GM – Home Exam Perception Test Plan

## Test plan goal:

To verify the module (stage) of perception in the autonomous car system is functioning as expected.

## Test criteria:

**Processing time**: it is important to measure the amount of time of processing in order maintain acceptable decision making

**Accuracy of detection**: is crucial, and it is measured both by counting the number of objects of each type that are detected and by comparing each bounding box found to bounding boxes that are saved as a reference.

All tests should be compared both to best results from previous tests or from a manually set baseline and to latest results to make sure that that no regression is introduced over time.

Detailed test results should be saved to a DB.

All 3 criteria (number of objects, time of processing and accuracy of bounding boxes) must be OK for the test result to be considered PASS.

If a test Fails the reason of failure is printed

## Test Method

Images are collected and fed to the script which sends the images for analysis to get back a list of objects and bounding boxes that are found in each image.

Each analysis is compared to a baseline of results

## Test Data

It is important to collect as many images as possible representing different conditions.

Images should be taken from various heights to represent the position and angle of the camera vers the ground. This might be very different if the camera is on a car vers if it is on a truck or a bus.

Conditions to consider are:

1. Urban driving and inner-city driving
2. Light conditions such as daylight night-driving, dusk, with & without street lights
3. Oncoming traffic and following traffic
4. Images of coming around a bend or of driving around traffic circles (the images of traffic will only be caught at an angle.

## Script setup

It can be downloaded from GitHub (<https://github.com/jrweis01/GM>)

The script is written in Python 3.6.8

Besides standard packages of python it uses the packages of pytest (for internal unit testing) and it uses the luminoth package which requires that tenserflow be installed as well.

The file DB.py is a file I am using to make the implementation easier and faster. The results should be written and then read from a DB but for now I simply created this file instead.

In order to setup a new file for testing first run it in “data” mode, copy the data printed to the output screen and add that data to the baseline\_results dictionary in the DB.py file then when you run the test in “test” mode the results will be compared to this baseline.

It can be downloaded from GitHub (<https://github.com/jrweis01/GM>)

## Running the script

The file to run in Main.py

The script takes 4 command line parameters (in any order). The first 3 parameters are optional, but the last parameter is required.

**--mode or -m** determines if you are running the script in order to create a baseline for latter comparison (“data”) or if you actually running tests with comparison (“test”)

**--save or -s** Boolean value which determines if the image with the bounding boxes d should be saved to the results folder in the hard drive

**--checkpoint or -c** a string value that signifies the name of the checkpoint to connect to and use when processing the images

**--images or -i** this must appear at least once and can be repeated as many times you want. This parameter should lead to the images which you want to use for testing.