

# Intel® Edge AI for IoT Developers Nanodegree Program

## Project #05 Computer Pointer Controller

Name: Marwan Saeed Alsharabbi

Date: 08-Jule-2020





# Instructions

In this project, you will use a gaze detection model to control the mouse pointer of your computer. You will be using the [Gaze Estimation](#) model to estimate the gaze of the user's eyes and change the mouse pointer position accordingly. This project will demonstrate your ability to run multiple models in the same machine and coordinate the flow of data between those models.

## How it works

You will be using the InferenceEngine API from Intel's OpenVino ToolKit to build the project. The **gaze estimation** model requires three inputs:

- The head pose
- The left eye image
- The right eye image.

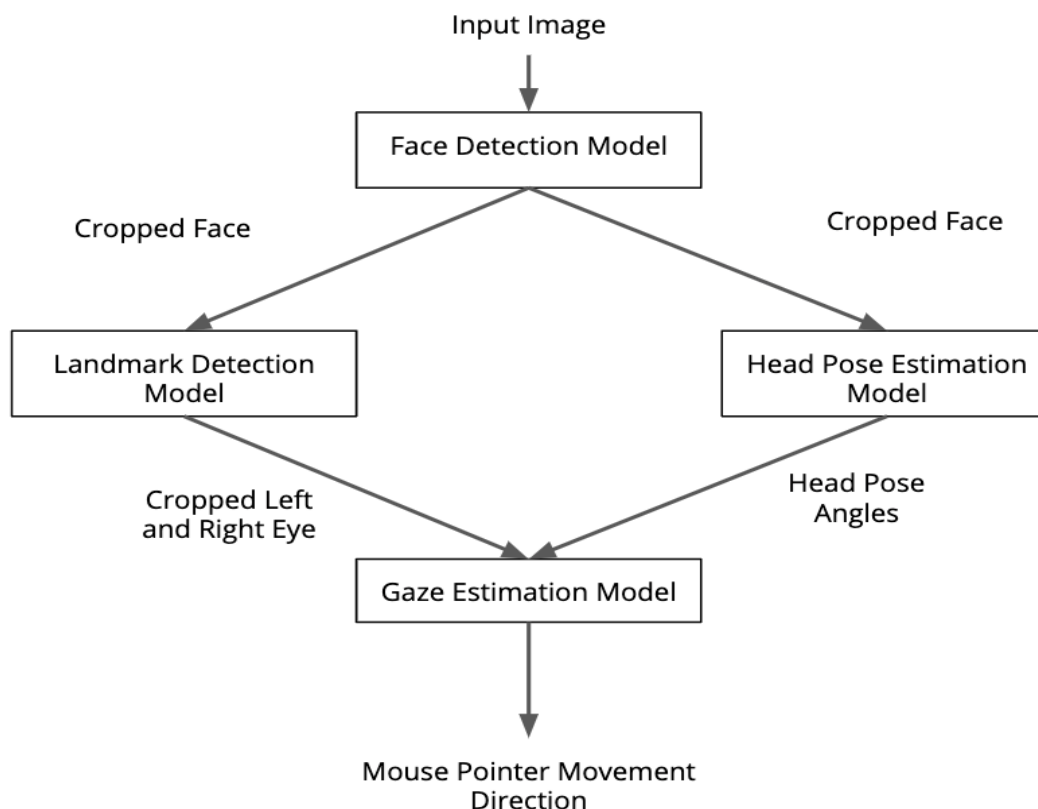
To get these inputs, you will have to use three other OpenVino models:

- [Face Detection](#)
- [Head Pose Estimation](#)
- [Facial Landmarks Detection](#).

## The Pipeline

You will have to coordinate the flow of data from the input, and then amongst the different models and finally to the mouse controller. The flow of data will look like this:





While building the flow, you will need to make sure that you are aware of the input and output shapes of each model. If the shapes or data format are different for a particular model, you can account for this in the preprocessing methods (will go over this in more detail shortly).

## Useful Links

You may find these links useful to you when building your project:

- [Inference Engine API Docs](#)
- [Model Documentation](#)

## Step 1: Project Setup

By the end of this task, you should have your dependencies installed and your project set up.





## Set up your local development environment

Before you start writing any code, you'll need to get your development environment set up on your own machine. Here are the main things to do:



- ✓ If you haven't already, download and install the **OpenVino Toolkit**. The installation directions for OpenVino can be found [here](#).
- ✓ Create a virtual environment for your project.

## Download the starter files

We've provided some helpful starter files that you'll need for the project. You can download them [here](#).

