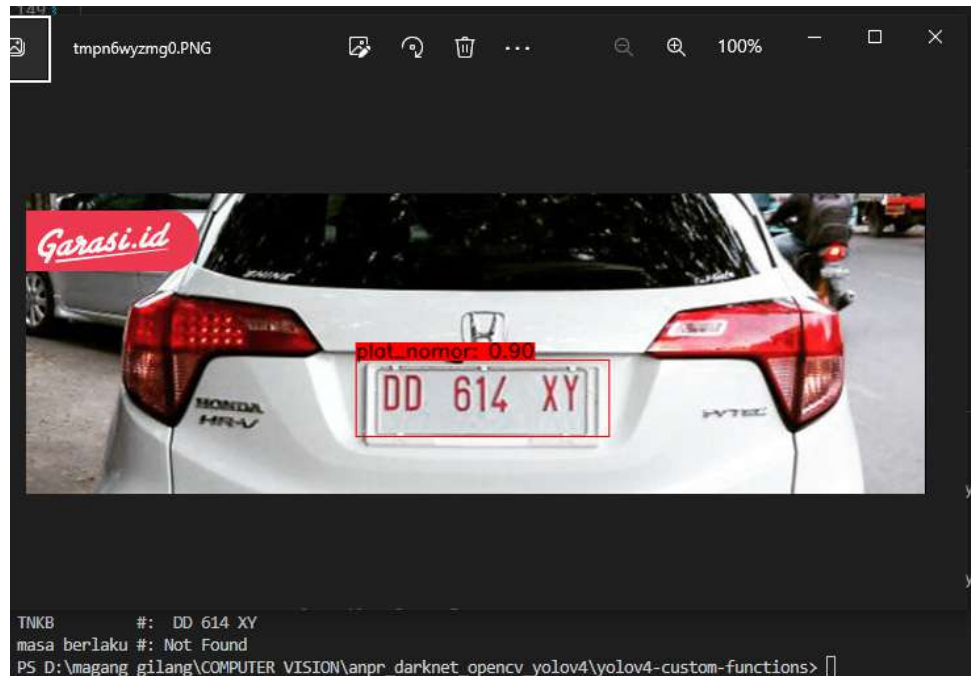
Seluruh karakter dari plat di atas dapat terbaca dengan baik.



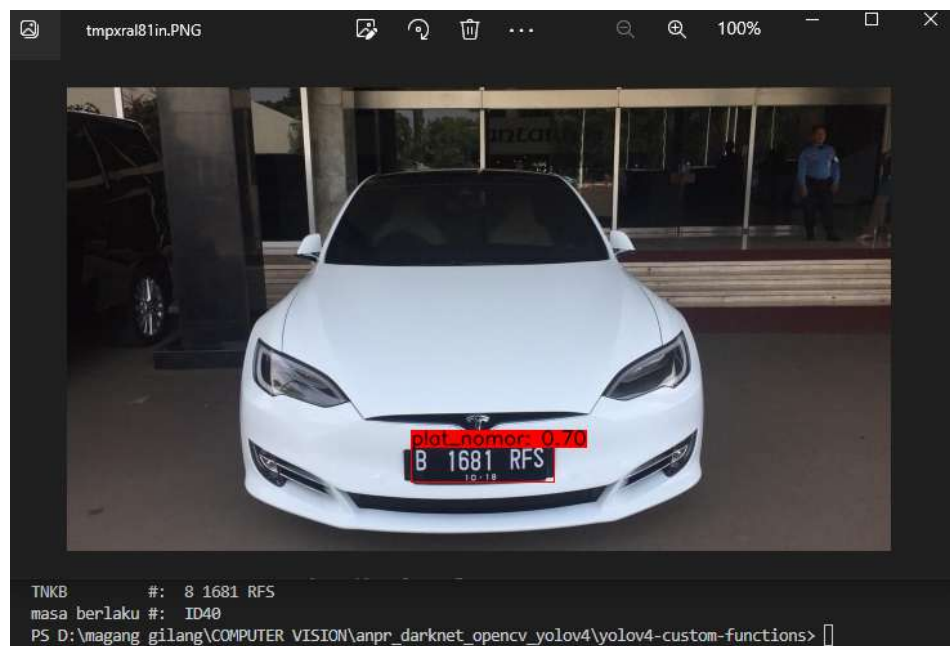
Nomor TNKB dapat terbaca dengan baik namun nomor masa berlaku tidak terbaca dengan baik, hal ini dapat disebabkan karena ukuran dari nomor masa berlaku terlalu kecil dan kurang jelas.



