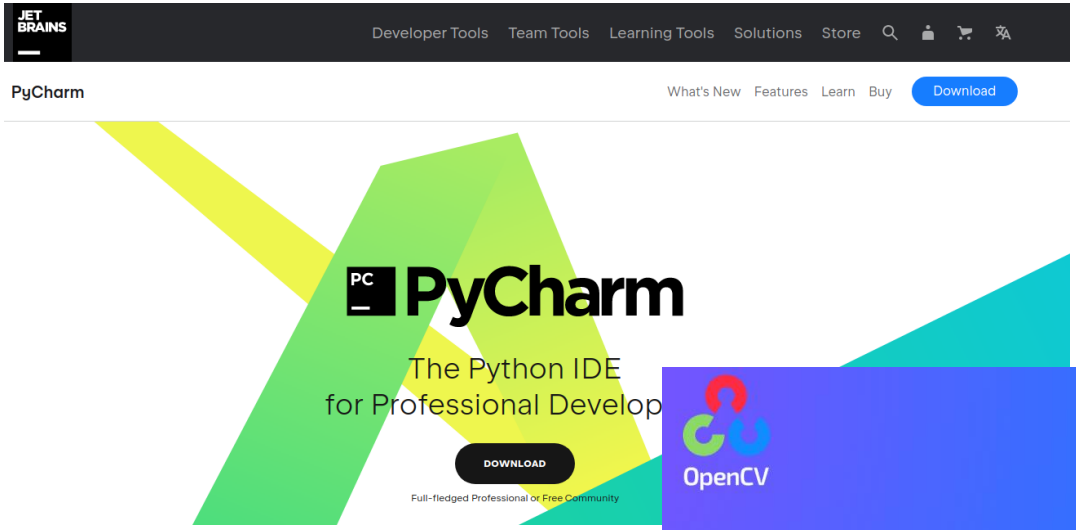
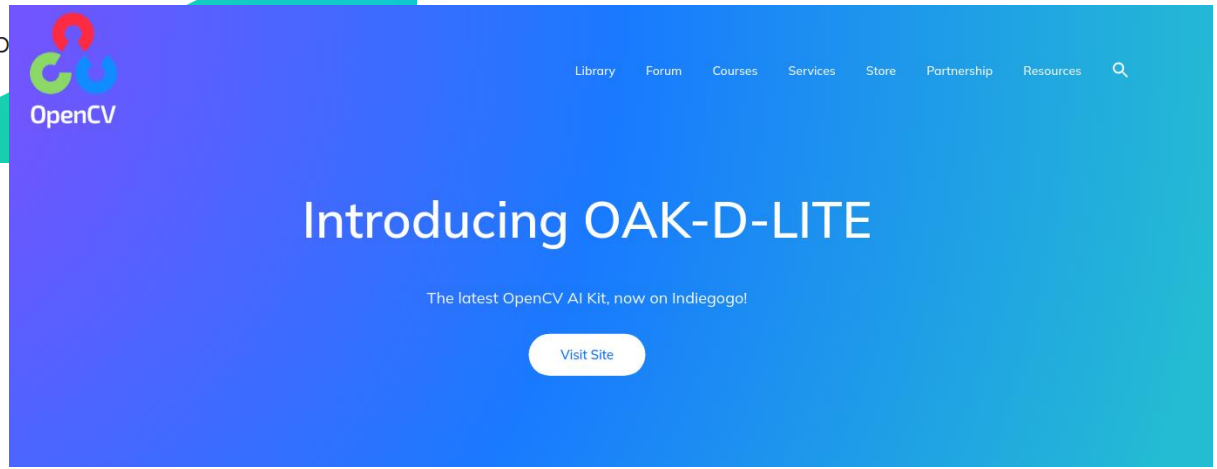
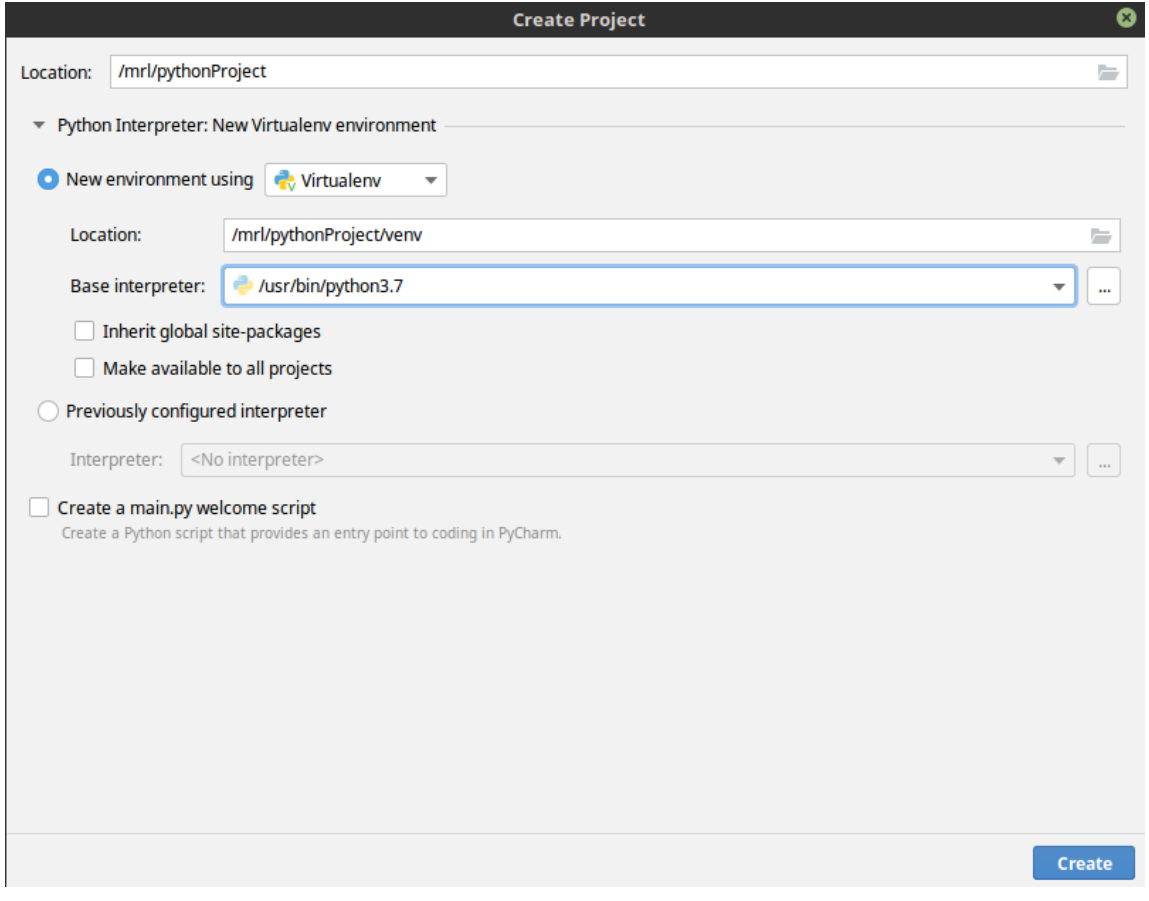
Opencv + Python





