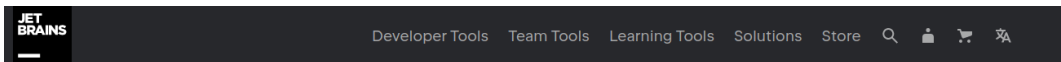


Opencv + Python



PyCharm [What's New](#) [Features](#) [Learn](#) [Buy](#) [Download](#)

PyCharm

The Python IDE
for Professional Developers

DOWNLOAD

Full-fledged Professional or Free Community



[Library](#) [Forum](#) [Courses](#) [Services](#) [Store](#) [Partnership](#) [Resources](#) [Search](#)

Introducing OAK-D-LITE

The latest OpenCV AI Kit, now on Indiegogo!

[Visit Site](#)

python-3.7.9-amd64
(OK with Dlib)

Opencv + Python

Create Project

Location: /mrl/pythonProject

Python Interpreter: New Virtualenv environment

☒ New environment using

Virtualenv

Location: /mrl/pythonProject/venv

Base interpreter: /usr/bin/python3.7

...

☐ Inherit global site-packages
☐ Make available to all projects

☐ Previously configured interpreter

Interpreter: <No interpreter>

...

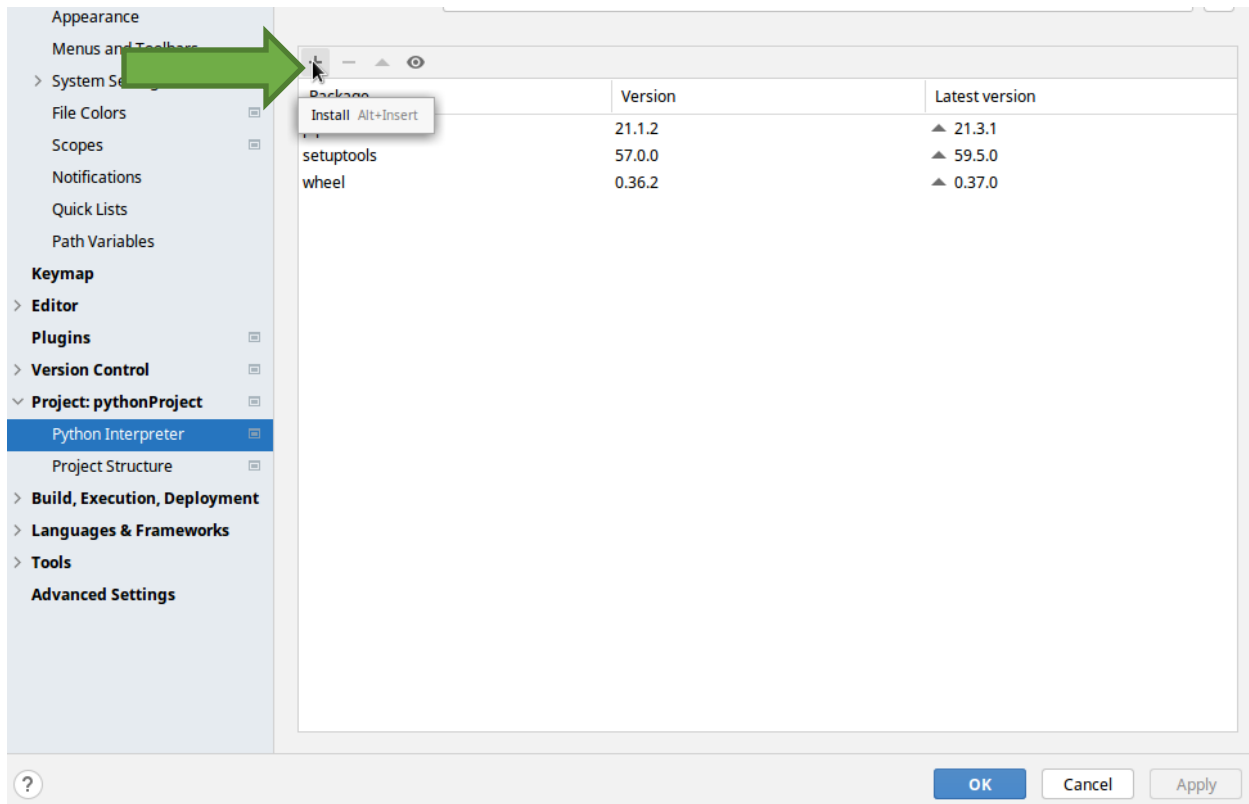
☐ Create a main.py welcome script
Create a Python script that provides an entry point to coding in PyCharm.