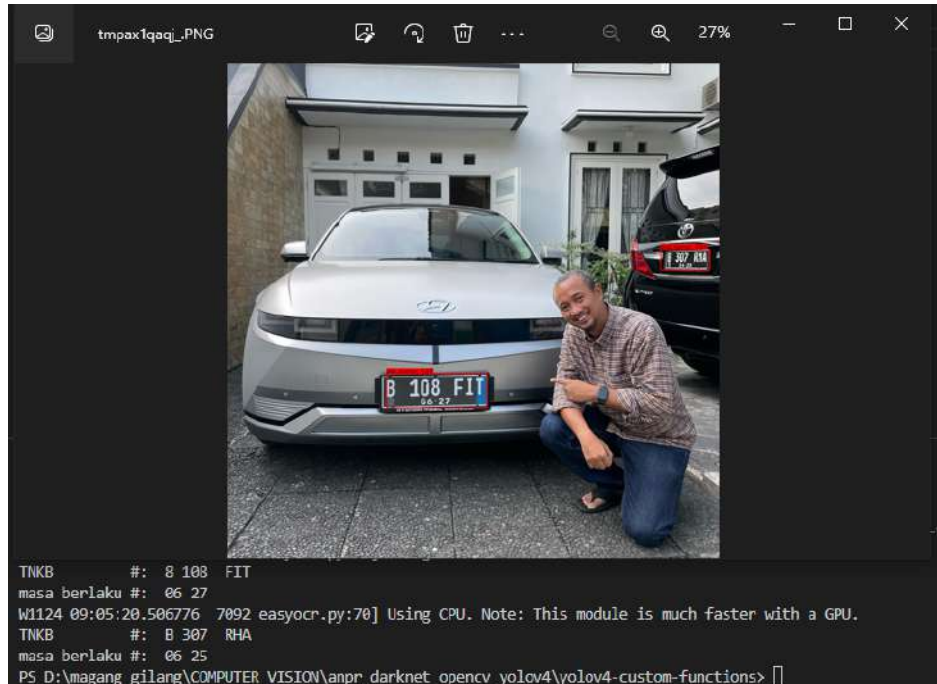
Terdapat kesalahan pada TNKB, EasyOCR kesulitan membedakan antara H dengan W, nomor masa berlaku juga terdapat kesalahan baca (yang benar adalah “12 25” namun terbaca “12628”).



TNKB dapat terbaca dengan baik, plat nomor tidak memiliki nomor masa berlaku.