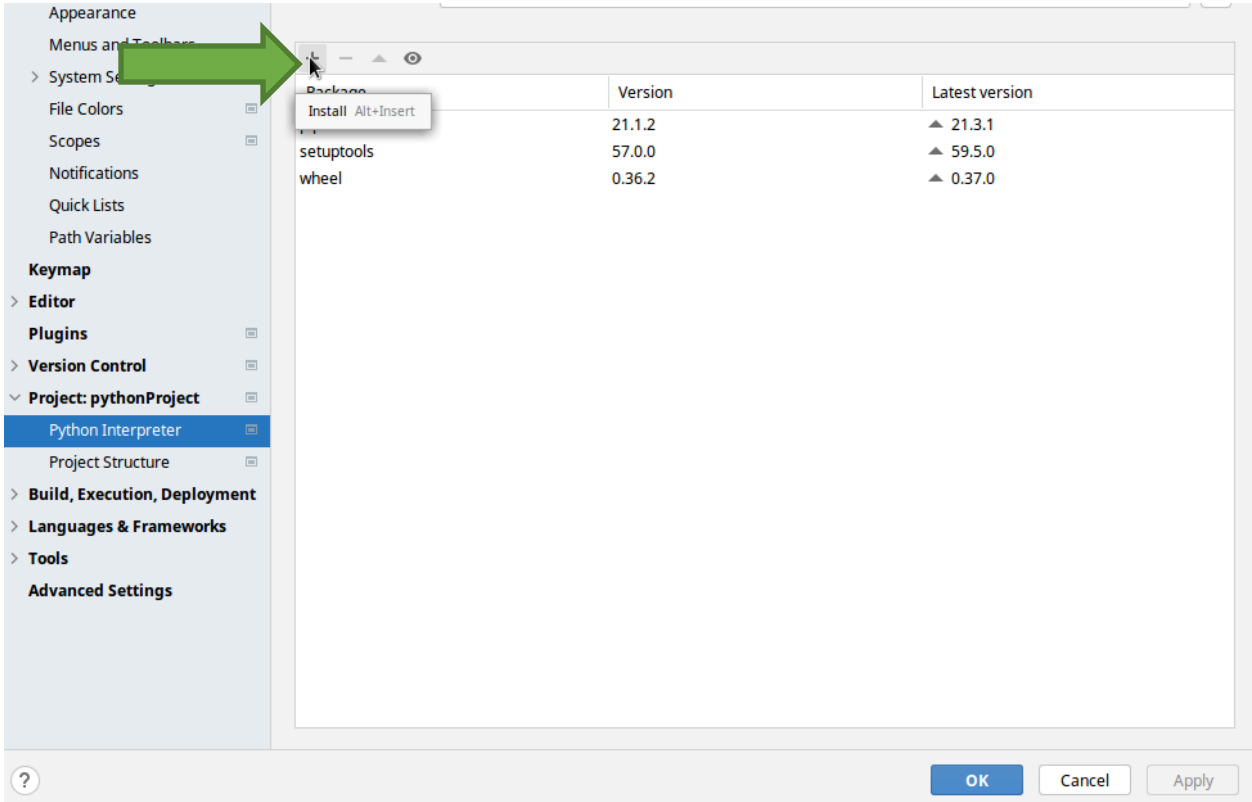
python-3.7.9-amd64
(OK with Dlib)