### Download the starter files

We've provided some helpful starter files that you'll need for the project. You can download them [here](#).

**Note:** After you have downloaded the starter files, you should do all of your work inside the provided directory structure—and when you're done, you'll zip and submit this entire directory structure. This will help ensure that we can successfully run your project and don't have problems getting your code working because of missing files or incorrect file paths.



- ✓ Download and unzip the starter files





### Set up your webcam (or locate the optional video file)

To do the gaze estimation, we'll need a video feed. You can either use a webcam stream as input to your model or, if you prefer, you can use the video file that we've provided (this is located in the `bin` folder of the starter files).



Set up your webcam or locate the video file

### Download the models

These can be downloaded using the `model downloader`. The models you will need are:



Face Detection



Head Pose Estimation



Facial Landmarks Detection



Gaze Estimation Model

## Step 2: Build the Inference Pipeline

Using the *Inference Engine API*, you will need to build a pipeline that runs inference on the models using the inputs. The starter code includes methods that will help you set up this pipeline.

Before you start writing any code, the first step is just to get familiar with the starter files and how they will be used to construct the inference pipeline. We'll go over each file in the video below. You should open up the copy you downloaded and follow along.





### Get familiar with the starter code

Make sure you can locate each of these files—and that you have at least a basic understanding of how it will be used in the pipeline:



- ✓ `model.py`
- ✓ `mouse_controller.py`
- ✓ `input_feeder.py`

### Make copies of the `model.py` file

The `model.py` file contains a sample class for loading, checking, and running inference on a single model. It also has methods to pre-process the inputs and outputs to the model, but these are not yet finished—that will be your job.

Since each of the models has different requirements, you'll need to make a copy of `model.py` for each of the models. There are four models, so you should create four copies of `model.py` and rename them as follows:



- ✓ `face_detection.py`
- ✓ `head_pose_estimation.py`
- ✓ `facial_landmarks_detection.py`
- ✓ `gaze_estimation.py`





Complete the `model.py` code for each model

Inside the copies of `model.py` that you just made, you'll find several methods that are marked with `TODO` comments.

It's your job to complete these methods for each of the three models, so that they accomplish the following goals:



- ✓ **Loading the model:** The `model.py` file has a `load_model` method. This method is meant to help you load the model.
- ✓ **Preprocessing the inputs:** Each model requires different inputs and you will have to pre-process the inputs before feeding them into the model. You can use the `preprocess_input` method to do that.
- ✓ **Inference:** The `predict` function is meant to get predictions from the model.
- ✓ **Preprocessing the outputs:** The outputs of the model need to be preprocessed before they can be used by other models or be used to change the mouse pointer. You can do that in the `preprocess_outputs` method.

### Step 3: Complete the README

An important part of your project is creating a README file that describes the project, explains how to set up and run the code, and describes your benchmarks and results. We've included a template in the starter files (that you downloaded earlier), with TODOs for each of the things you should include.





An important part of your project is creating a `README` file that describes the project, explains how to set up and run the code, and describes your benchmarks and results. We've included a template in the starter files (that you downloaded earlier), with `TODO`s for each of the things you should include.

***Note:** You'll see some optional `TODO`s for "standout suggestions". We'll go over these on the next page.*



Complete all of the required `TODO`s in the `README` template.

## Part 4: Standout suggestions

If you've been having a good time with this project and want to take it further, here are some suggestions for a whole bunch of additional things you can do with it. If you want the additional practice or are wanting to turn this project into a nice portfolio piece, we recommend taking a shot at these—but note that these are all optional, so you can skip any (or all) of them and go straight to the project submission page.

For your reference, here's a summary of the standout suggestions:



**Can you improve your inference speed without significant drop in performance by changing the precision of some of the models?** In your `README`, include a short write-up that explains the procedure and the experiments you ran to find out the best combination of precision.



**Benchmark the running times of different parts of the preprocessing and inference pipeline and let the user specify a CLI argument if they want to see the benchmark timing.** Use the `get_perf_counts` API to print the time it takes for each layer in the model.







- ✓ **Use the VTune Amplifier to find hotspots in your Inference Engine Pipeline.** Write a short write-up in the README about the hotspots in your system and how you solved them.
- ✓ **There will be certain edge cases that will cause your system to not function properly.** Examples of this include: lighting changes, multiple people in the same input frame, and so on. Make changes in your preprocessing and inference pipeline to solve some of these issues. Write a short write-up in the README about the problem it caused and your solution.
- ✓ **Add a toggle to the UI to shut off the camera feed and show stats only (as well as to toggle the camera feed back on).** Show how this affects performance and power as a short write up in the README file.
- ✓ **Build an inference pipeline for both video file and webcam feed as input.** Allow the user to select their input option in the command line arguments.