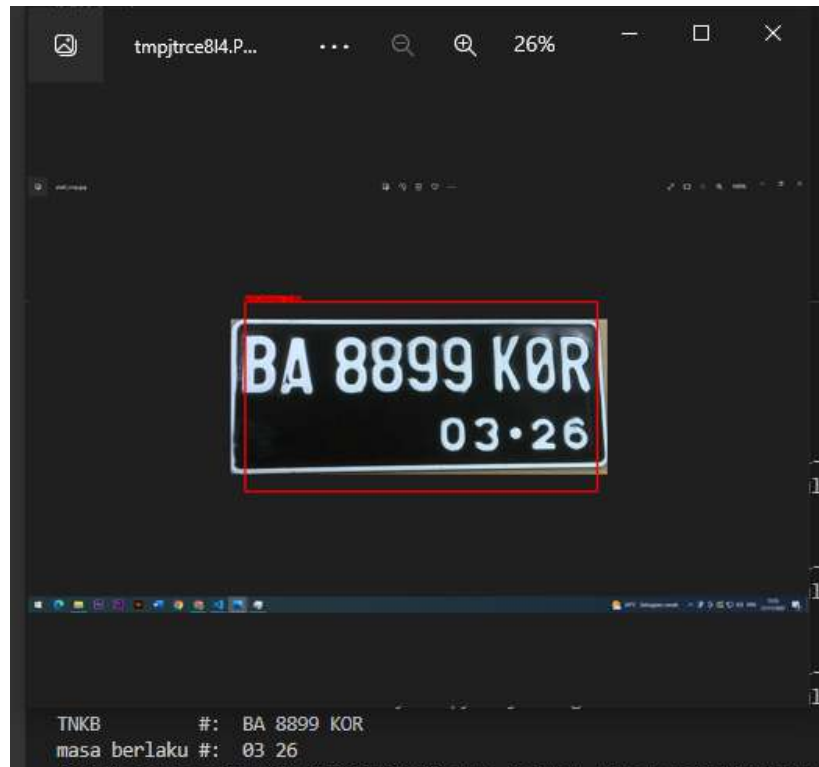
Terdapat satu kesalahan baca pada TNKB, EasyOCR kesulitan membedakan antara huruf “B” dan angka “8”, seluruh nomor masa berlaku salah terbaca karena ukuran yang terlalu kecil.



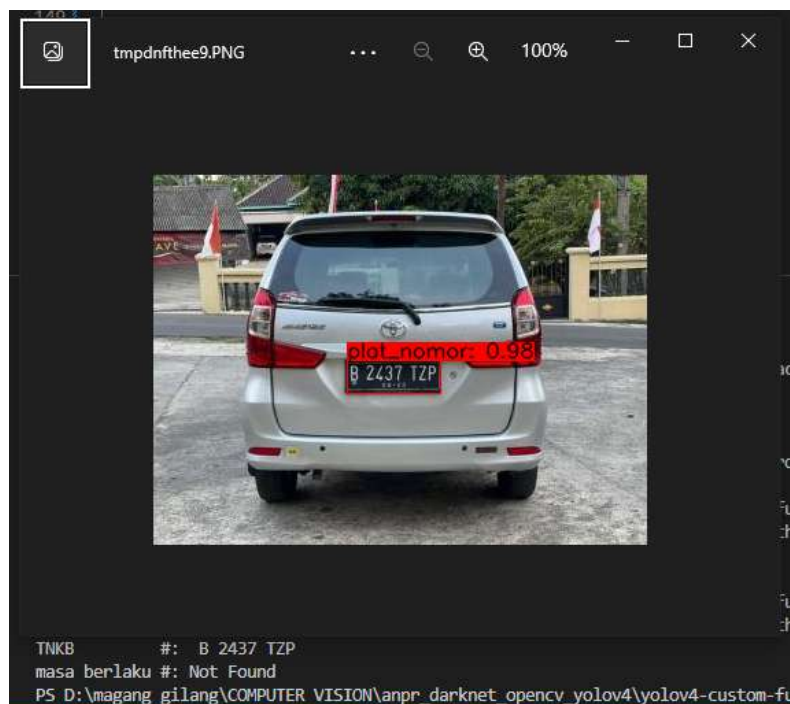
Gambar di atas terbaca dua plat nomor, pertama adalah “**B 108 FIT**” dengan kesalahan huruf “B” terbaca angka “8”, nomor masa berlaku dapat terbaca dengan baik. Plat nomor kedua adalah “**B 307 RMA**” dengan kesalahan huruf “M” menjadi “H”, nomor masa berlaku “04 25” terdapat kesalahan menjadi “06 25”.



TNKB dapat terbaca dengan baik, nomor masa berlaku tidak terdeteksi karena terlalu kecil.