Opencv + Python

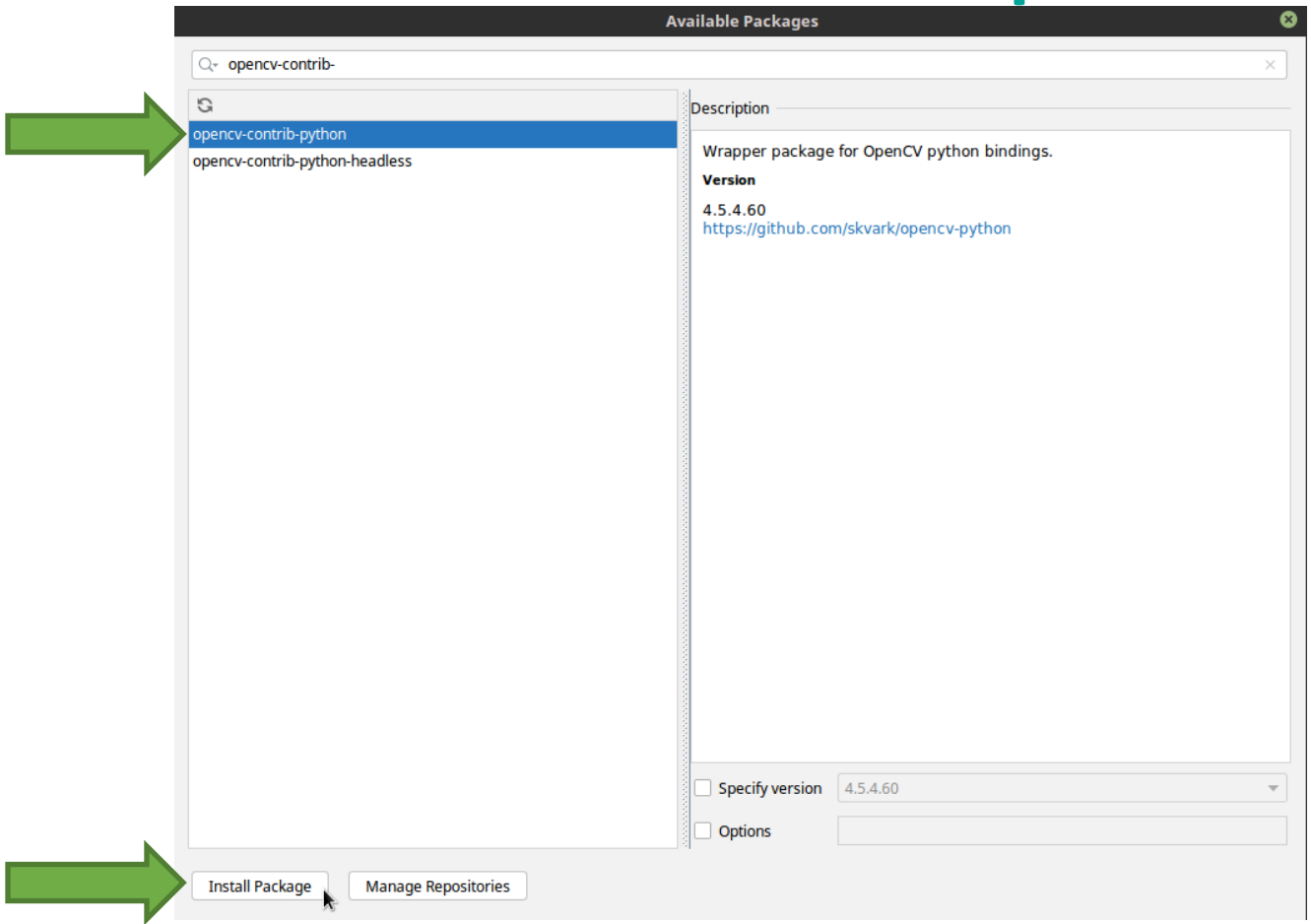


Opencv + Python

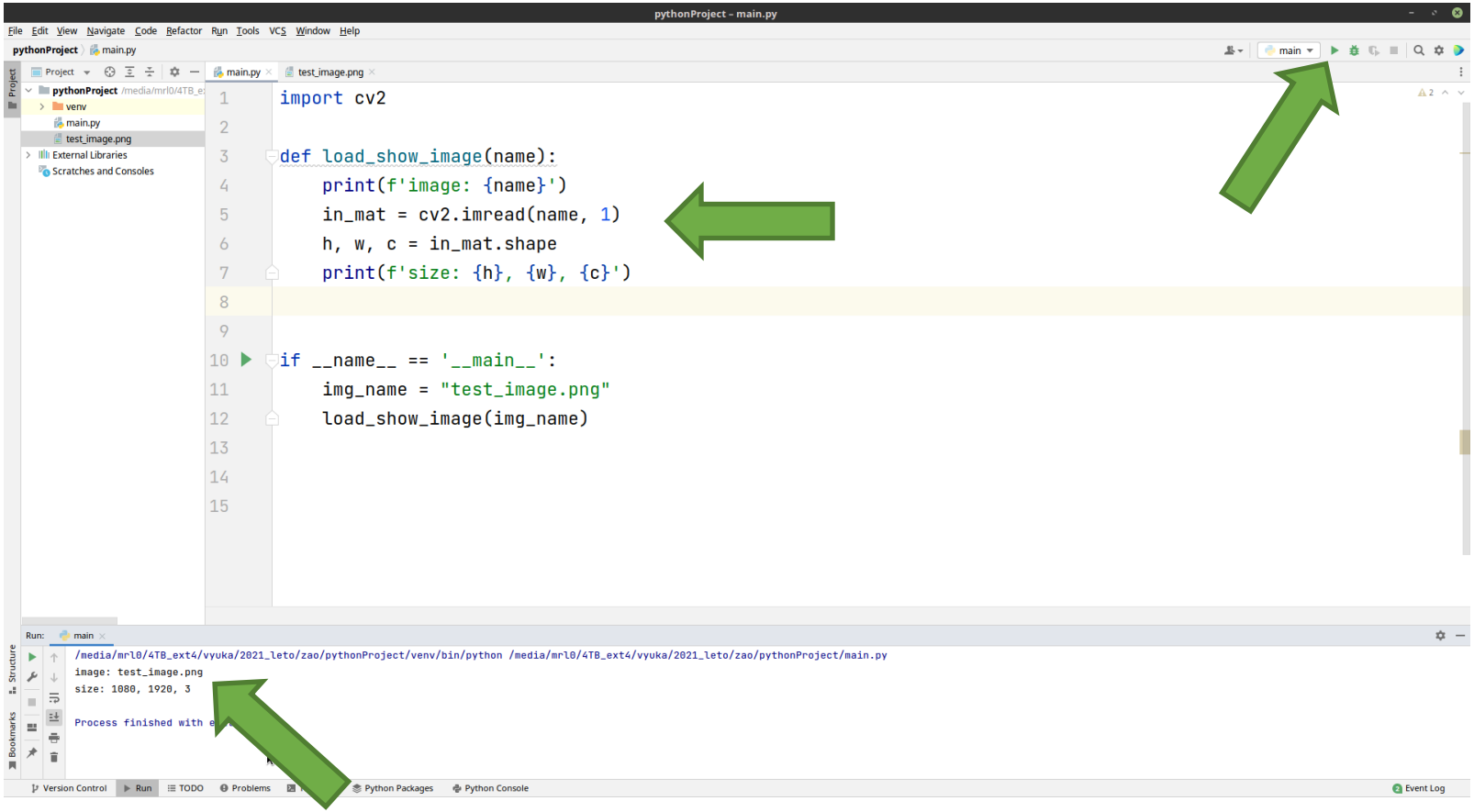
Ctrl+Alt+S



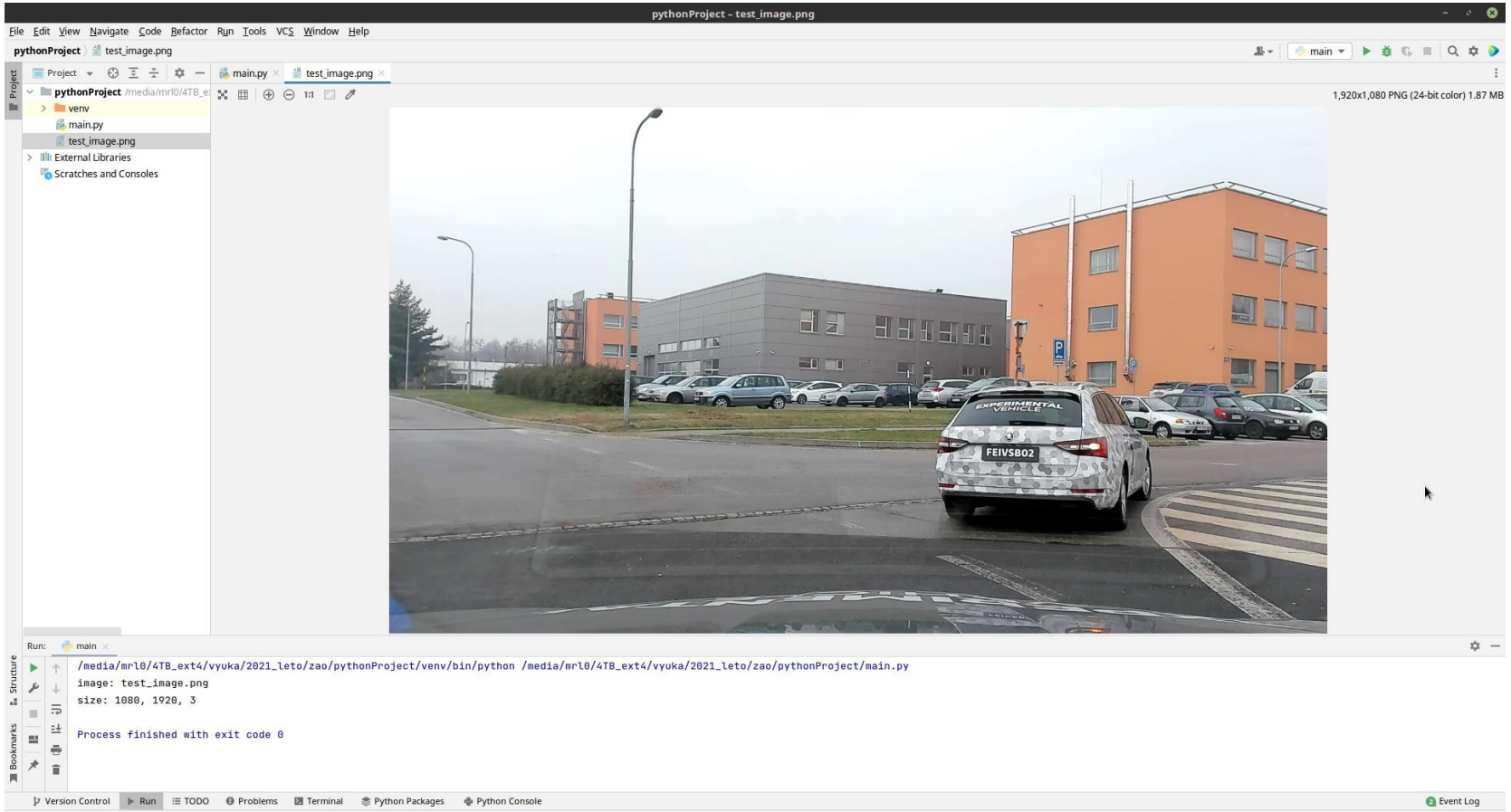
Opencv + Python



Opencv + Python



Opencv + Python



Opencv + Python

pythonProject – main.py

File Edit View Navigate Code Refactor Run Tools VCS Window Help

pythonProject › main.py

Project: pythonProject /media/mrl0/4TB_e…, venv, main.py, test_image.png, External Libraries, Scratches and Consoles

main.py | _init_.py | test_image.png

```python
import cv2


def load_show_image(name):
    print(f'image: {name}')
    in_mat = cv2.imread(name, 1)
    h, w, c = in_mat.shape
    print(f'size: {h}, {w}, {c}')

    win_name = "image"