Create

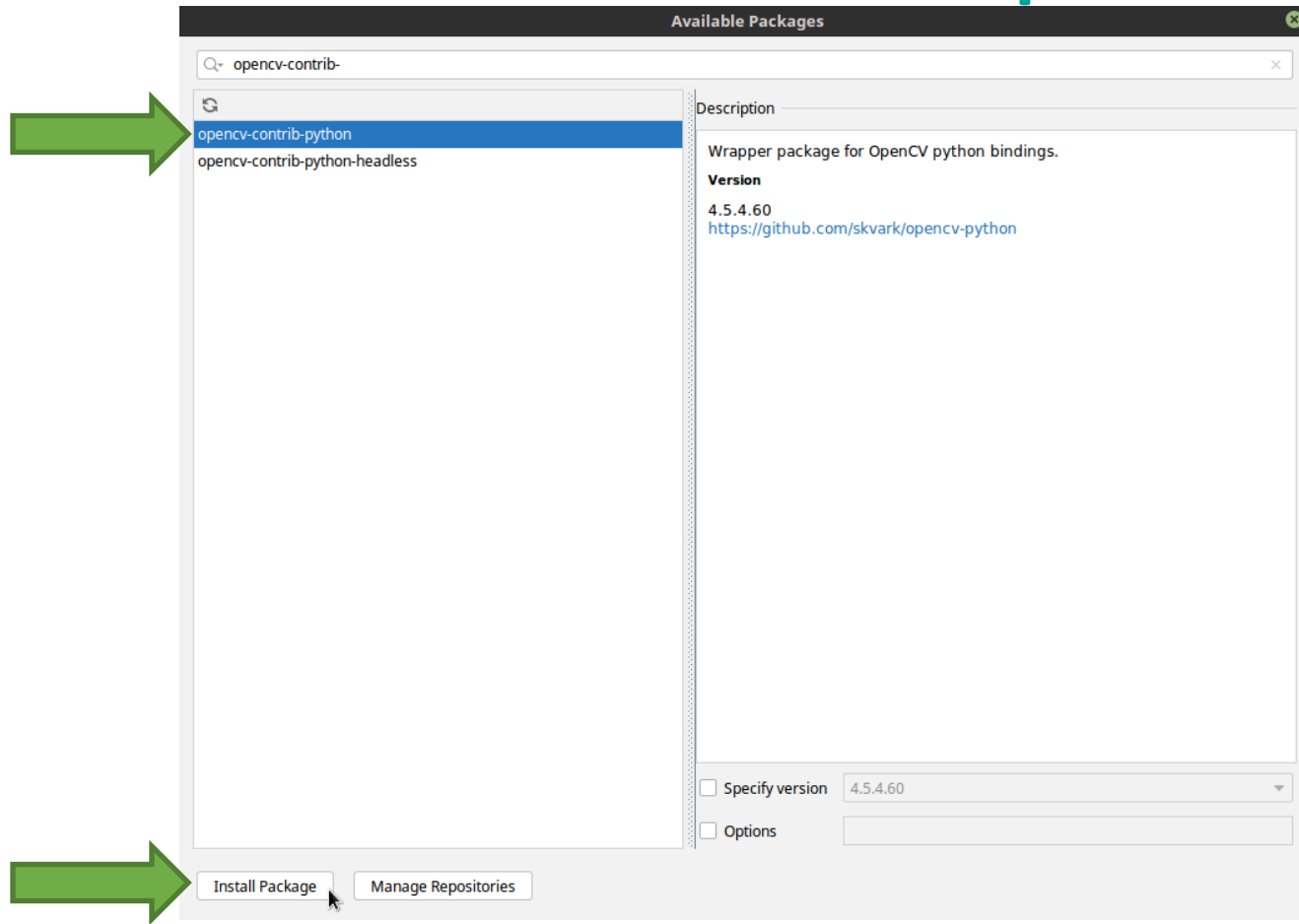
27

Opencv + Python

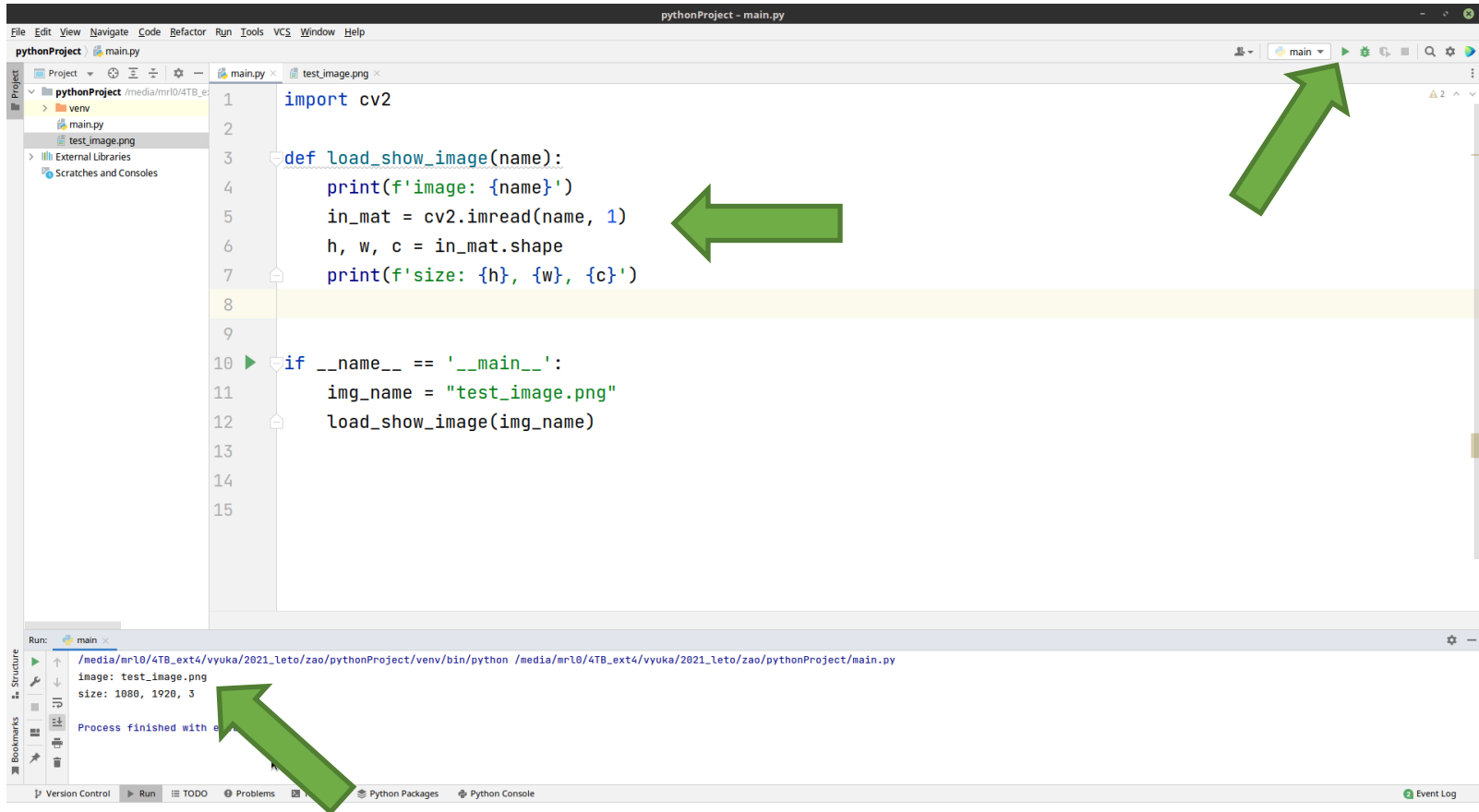
Ctrl+Alt+S



Opencv + Python



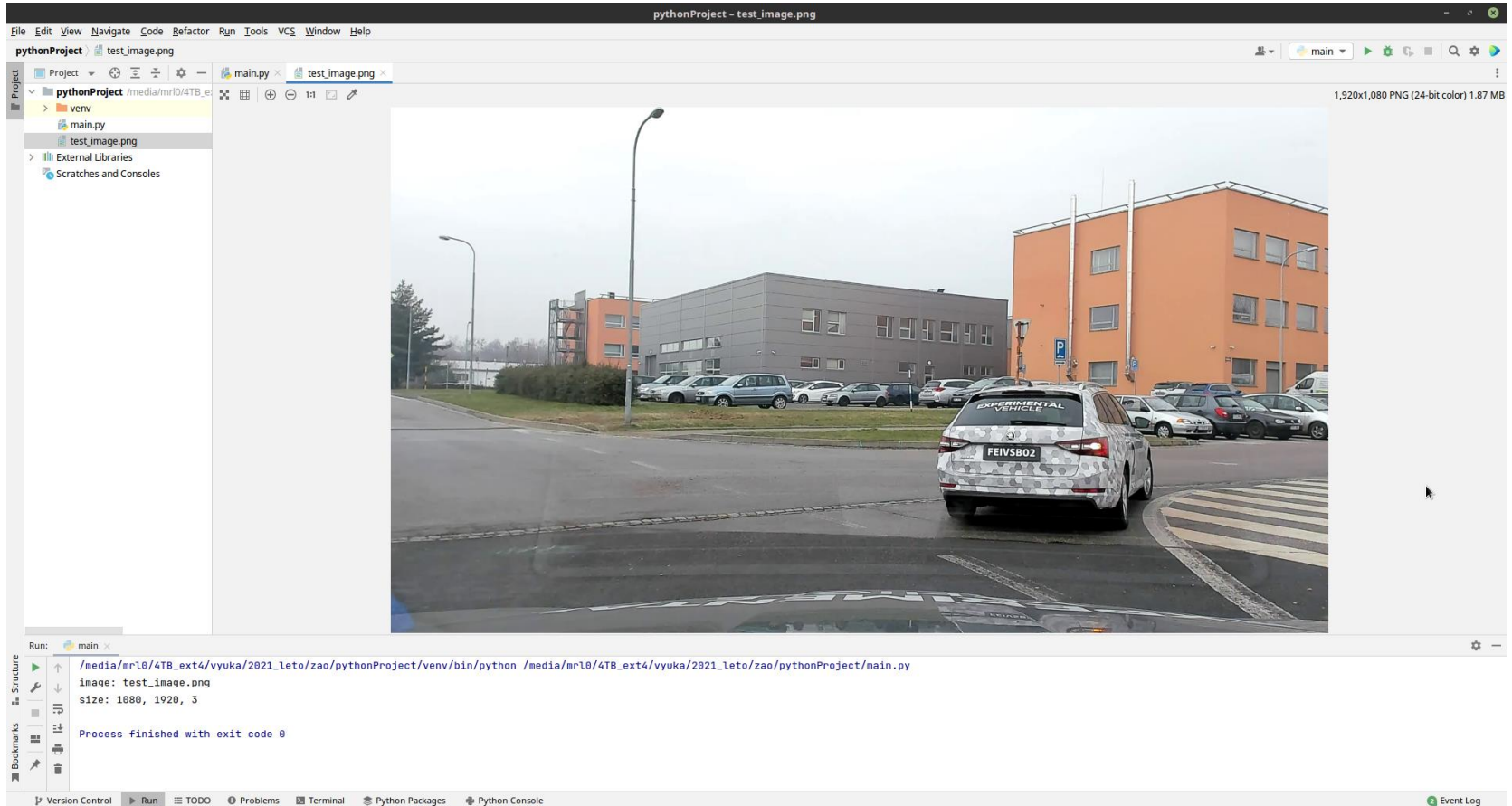
Opencv + Python



```

pythonProject - main.py
File Edit View Navigate Code Refactor Run Tools VCS Window Help
pythonProject /media/mr10/4TB_e
main.py test_image.png
Project
  pythonProject /media/mr10/4TB_e
    > venv
    main.py
    test_image.png
  External Libraries
  Scratches and Consoles
1 import cv2
2
3 def load_show_image(name):
4     print(f'image: {name}')
5     in_mat = cv2.imread(name, 1)
6     h, w, c = in_mat.shape
7     print(f'size: {h}, {w}, {c}')
8
9
10 if __name__ == '__main__':
11     img_name = "test_image.png"
12     load_show_image(img_name)
13
14
15
Run: main
  /media/mr10/4TB_ext4/vyuka/2021_let0/zao/pythonProject/venv/bin/python /media/mr10/4TB_ext4/vyuka/2021_let0/zao/pythonProject/main.py
  image: test_image.png
  size: 1080, 1920, 3
  Process finished with e
  Version Control Run TODO Problems Python Packages Python Console Event Log
  