TNKB dan masa berlaku dapat terbaca dengan baik.



TNKB dapat terbaca dengan baik namun masa berlaku tidak terbaca karena terlalu kecil.



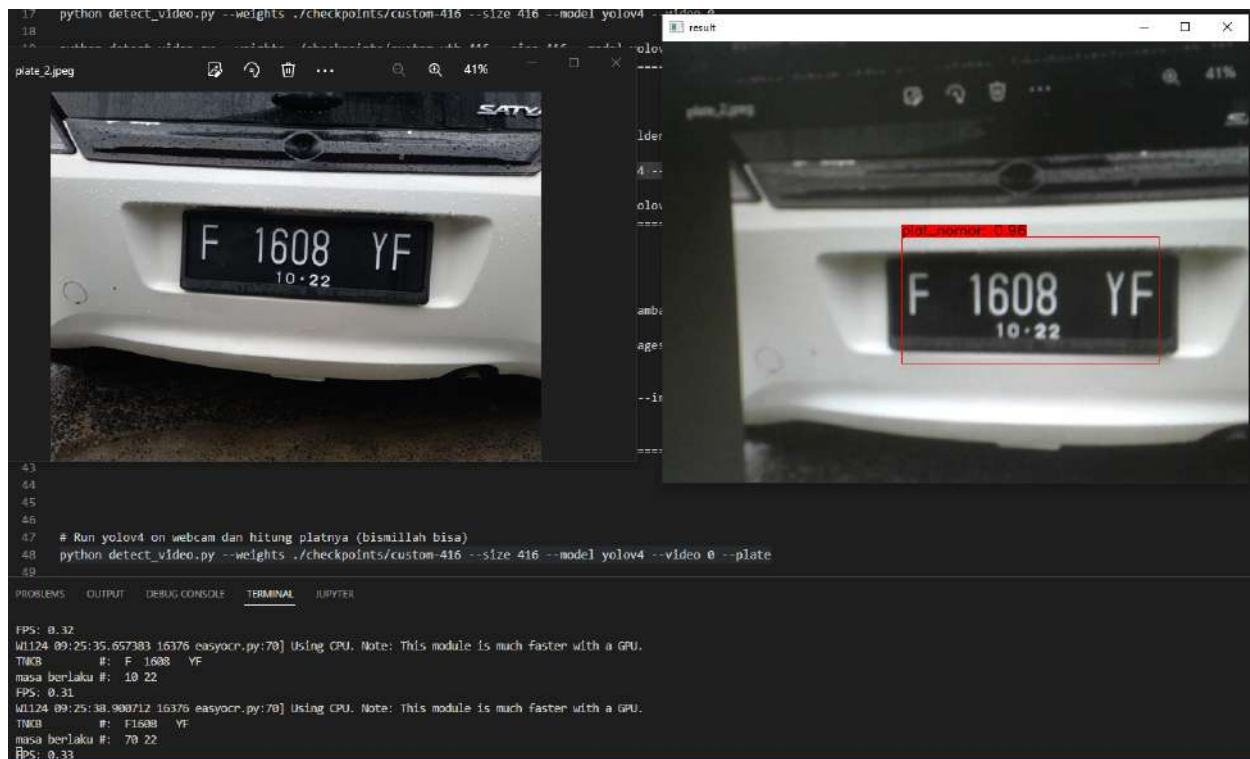
TNKB dan masa berlaku dapat terbaca dengan baik.