Update your `README`



- ✓ If you've done any of the above standout suggestions, document these in your `README` file.





## Part 5: Check Your Work

Your reviewer will use the instructions you provide to run and test your project, so it's important to make sure your instructions are clear, that your code works, and that all necessary dependencies are included.

**Note:** *The models themselves are large files and you do not need to include them in your submission (your reviewer will have a copy of these themselves).*

Before submitting, be sure you've done the following:



- ✓ Make sure all of the files for your project are in the same directory.
- ✓ Make sure the instructions for running your project are in your README.
- ✓ Test the instructions to make sure your project runs!
- ✓ Zip the entire directory containing your code and all required files/dependencies (except the model files). Make sure you've included everything that your reviewer will need to run your project.
- ✓ To reduce the size of your submission, make sure the zip file does *not* include the model files. The final zip file should be no more than a few megabytes in size.
- ✓ If you'd like to be especially thorough, you can have a look at the project [rubric](#). This describes the exact requirements that your reviewer will be looking at when they go over your submission.





## Computer Pointer Controller

This is the third project for the Udacity course Intel® Edge AI for IoT Developers. The purpose of this project is to use multiple deep learning models to move a mouse cursor on a screen using eye and head pose from a webcam or video.

When the program starts, a window showing the input video appears in the middle of the screen and the mouse pointer is automatically moved to the center of the screen also. As the person in the video moves their eyes and head, the mouse cursor will move in the same direction. Only one person must be in the frame for the mouse cursor to move.

The mouse controller is hardcoded for fast, high-precision movement but you will still see it moving very slowly.

The video and detections from model outputs are shown by default but they can be turned off using command line arguments.

## Project Set Up and Installation

- Install OpenVINO for [Windows](#).
- Source OpenVINO environment variables:
  - Windows:

```
"C:\Program Files (x86)\IntelSWTools\openvino\bin\setupvars.bat"
```

- Create Virtual Environment using command `virtualenv venv` in the command prompt

```
virtualenv [name of the virtual environment. For example, my_project]
```

```
C:\Users\enmar>virtualenv venv
Using base prefix 'c:\users\enmar\appdata\local\programs\python\python36'
New python executable in C:\Users\enmar\venv\Scripts\python.exe
Installing setuptools, pip, wheel...
done.
Overwriting C:\Users\enmar\venv\Scripts\activate.bat with new content
Overwriting C:\Users\enmar\venv\Scripts\deactivate.bat with new content
Overwriting C:\Users\enmar\venv\Scripts\activate.ps1 with new content
Overwriting C:\Users\enmar\venv\Scripts\activate with new content
Overwriting C:\Users\enmar\venv\Scripts\activate_this.py with new content
Overwriting C:\Users\enmar\venv\Scripts\activate.xsh with new content
```





```
C:\Users\enmar>virtualenv venv
Using base prefix 'c:\users\enmar\appdata\local\programs\python\python36'
New python executable in C:\Users\enmar\venv\Scripts\python.exe
Installing setuptools, pip, wheel...
done.

C:\Users\enmar>cd venv/Scripts/

C:\Users\enmar\venv\Scripts>activate

(venv) C:\Users\enmar\venv\Scripts>cd C:\Program Files (x86)\IntelSWTools\openvino\bin\

(venv) C:\Program Files (x86)\IntelSWTools\openvino\bin>setupvars.bat
Python 3.6.5
ECHO is off.
PYTHONPATH=C:\Program Files (x86)\IntelSWTools\openvino\deployment_tools\open_model_zoo\tools\accuracy_checker;C:\Program Files (x86)\IntelSWTools\openvino\python\python3.6;C:\Program Files (x86)\IntelSWTools\openvino\python\python3;C:\Program Files (x86)\IntelSWTools\openvino\deployment_tools\model_optimizer;
[setupvars.bat] OpenVINO environment initialized

(venv) C:\Program Files (x86)\IntelSWTools\openvino\bin>
```

