Mini Project 1 – Object Detection with YOLO

Teams: **Mixed undergraduate/graduate teams are not permitted**. Each team comes up with a team name. One team member will submit the report and Python code for the entire team. Submit as one PDF.
Grads: MP1 - MP3 are same groups as for the semester graduate project.

Writing: All writing uses 1" borders, 11-point Times New Roman font, and single spacing. Writing is required to be typed – not handwritten. Submit as a single PDF. Also submit all Python codes and running instructions in the appendices (see below).

Project Objectives: (1) Install YOLO v3 (see YOLO install handout) or a later version and use it to detect signs and other objects in MAVS imagery.
(2) Use the MAVS simulator Python API to extract simulated camera data and then detect objects.

Instructions:

1. Jospeh Redmon is the creator of YOLO. Watch this YouTube TED talk video and summarize in a few paragraphs in your report.

   https://www.ted.com/talks/joseph_redmon_how_a_computer_learns_to_recognize_objects_instantly/discussion?language=en

2. Using the YOLO install handout, install YOLO and run it on TensorFlow. The handout shows how to install YOLO v3, but you can install a more recent version, as long as it runs in TensorFlow.

3. Using MAVS:

   Then clone the examples repo (https://github.com/CGoodin/MAVS-Examples)
   The main file is **sim_example_keystrokes.py**.
   It also depends on **mavs_spav_simulation.py**.

   Edit line 18 in "mavs_spav_simulation.py" to match the file path to their "mavs_python" example, just like in setting up simulation_example.py. You can copy the file paths from simulation_example.py. If you installed in the C directory following the instructions, you do not need to edit your path

   You drive using "w", "a", "s" and "d" keys and press "c" to capture a frame (image).

   Drive around in the MAVS simulation and extract at least 50 images with no signs and at least 50 images with signs. (You can extract more for bonus points).

In your report, clearly state how many images with and without signs were captured.

4. Using the captured imagery, run YOLO and try to detect the signs. Also make note of what else is detected or if signs are missed.
   OPTIONAL for 5% bonus: Retrain yolo with sign and car images. Does performance improve over baseline model? How many images were used and where did they come from?

5. Write a report discussing YOLO performance for MAVS. Discuss what you learned. What items did YOLO detect in the imagery? What items did it incorrectly classify? Place a few of the output images in your report as examples. Make sure the figures have captions and the figures are referenced (discussed) in the text.

For performance assessment, assess qualitatively (e.g., it did very well, or it missed signs that were partially occluded…) and quantitatively (it detected XXX % of stop signs, and missed detections of YYY %), etc.

Report: Write a report using the following format:

- Cover page with names of students and NetId's.
  Also include a team name, the assignment "Mini-Project 1" and the due date.
  Note: Don't use the Canvas team name, e.g., Group 1. You must come up with a good name for the team.
- Next page – Table listing each student, their contributions, and total time spent on project in hours.
- Start rest on a new page.
- Abstract – One paragraph that briefly describes this whole exercise. Conclude with a few sentences summarizing the results.
- Introduction – Briefly discuss MAVS in your own words.
  Briefly overview YOLO and describe at a high level what it does and how it works (see #1 above – TED Talk).
  Discuss what version of YOLO was used and why this was chosen.
  Give a brief discussion of using simulated imagery in automotive autonomy. List one advantage and one disadvantage of using simulated imagery. Hint: Find 2-3 papers discussing using simulated data. Cite the papers.
- Methodology – Discuss how your team selected images and give a brief explanation of your YOLO processing.
- Results & Discussion – Please figures and tables showing results here. Discuss the results. Make a table of results. Show a few example plots. For each figure and table, provide a caption and reference the caption in your text.
- Conclusions. Draw conclusions. Answer the following questions:
    a. Overall, how well did the system perform?
    b. What can be improved in the system?
    c. What items did YOLO detect in the imagery? What did it miss?

   d. A few sentences on how simulations can be used in autonomy.