18. Percobaan Beberapa Plat Nomor Melalui Kamera

Untuk mengaktifkan deteksi plat nomor dengan kamera, jalankan syntax di bawah ini:

```
“python detect_video.py --weights ./checkpoints/custom-416 --size 416 --model yolov4 --video 0 --plate”
```

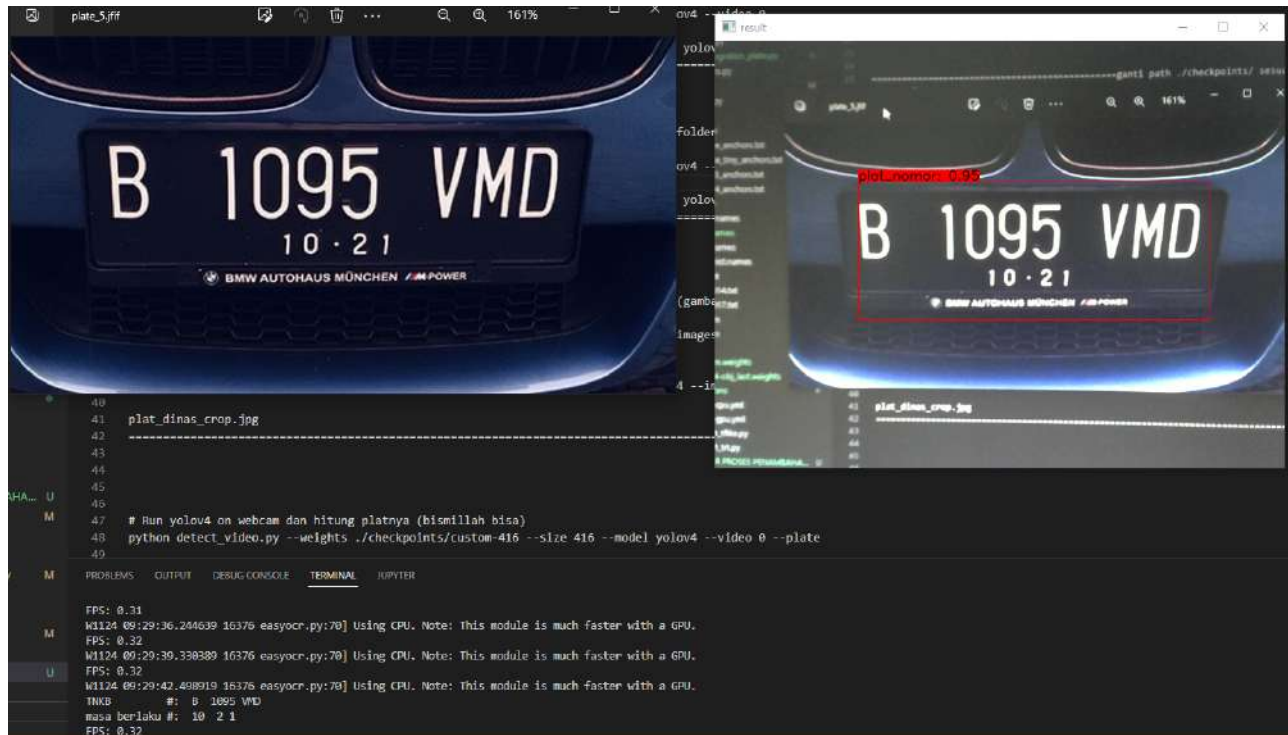
Posisi kamera dan resolusi cukup berpengaruh dalam deteksi karakter secara langsung
Berikut adalah beberapa hasilnya:



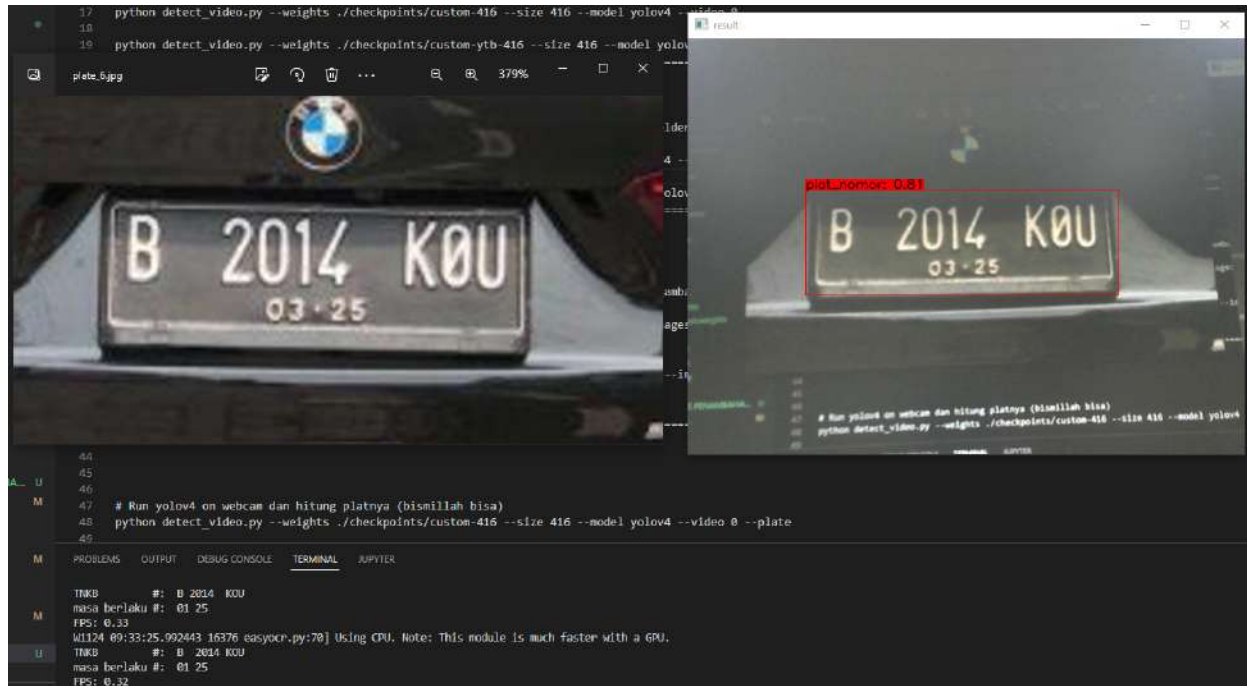
TNKB dan masa berlaku dapat terbaca dengan baik, namun pada frame selanjutnya terjadi kesalahan pada nomor masa berlaku dari “10 22” menjadi “70 22”.



TNKB dan masa berlaku dapat terbaca dengan baik.



TNKB dan masa berlaku dapat terbaca dengan baik.



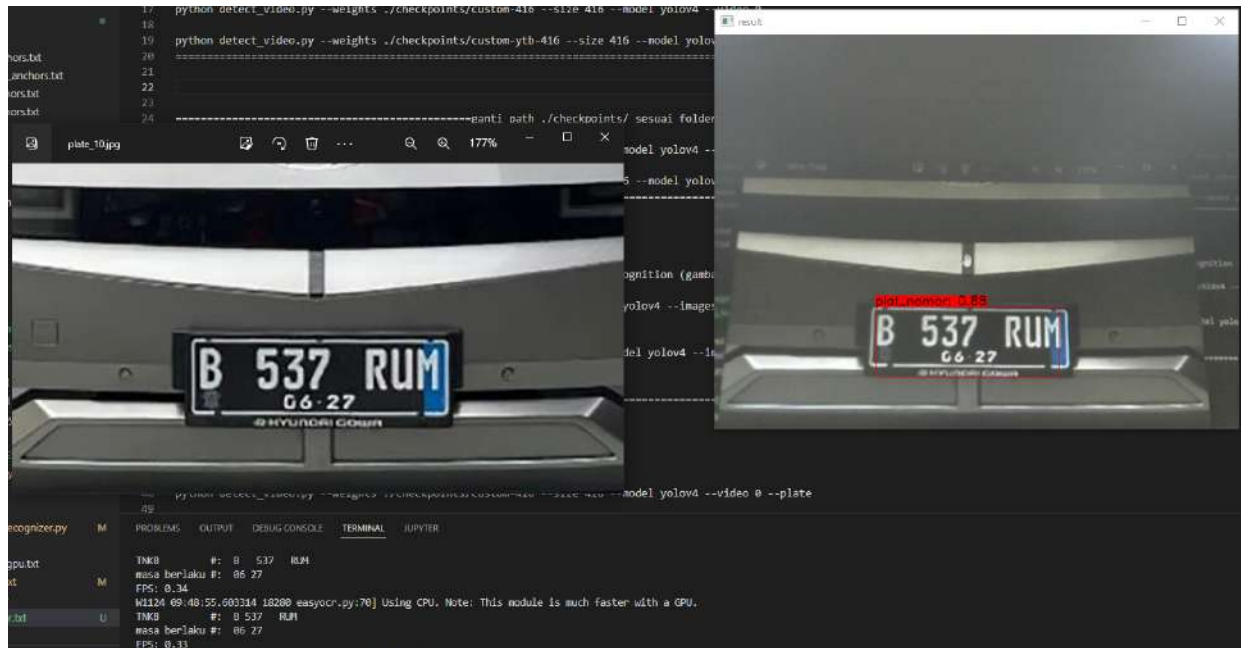
TNKB terbaca dengan baik, masa berlaku pada deteksi pertama dan kedua sama-sama salah dari “03 25” menjadi “01 25” kemungkinan karena angka “3” pada masa berlaku terlalu kecil.