    cv2.namedWindow(win_name, 0)
    cv2.resizeWindow(win_name, int(w/3), int(h/3))
    cv2.imshow(win_name, in_mat)
    cv2.waitKey(0)


if __name__ == '__main__':
    img_name = "test_image.png"
    load_show_image(img_name)
```

Run: main

```
/media/mrl0/4TB_ext4/vyuka/2021_leto/zao/pythonProject/venv/bin/python /media/mrl0/4TB_ext4/vyuka/2021_leto/zao/pythonProject/main.py
image: test_image.png
size: 1080, 1920, 3
```

Version Control | Run | TODO | Problems | Terminal | Python Packages | Python Console | Event Log

Opencv + Python

pythonProject – main.py

File Edit View Navigate Code Refactor Run Tools VCS Window Help

pythonProject 〉 main.py

Project

pythonProject /media/mrl0/4TB_e

venv

gray_image.png

main.py

test_image.png

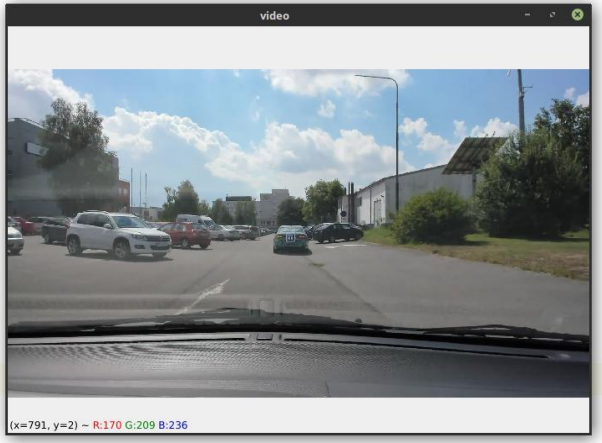
test_video.mp4

External Libraries

Scratches and Consoles

main.py

```
20  def load_show_video(name):
21      video_cap = cv2.VideoCapture(name)
22      win_video = "video"
23      cv2.namedWindow(win_video, 0)
24      while True:
25          ret, frame = video_cap.read()
26          if ret is True:
27              cv2.imshow(win_video, frame)
28              if cv2.waitKey(1) == ord('q'):
29                  break
30
31
32  if __name__ == '__main__':
33      img_name = "test_image.png"
34      vid_name = "test_video.mp4"
35      #load_show_image(img_name)
36      load_show_video(vid_name)
37
38
39
```



