User manual to running the program for Windows 10 users

1. Installing Visual Studio

We can write and test our scripts in the Visual Studio integrated development environment (IDE). You can download the latest version from Microsoft's website (Visual Studio 2019, https://visualstudio.microsoft.com/downloads/). On my computer I already had Visual Studio 2017 installed, so I used this version.

2. Installing OpenCV

OpenCV is an open-source software library which contains some algorithms and functions used for computer vision and machine learning. On their website (link: https://opencv.org/releases/) you can find many versions available (the newest one at the moment is 4.3.0), for making the project work you must download at least version 3.4.2 (OpenCV supports the YOLOv3 since this version). For this project I used OpenCV 3.4.9.

The steps of the installation are the following:

I. From https://opencv.org/releases/ select your preferred version of OpenCV (as I mentioned, at least version 3.4.2) (Windows). The download starts automatically.

II. Create a new folder on the C: drive called *"C:\OpenCV-X.X.X"* where X.X.X indicates the number of version (for example: *"C:\OpenCV-3.4.9"*). Run the downloaded installation file and at the "Extract to" option select the path to the created folder.

III. When the installation finished, add the *bin* directory corresponding to the version of Visual Studio and OpenCV to the path of the operation system. The format of the path to the *bin* directory is the following: *"C:\OpenCV-X.X.X\opencv\build\x64\vcYY\bin",* where X.X.X once again indicates the version of OpenCV, and vcYY indicates the version of Visual Studio (for Visual Studio 2015: vc14, for Visual Studio 2017: vc15, for Visual Studio 2019: vc16). For example for OpenCV 3.4.9 and Visual Studio 2017 the path is *"C:\OpenCV-3.4.9\opencv\build\x64\vc15\bin".*
(How to add the *bin* directory to the Path in Windows 10: open Control Panel → System → Advanced system settings → Environment variables.. → In the System variables window click on line Path → Edit.. → New → Type in the path for the *bin* directory, then OK for everything.)

3. Downloading the program files

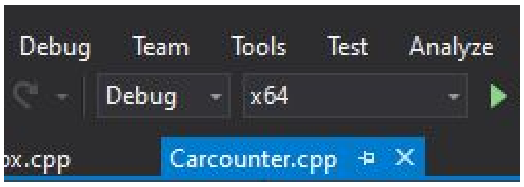
You can download the program files from **https://github.com/MartonPolcz/Cpp-CarCounting-with-YOLOv3** . The required files are the following: **BoundingBox.h, BoundingBox.cpp, Carcounter.cpp**. The input video file is **traffic.mp4**. You can download YOLOv3's names, config and weights files from the following links:

- **coco.names**: https://github.com/pjreddie/darknet/blob/master/data/coco.names
- **yolov3-tiny.cfg**: https://github.com/pjreddie/darknet/blob/master/cfg/yolov3-tiny.cfg
- **yolov3-tiny.weights**: https://github.com/smarthomefans/darknet-test/blob/master/yolov3-tiny.weights

4. Opening Visual Studio, creating a new project, doing the settings

Open Visual Studio. In the top left corner click on File → New → Project. From the left side of the pop-up window let's select *Visual C++*, then select *Empty Project*-et, give a name to the project (for example: Carcounter), select the project's folder (for example: Desktop), and make sure that the *"Create directory for solution"* and *"Add to Source Control"* boxes in the bottom right corner are not ticked. Click on OK, so it creates our project. On the right side of the screen, in *Solution Explorer* window right click on *Header Files, Add → New Item,* select the *Header file* field, for a

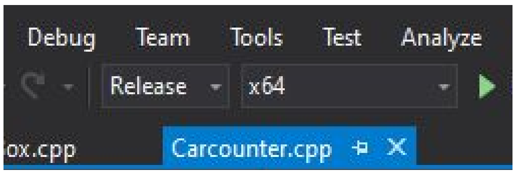
name type in *BoundingBox.h* then click on *Add*. Then repeat this whole procedure two more times from the start, but this time right click on *Source Files*, and select the *C++ file* field, and for the names first type in *BoundingBox.cpp* and for the second time type in *Carcounter.cpp*. Into the created header and cpp files copy the scripts from **BoundingBox.h, BoundingBox.cpp** and **Carcounter.cpp** then save the project.
When it's all done, make sure that in the top of Visual Studio's window the menus *"Solution Configurations"* is set to **Debug**, and *"Solution Platforms"* is set to **x64**.



Then in Visual Studio do the settings with the following **steps**:

1. On the top of the window, click on *Project* tab → *(name of the project) Properties* → *Configuration properties* → *VC++ directories* → *Include directories,* and here add the include directory of our version of OpenCV (for example: *C:\OpenCV-3.4.9\opencv\build\include;$(IncludePath)*).
2. On the top of the window, click on *Project* tab → *(name of the project) Properties* → *Configuration properties* → *VC++ directories* → *Library directories,* add the include directory of our version of OpenCV and Visual Studio (for example.: *C:\OpenCV-3.4.9\opencv\build\x64\vc15\lib;$(LibraryPath)*).
3. On the top of the window, click on *Project* tab → *(name of the project) Properties* → *Configuration properties* → *Linker* → *Input* → *Additional dependencies,* and here add the debug libraries of our version of OpenCV and VS (.lib files starting with d). For OpenCV 3.4.9 we have only one libary in general, called *opencv_worldXXXd.lib,* where XXX indicates the number of version. Add this file name to *Additional dependencies*.
4. On the top of the window, click on *Project* tab → *(name of the project) Properties* → *Configuration properties* → *Debugging* → *Command Arguments,* copy here the following: *./object_detection_yolo.out -video=traffic.mp4* . It means that our carcounting program will run on *traffic.mp4*. If you would like to select an other video, then change *traffic.mp4* to the name of the other video.

(If you would like to run the program faster, then set *"Solution Configurations"* menu to **Release**, and leave the *"Solution Platforms"* menu on **x64**.



Then repeat **steps 1 - 4.** once again, with the only difference, that in **step 3** type *opencv_worldXXXd.lib* **instead of** *opencv_worldXXX.lib*)

5. Adding the downloaded files to the project, running the program

In the last step let's copy the downloaded **traffic.mp4, coco.names, yolov3-tiny.cfg** and **yolov3-tiny.weights** files to our project's folder. Press the key *F5* to run the program with debugging, for running without debugging press *F5+Ctrl*. After running the program, the project's folder will contain *times.txt* that stores the frame processing times and *traffic_yolo.avi*, which is our output video file.