```

Opencv + Python



Opencv + Python

```

1 import cv2
2
3
4 def load_show_image(name):
5     print(f'image: {name}')
6     in_mat = cv2.imread(name, 1)
7     h, w, c = in_mat.shape
8     print(f'size: {h}, {w}, {c}')
9
10     win_name = "image"
11     cv2.namedWindow(win_name, 0)
12     cv2.resizeWindow(win_name, int(w/3), int(h/3))
13     cv2.imshow(win_name, in_mat)
14     cv2.waitKey(0)
15
16
17 if __name__ == '__main__':
18     img_name = "test_image.png"
19     load_show_image(img_name)
20
21

```

Run: main

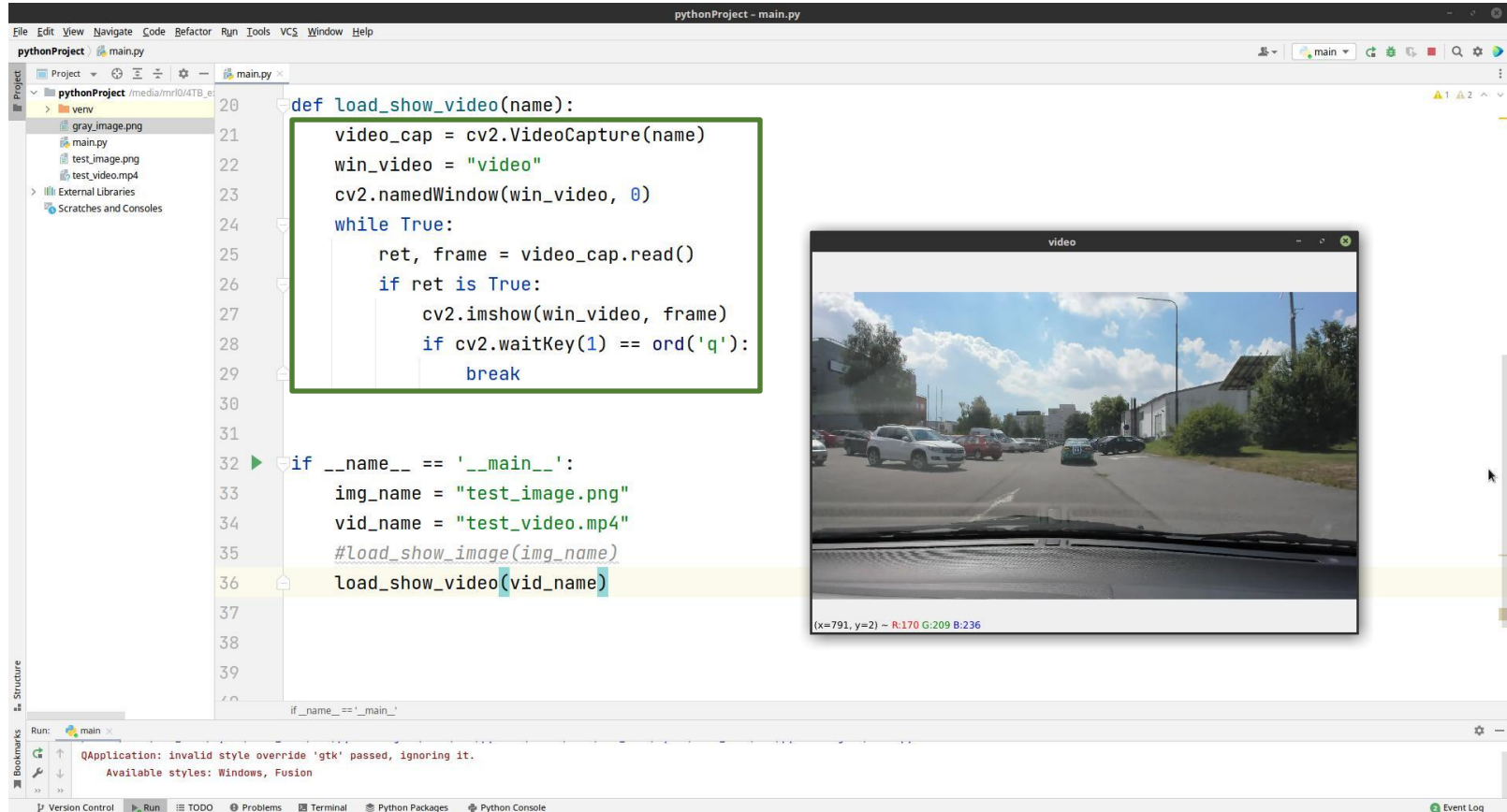
```

/media/mr10/4TB_ext4/vyuka/2021_let0/zao/pythonProject/venv/bin/python /media/mr10/4TB_ext4/vyuka/2021_let0/zao/pythonProject/main.py
image: test_image.png
size: 1080, 1920, 3