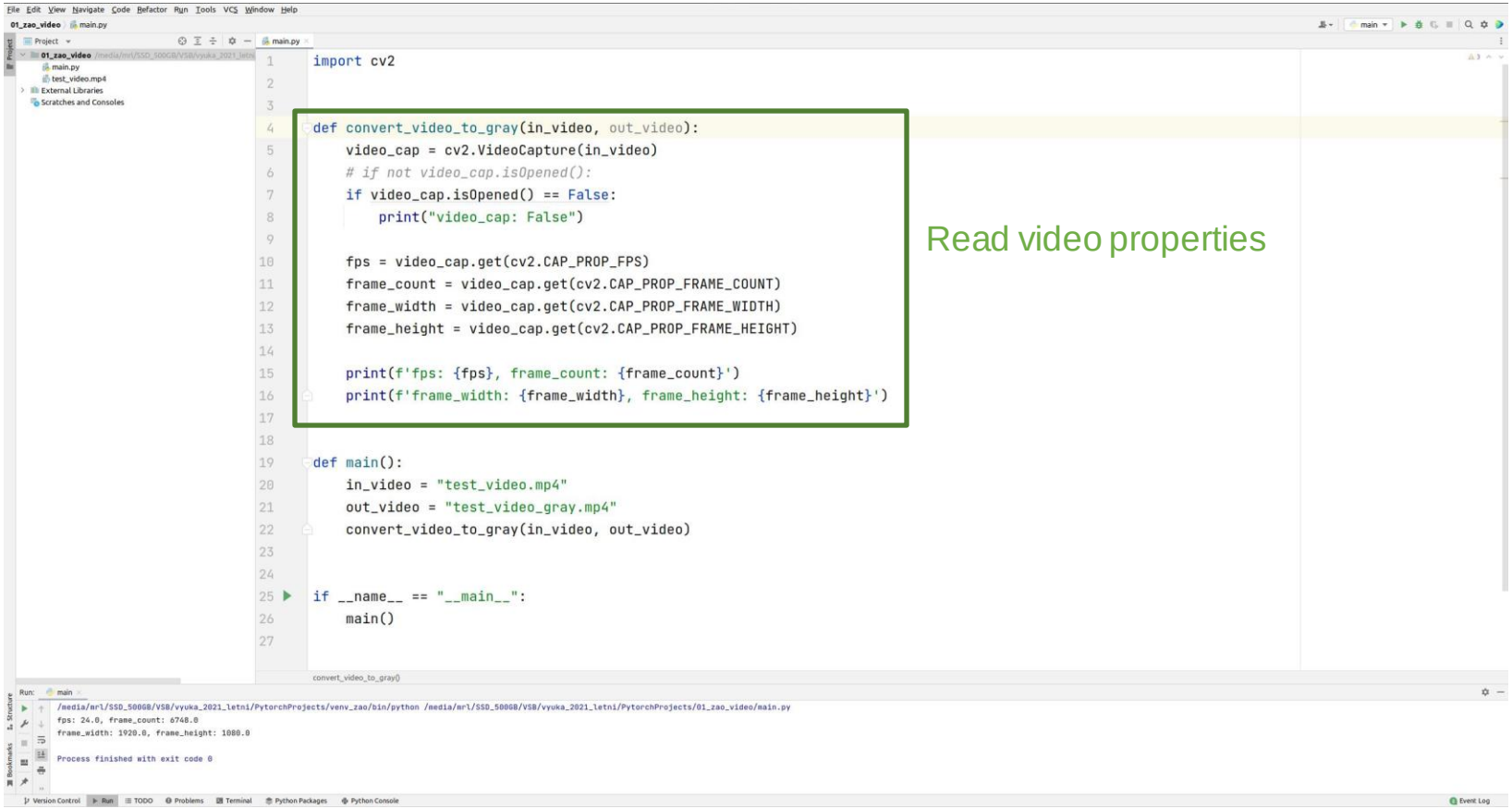
if __name__ == '__main__'

Run: main

```
QApplication: invalid style override 'gtk' passed, ignoring it.
    Available styles: Windows, Fusion
```

Version Control | Run | TODO | Problems | Terminal | Python Packages | Python Console | Event Log

Opencv + Python



Read video properties

Opencv + Python

```python
def convert_video_to_gray(in_video, out_video):
    video_cap = cv2.VideoCapture(in_video)
    # if not video_cap.isOpened():
    if video_cap.isOpened() == False:
        print("video_cap: False")

    fps = video_cap.get(cv2.CAP_PROP_FPS)
    frame_count = video_cap.get(cv2.CAP_PROP_FRAME_COUNT)
    frame_width = video_cap.get(cv2.CAP_PROP_FRAME_WIDTH)
    frame_height = video_cap.get(cv2.CAP_PROP_FRAME_HEIGHT)

    print(f'fps: {fps}, frame_count: {frame_count}')
    print(f'frame_width: {frame_width}, frame_height: {frame_height}')

    frame_size = (int(frame_width), int(frame_height))
    video_writer = cv2.VideoWriter(out_video,
                                   cv2.VideoWriter_fourcc('M', 'J', 'P', 'G'),
                                   fps,
                                   frame_size,
                                   False)
    frame_num = 0
    while (video_cap.isOpened()):
        ret, frame = video_cap.read()
        if ret:
            frame_gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
            video_writer.write(frame_gray)
            frame_num = frame_num + 1
            print("frame_num: ", frame_num)
```

Save video

VideoWriter() [2/3]

```
cv::VideoWriter::VideoWriter ( const String &  filename,
                               int             fourcc,
                               double          fps,
                               Size            frameSize,
                               bool            isColor = true
                             )
```

Python:

```
cv.VideoWriter(                                                    ) -> <VideoWriter object>
cv.VideoWriter( filename, fourcc, fps, frameSize[, isColor]        ) -> <VideoWriter object>
cv.VideoWriter( filename, apiPreference, fourcc, fps, frameSize[, isColor] ) -> <VideoWriter object>
```

This is an overloaded member function, provided for convenience. It differs from the above function only in what argument(s) it accepts.