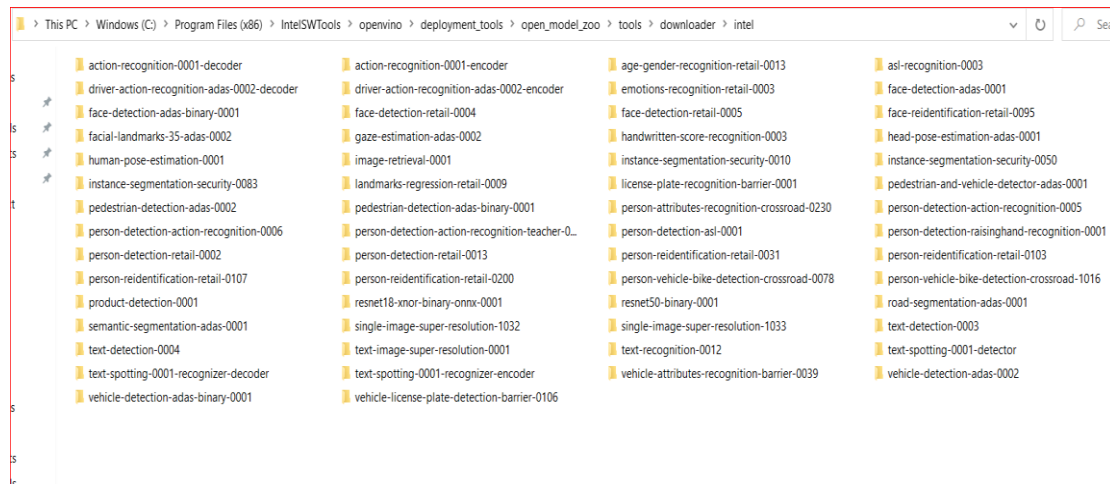
Source [my\_Project]/bin/activate  
pip3 install -r ../requirements.txt

```
Command Prompt
Downloading pickleshare-0.7.5-py2.py3-none-any.whl (6.9 kB)
Collecting idna<2.9,>=2.5
  Downloading idna-2.8-py2.py3-none-any.whl (58 kB)
  58 kB 1.2 MB/s
Collecting charset3.1.0,>=3.0.2
  Using cached charset-3.0.4-py2.py3-none-any.whl (133 kB)
Collecting certifi>=2017.4.17
  Downloading certifi-2020.6.20-py2.py3-none-any.whl (156 kB)
  156 kB 726 kB/s
Collecting urllib3<1.25,0.11.25,>=1.25.1
  Downloading urllib3-1.25.9-py2.py3-none-any.whl (126 kB)
  126 kB 656 kB/s
Collecting six<2,>=1.9.0
  Downloading six-1.15.0-py2.py3-none-any.whl (10 kB)
Collecting importlib-metadata<2,>=1.0; python_version < "3.8"
  Downloading importlib_metadata-1.7.0-py2.py3-none-any.whl (31 kB)
Collecting distlib<1,>=0.3.0
  Downloading distlib-0.3.1-py2.py3-none-any.whl (335 kB)
  335 kB 1.1 MB/s
Collecting filelock<3,>=3.0.0
  Using cached filelock-3.0.12-py3-none-any.whl (7.6 kB)
Collecting importlib-resources<2,>=1.0; python_version < "3.7"
  Downloading importlib_resources-1.5.0-py2.py3-none-any.whl (21 kB)
Collecting appdirs<2,>=1.4.3
  Downloading appdirs-1.4.4-py2.py3-none-any.whl (9.6 kB)
Collecting salsparse<0.2.2
  Using cached salsparse-0.2.1-py2.py3-none-any.whl (40 kB)
Collecting pytz
  Downloading pytz-2020.1-py2.py3-none-any.whl (510 kB)
  510 kB 939 kB/s
Collecting asgiref<=3.2
  Using cached asgiref-3.2.10-py3-none-any.whl (19 kB)
Collecting ipython-genutils
  Downloading ipython_genutils-0.2.0-py2.py3-none-any.whl (26 kB)
Collecting parso<0.8.0,>=0.7.0
  Downloading parso-0.7.0-py2.py3-none-any.whl (100 kB)
  100 kB 1.2 MB/s
Collecting wcwidth
  Downloading wcwidth-0.2.5-py2.py3-none-any.whl (30 kB)
Collecting zipp>=0.5
  Using cached zipp-3.1.0-py3-none-any.whl (4.9 kB)
Installing collected packages: salsparse, pytz, asgiref, django, Pillow, image, decorator, pygments, ipython-genutils, six, traitlets, parso, jedi, wcwidth, prompt-toolkit, backcall, colorama, pickleshare, ipython, ipdb, numpy, idna, charset, certifi, urllib3, requests, zipp, importlib-metadata, distlib, filelock, importlib-resources, appdirs, virtualenv, openvc-python
Successfully installed Pillow-6.2.1 appdirs-1.4.4 asgiref-3.2.10 backcall-0.2.0 certifi-2020.6.20 charset-3.0.4 colorama-0.4.3 decorator-4.4.2 distlib-0.3.1 django-3.0.8 filelock-3.0.12 idna-2.8 image-1.5.27 imp
pylib-metadata-1.7.0 importlib-resources-1.5.0 ipdb-0.12.1 ipython-7.10.2 ipython-genutils-0.2.0 jedi-0.17.1 numpy-1.19.4 openvc-python-4.1.0.25 parso-0.7.0 pickleshare-0.7.5 prompt-toolkit-3.0.5 pygments-2.6.1
pytz-2020.1 requests-2.22.0 six-1.15.0 salsparse-0.3.1 traitlets-4.3.1 urllib3-1.25.5 virtualenv-20.0.16 wcwidth-0.2.5 zipp-3.1.0

(venv) C:\Adacite\Project\starter>
```