```

Event Log

Opencv + Python



The image shows a screenshot of an IDE (likely PyCharm) with a Python project named 'pythonProject'. The main file, 'main.py', contains the following code:

```

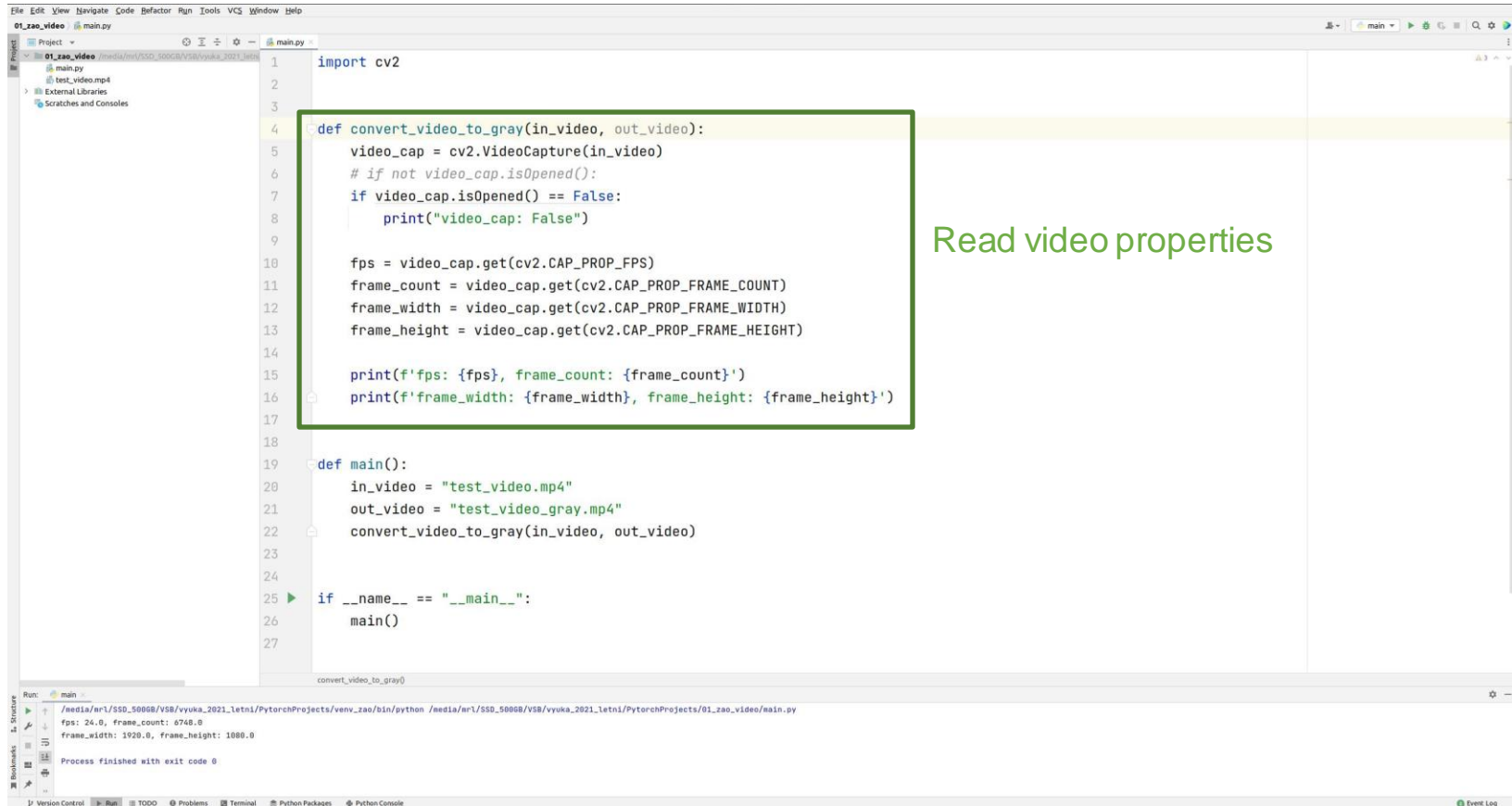
20 def load_show_video(name):
21     video_cap = cv2.VideoCapture(name)
22     win_video = "video"
23     cv2.namedWindow(win_video, 0)
24     while True:
25         ret, frame = video_cap.read()
26         if ret is True:
27             cv2.imshow(win_video, frame)
28             if cv2.waitKey(1) == 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
40 if __name__ == '__main__':