Parameters

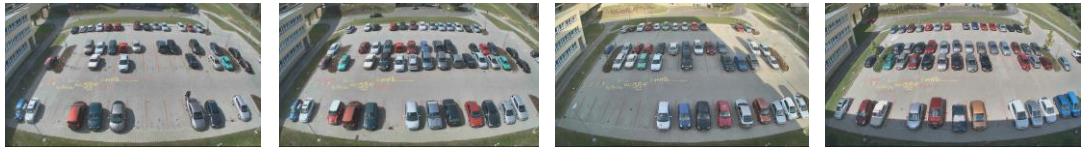
filename	Name of the output video file.
fourcc	4-character code of codec used to compress the frames. For example, **VideoWriter::fourcc**('P','I','M','1') is a MPEG-1 codec, **VideoWriter::fourcc**('M','J','P','G') is a motion-jpeg codec etc. List of codes can be obtained at Video Codecs by FOURCC page. FFMPEG backend with MP4 container natively uses other values as fourcc code: see ObjectType, so you may receive a warning message from OpenCV about fourcc code conversion.
fps	Framerate of the created video stream.
frameSize	Size of the video frames.
isColor	If it is not zero, the encoder will expect and encode color frames, otherwise it will work with grayscale frames (the flag is currently supported on Windows only).

Tips:

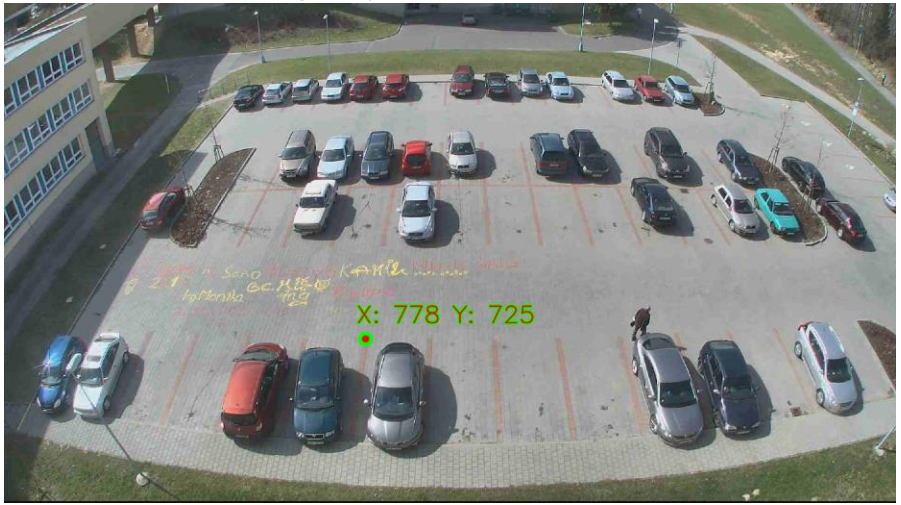
- With some backends `fourcc=-1` pops up the codec selection dialog from the system.
- To save image sequence use a proper filename (eg. `img_%02d.jpg`) and `fourcc=0` OR `fps=0`. Use uncompressed image format (eg. `img_%02d.BMP`) to save raw frames.
- Most codecs are lossy. If you want lossless video file you need to use a lossless codecs (eg. FFMPEG FFV1, Huffman HFYU, Lagarith LAGS, etc...)
- If FFMPEG is enabled, using `codec=0; fps=0;` you can create an uncompressed (raw) video file.

Exercise – 01

1. (1b) Read all parking images from folder using OpenCV
http://mrl.cs.vsb.cz/data/vyuka/zao/parking_images_cv01.zip