- References – list in IEEE format. Provide at least 3-4 references from journal or conference papers. Hint: MAVS and YOLO have tons of papers. The last Slide of the MAVS presentation (see Module 2) has links for papers.
- Appendix A – Python codes.
- Appendix B – List of commands or actions to run the code. Give the commands/actions to run your code and list the directory that your code resides.

Citations are to be in IEEE format. For examples of IEEE format citations, refer to https://ieee-dataport.org/sites/default/files/analysis/27/IEEE%20Citation%20Guidelines.pdf

Grading Rubric:

Area	Percent
Followed directions & report contains required materials	20
Technical content, adequate discussion of topics	50
Citations format and adequate citations Optional bonus for many good citations	10
Good grammar and proper technical writing style	20
Bonus (optional) – more than 100 images captured and used, extra weather effects, retrain YOLO, use another YOLO above v3, etc.	varies

Appendix A - Instructions for installing YOLO v3 that runs in TF 2
This material only applies if you choose to run YOLO v3

https://github.com/AryanShekarlaban/YOLO-V3

In Anaconda

```
conda create -n yolo numpy python==3.9
conda activate yolo
```

Now change to a directory to work under. I used this one

```
cd c:\spav\
```

Grab YOLO from GitHub. This will create a new directory yolo-V3.

```
conda install -c anaconda git
git clone https://github.com/AryanShekarlaban/YOLO-V3
cd yolo-V3
```

Edit the *requirements.txt* file and remove “=XXX” from these lines:

```
tensorflow
opencv-python
```

It should look like this:

```
tensorflow
numpy
opencv-python
lxml
tqdm
flask
seaborn
pillow
```

Install packages for YOLO.

```
pip install .
```

If that doesn't work, try

```
pip install -r requirements.txt
```

Install latest wget.

```
conda install -c menpo wget
```

Install keyboard and numpy

```
conda install conda-forge::keyboard
pip install numpy
```

Get YOLO trained weights.

```
mkdir weights
```

```
wget --no-check-certificate
https://pjreddie.com/media/files/yolov3.weights -O
weights/yolov3.weights
```

Note: Copy and paste the above as one line. This can take some time to download.

Prepare the network.

```
python load_weights.py
```

Now try a test image:

```
mkdir detections
```

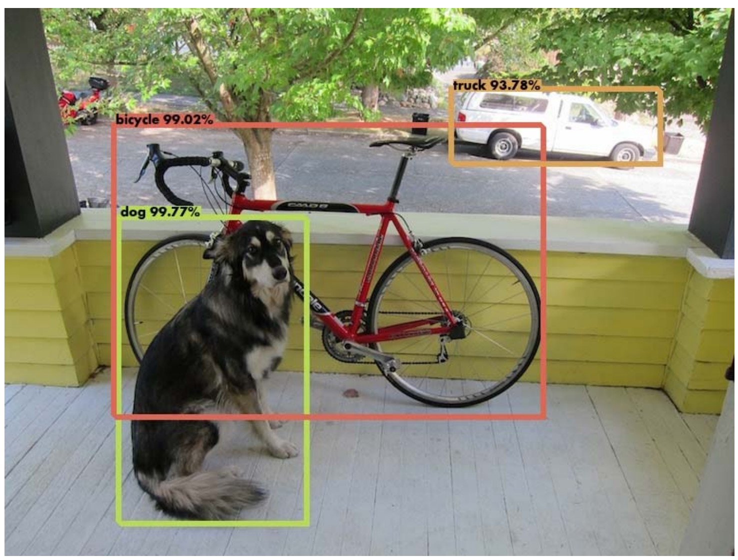
```
python detect.py --images "data/images/dog.jpg,
data/images/office.jpg"
```

Note: Second command is one line.