```

The code is highlighted with a green box around the `load_show_video` function and a yellow box around the `if __name__ == '__main__':` block. A window titled 'video' is open, displaying a frame from a video showing a street scene with cars and buildings. The window has a status bar at the bottom showing coordinates and color values: `(x=791, y=2) ~ R:170 G:209 B:236`.

The IDE interface includes a Project view on the left showing the file structure, a Run view at the bottom showing the execution of the 'main' module, and a status bar at the very bottom with various tool icons.

Opencv + Python



```
1 import cv2
2
3
4 def convert_video_to_gray(in_video, out_video):
5     video_cap = cv2.VideoCapture(in_video)
6     # if not video_cap.isOpened():
7     if video_cap.isOpened() == False:
8         print("video_cap: False")
9
10    fps = video_cap.get(cv2.CAP_PROP_FPS)
11    frame_count = video_cap.get(cv2.CAP_PROP_FRAME_COUNT)
12    frame_width = video_cap.get(cv2.CAP_PROP_FRAME_WIDTH)
13    frame_height = video_cap.get(cv2.CAP_PROP_FRAME_HEIGHT)
14
15    print(f'fps: {fps}, frame_count: {frame_count}')
16    print(f'frame_width: {frame_width}, frame_height: {frame_height}')
17
18
19 def main():
20     in_video = "test_video.mp4"
21     out_video = "test_video_gray.mp4"
22     convert_video_to_gray(in_video, out_video)
23
24
25 if __name__ == "__main__":
26     main()
27
```

Read video properties

Run: main

/media/nr1/SSD_500GB/VSB/vyuka_2021/letni/PytorchProjects/venv_zao/bin/python /media/nr1/SSD_500GB/VSB/vyuka_2021/letni/PytorchProjects/01_zao_video/main.py