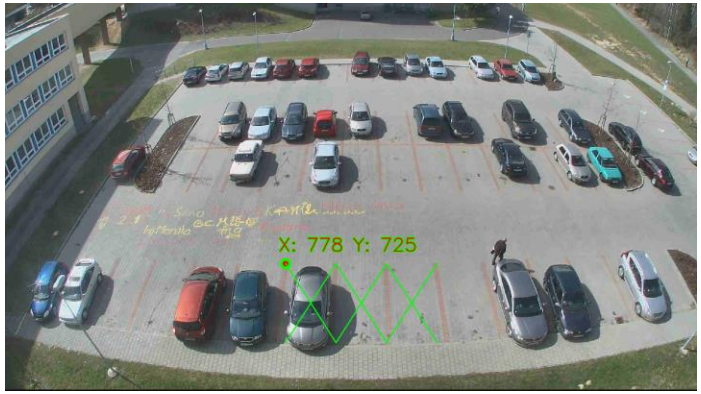


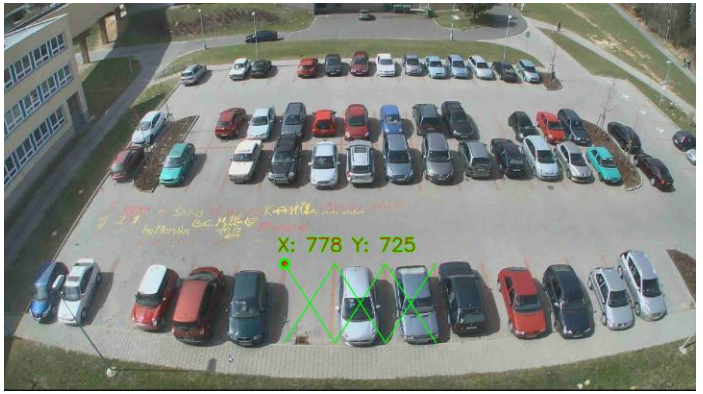
2. (2b) In each image, extract three parking spaces (from first row), starting point is x1 = 778, y1 = 725
set size of each extracted parking space to w = 140, h = 220
this operation should be created using loops

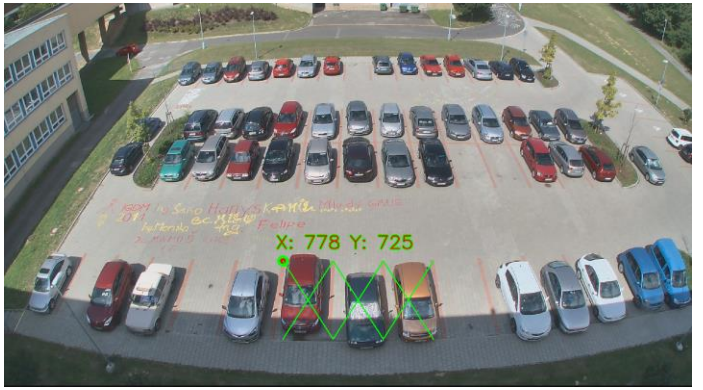


Exercise - 01

Visualization of the particular spaces is shown in following images:

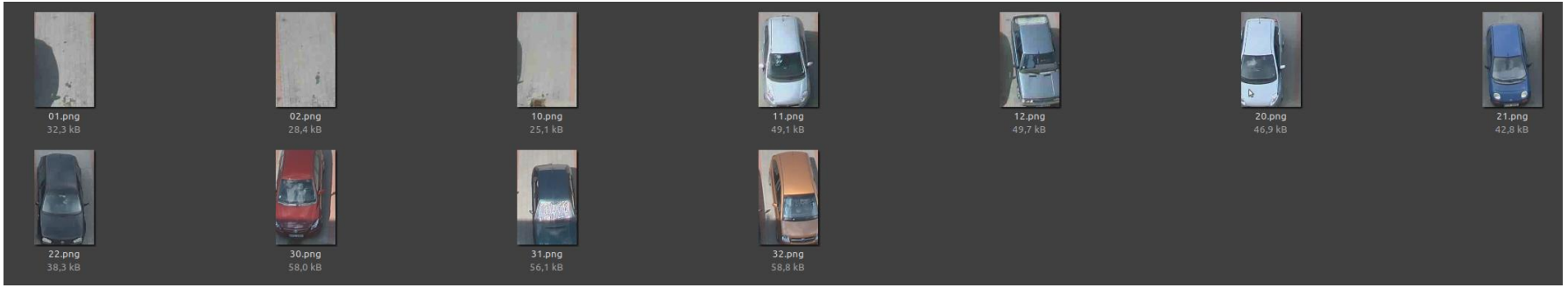






Exercise - 01

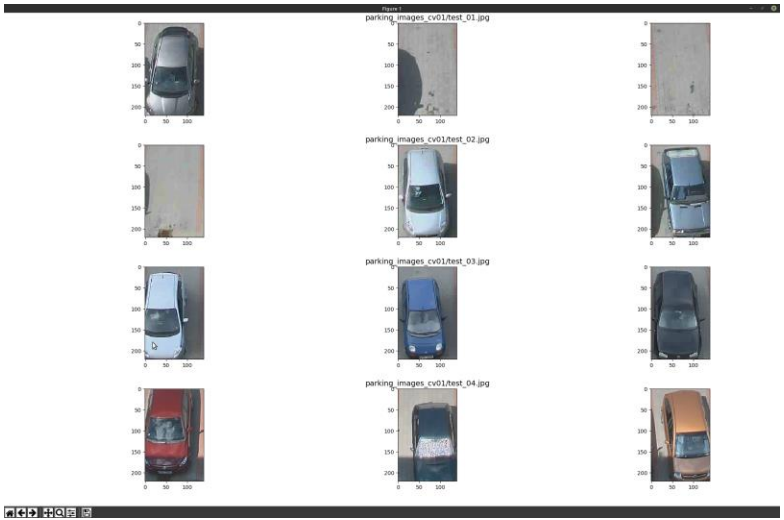
3. (1b) save all extracted images into new folder "cropped"



HINT:
one loop - over images in folder
second loop - over parking places (increment the appropriate coordinate)

Exercise - 01

4. (1b) BONUS – show all extracted parking places in one image (e.g. using subplots via matplotlib) + create a visualization (e.g. using green lines) of these parking spaces



matplotlib

visualization