You should see outputs something like this

```
weights loaded
classes loaded
I0131 21:51:58.298410 15804 detect.py:60] time: 0.4768221378326416
detections:
        dog, 0.9977410435676575, [0.15920368 0.38621816 0.41723308 0.9451945 ]
        bicycle, 0.9901672005653381, [0.15322673 0.21770424 0.7412542  0.7511743 ]
        truck, 0.9378406405448914, [0.6156052  0.15119573 0.901889   0.2891039 ]
output saved to: ./detections/detection1.jpg
I0131 21:51:58.773138 15804 detect.py:60] time: 0.43175220489501953
detections:
        person, 0.9976905584335327, [0.72928727 0.3384121  0.86085737 0.7226849 ]
        chair, 0.9900699853897095, [0.22614767 0.69921005 0.35763782 0.98113084]
        person, 0.9753459095954895, [0.10638109 0.34501737 0.1945675  0.5991507 ]
        person, 0.9746911525726318, [0.6348268  0.39427337 0.69545674 0.6952963 ]
        tvmonitor, 0.910828709602356, [0.4215678  0.48383757 0.49701184 0.5818063 ]
        person, 0.8662540912628174, [0.6807398  0.3896677  0.7327339  0.66745704]
        chair, 0.8610156178474426, [0.5247927  0.574401   0.58817005 0.7867807 ]
        chair, 0.7314181923866272, [0.593796   0.5604436  0.6511531  0.68695456]
        person, 0.6713800430297852, [0.4971737  0.46682107 0.55845916 0.57764745]
        chair, 0.5135219693183899, [0.4089017 0.5883575 0.5332905 0.8246476]
output saved to: ./detections/detection2.jpg
```

The outputs are in the detections directory. They should look something like this:



How to download MAVS files for MP1

Open anaconda prompt

```
conda activate yolo
```

Install keyboard

```
conda install -c conda-forge keyboard
```

Change to the c drive

```
cd c/
```

Clone the repo and pull missing files down

```
git clone https://github.com/CGoodin/MAVS-Examples.git
```

Go to the directory containing the code (example shown below)

```
cd mavs-binaries/MAVS-Examples/PythonExamples/SPAV
```

Go to repisitory and download **mavs_spav_simulation.py** and **sim_example_keystrokes.py**

https://github.com/CGoodin/MAVS-Examples/tree/main/PythonExamples/SPAV

Edit the file **mavs_spav_simulation.py**

Check the path on line 18. If you followed the instructions for cloning MAVS on your C drive, you won't need to change it.

```
sys.path.append(r'c:/mavs-binaries/mavs_python')
```

Also, if you only want to see the camera (and not lidar) images, comment line 80 out in **sim_example_keystrokes.py**

line 80

```
#sim.lidar.DisplayPerspective()
```

If you want less fog, change these lines in the file **mavs_spav_simulation.py**:

line 45

```
self.env.SetFog(0.0) # 0.0-100.0, was 50.0
```

To change the resolution, edit line 53

```
self.cam.Initialize(640,360, 0.006222, 0.0035, 0.0035)
```

To change the exposure and compression, edit line 60

```
self.cam.SetGammaAndGain(0.4, 3.0)  # Was (0.5,2.0)
```

Run the code

```
python spa_sim_example.py human
```

To drive, use w-a-s-d keys

w = forward
s = reverse
a = left
d - right

To capture a frame, use c key

c - capture a frame (for YOLO)

To end simulation, hit **ctrl-break**

Try getting close to some signs and saving several snapshots from a few distances.

Copy the supplied detector processing files to your YOLO directory (mine is c:\spav\yolo-v3)

post_process_yolo.py
detect_rename.py

Open **post_process_yolo.py** and change base_folder (line 14) to match your directory

Run yolo on all the saved images from MAVS

```
cd c:\spav\yolo-v3
python post_process_yolo.py
```

The output files will be in the detections directory. They will have the filename “detection_xxx.jpg” where the xxx corresponds to the filename saved using MAVS simulation.