TNKB dapat terbaca dengan baik namun nomor masa berlaku tidak terdeteksi.



Terdapat pada kesalahan pada TNKB dari “B 1050 SPW” menjadi “B 1050 SPH” serta masa berlaku dari “12 25” menjadi “12 28”.



TNKB dan masa berlaku dapat terbaca dengan baik

19. Kesimpulan

Kesimpulan sementara selama pembuatan deteksi plat nomor yaitu TesseractOCR dapat berjalan lebih cepat namun membutuhkan manipulasi gambar plat nomor agar TesseractOCR dapat mengenali karakter, segmentasi karakter dengan OpenCV dapat membuat deteksi karakter menjadi lebih akurat karena hanya karakter dengan ukuran tertentu yang akan dipilih untuk dikenali oleh TesseractOCR. Namun terjadi kendala pada manipulasi gambar dengan OpenCV, saat dijalankan secara langsung dengan kamera tiba-tiba terjadi error yang tidak tentu (terkadang saat plat terdeteksi langsung error, terkadang dapat mendeteksi karakter selama beberapa detik kemudian terjadi error kembali).

Alternatif yang saya gunakan adalah EasyOCR yang tidak perlu menggunakan OpenCV untuk manipulasi gambar, EasyOCR dapat mendeteksi karakter dalam kondisi warna yang beragam namun performanya lebih lambat dibandingkan TesseractOCR, dari tes yang saya lakukan dengan gambar yang sama TesseractOCR mendapatkan waktu deteksi 19,96 detik sedangkan EasyOCR mendapatkan waktu deteksi 26,63 detik, sedangkan tes dengan kamera frame yang didapatkan oleh TesseractOCR adalah 0,6 FPS sedangkan EasyOCR hanya 0,3 FPS. Website dokumentasi EasyOCR menyarankan untuk menggunakan GPU agar performa deteksi karakter dapat lebih cepat, tentu hal ini akan membuat deteksi plat nomor secara live dapat berjalan lebih baik. Selama menggunakan EasyOCR belum terjadi error yang membuat kamera force close. Beberapa hal yang dapat mempengaruhi hasil dari deteksi karakter plat nomor adalah posisi cropping bounding box, jika cropping terlalu jauh dari karakter maka ada kemungkinan karakter di luar plat dapat terdeteksi, namun jika cropping melewati karakter maka OCR dapat salah

membaca karakter tersebut, hal lain yang berpengaruh adalah posisi plat nomor, resolusi atau ketajaman gambar plat nomor, dan tingkat cahaya.