fps: 24.0, frame_count: 4748.0
frame_width: 1920.0, frame_height: 1080.0

Process finished with exit code 0

Opencv + Python

```

4 def convert_video_to_gray(in_video, out_video):
5     video_cap = cv2.VideoCapture(in_video)
6     # if not video_cap.isOpened():
7     if video_cap.isOpened() == False:
8         print("video_cap: False")
9
10    fps = video_cap.get(cv2.CAP_PROP_FPS)
11    frame_count = video_cap.get(cv2.CAP_PROP_FRAME_COUNT)
12    frame_width = video_cap.get(cv2.CAP_PROP_FRAME_WIDTH)
13    frame_height = video_cap.get(cv2.CAP_PROP_FRAME_HEIGHT)
14
15    print(f'fps: {fps}, frame_count: {frame_count}')
16    print(f'frame_width: {frame_width}, frame_height: {frame_height}')
17
18    frame_size = (int(frame_width), int(frame_height))
19    video_writer = cv2.VideoWriter(out_video,
20                                  cv2.VideoWriter_fourcc('M', 'J', 'P', 'G'),
21                                  fps,
22                                  frame_size,
23                                  False)
24
25    frame_num = 0
26    while (video_cap.isOpened()):
27        ret, frame = video_cap.read()
28        if ret:
29            frame_gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
30            video_writer.write(frame_gray)
31            frame_num = frame_num + 1
32            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).

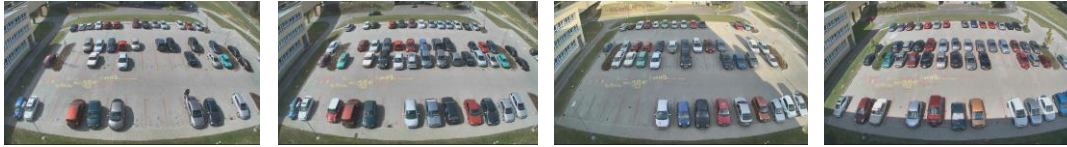
Tips:

- 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 FV1, 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

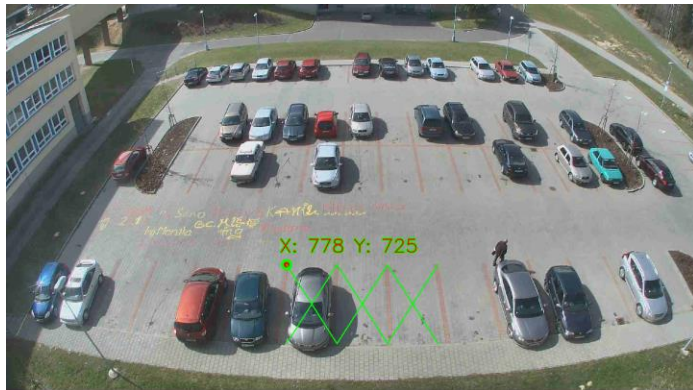


2. (2b) In each image, extract three parking spaces (from first row), starting point is $x1 = 778, y1 = 725$
set size of each extracted parking space to $w = 140, h = 220$
this operation should be created using loops



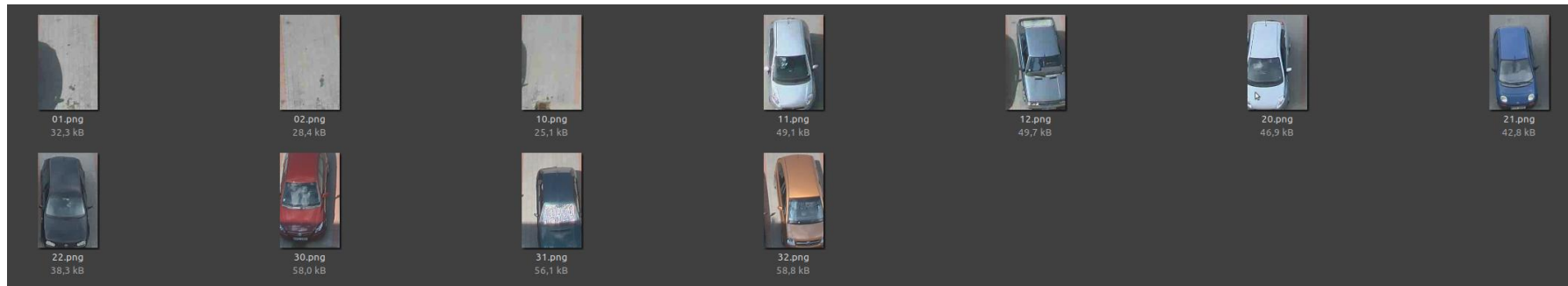
Exercise - 01

Visualization of the particular spaces is shown in following images:



Exercise - 01

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



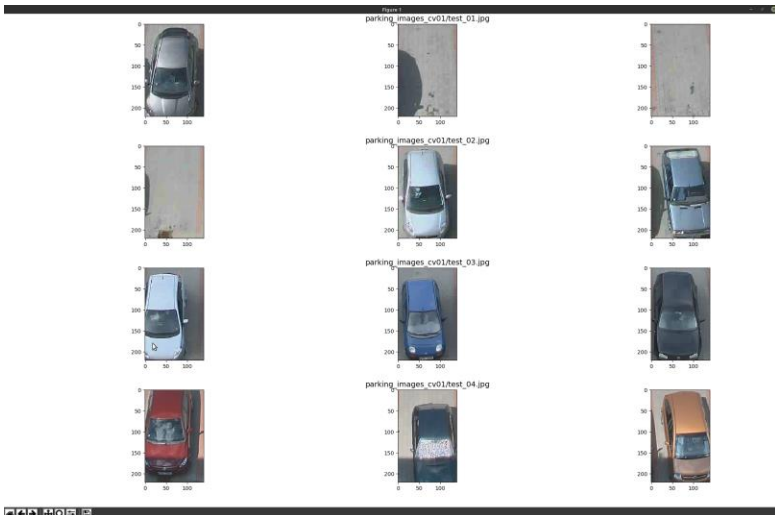
HINT:

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