- Install the project requirements by running  
`pip install -r requirements.txt`
- Download the pre-trained models in the `models/` folder by running `download_models.bat` from the root directory





I've selected the models for project and copy the models and transfer to file project

To run this application locally, one needs to have openVINO toolkit installed.

First enable the virtual environment:

```
cd C:\Program Files (x86)\IntelSWTools\openvino\bin\setupvars.sh
```

Secondly, activate the python virtual environment. To install it, run the following commands in your terminal:

I have checked Inference Time, Model Loading Time, and Frames per Second model for FP16, FP32, and FP32-INT8 of all the models except Face Detection Model. Face Detection Model was only available on FP32-INT1 precision.

You can use below commands to get outcomes for respective precisions,.

### Device CPU – Model - FP32-INT8

```
python main.py -fdm C:/UdaciteProject/starter/models/face-detection-adas-0001/FP32-INT8/face-detection-adas-0001.xml
```

```
-lrm C:/UdaciteProject/starter/models/landmarks-regression-retail-0009/FP32-INT8/landmarks-regression-retail-0009.xml
```

```
-hpm C:/UdaciteProject/starter/models/head-pose-estimation-adas-0001/FP32-INT8/head-pose-estimation-adas-0001.xml -gem C:/UdaciteProject/starter/models/gaze-estimation-adas-0002/FP32-INT8/gaze-estimation-adas-0002.xml -inp C:/UdaciteProject/starter/bin/demo.mp4 -d CPU -o C:/UdaciteProject/starter/outcomes_CPU/FP32-INT8/ -flags face_frame face_eyes
```





```
(venv) C:\UdaciteProject\starter\src>python main.py -fdm C:/UdaciteProject/starter/models/face-detection-adas-0001/FP32/face-detection-adas-0001.xml -lrm C:/UdaciteProject/starter/models/landmarks-regression-retail-0009/FP32-INT8/landmarks-regression-retail-0009.xml -hpm C:/UdaciteProject/starter/models/head-pose-estimation-adas-0001/FP32-INT8/head-pose-estimation-adas-0001.xml -gem C:/UdaciteProject/starter/models/gaze-estimation-adas-0002/FP32-INT8/gaze-estimation-adas-0002.xml -inp C:/UdaciteProject/starter/bin/demo.mp4 -d CPU -o C:/UdaciteProject/starter/outcomes_CPU/FP32-INT8/ -flags face_frame face_eyes
MFX: Unsupported extension: C:/UdaciteProject/starter/bin/demo.mp4
MFX: Unsupported FourCC: avc1 (0x31637661)
[WinError 183] Cannot create a file when that file already exists: 'C:/UdaciteProject/starter/outcomes_CPU/FP32-INT8/'
Total time: 32.02290606498718 s
Total inference time: 32.0 s
Total model load time, 2.662998914718628 s
FPS: 1.84375 frames/s
```

## CPU – Model -FP16

```
python main.py -fdm C:/UdaciteProject/starter/models/face-detection-adas-0001/FP16/face-detection-adas-0001.xml
-lrm C:/UdaciteProject/starter/models/landmarks-regression-retail-0009/FP16/landmarks-regression-retail-0009.xml -hpm C:/UdaciteProject/starter/models/head-pose-estimation-adas-0001/FP16/head-pose-estimation-adas-0001.xml
-gem C:/UdaciteProject/starter/models/gaze-estimation-adas-0002/FP16/gaze-estimation-adas-0002.xml
-inp C:/UdaciteProject/starter/bin/demo.mp4 -d CPU -o
C:/UdaciteProject/starter/outcomes_CPU/FP16/ -flags face_frame face_eyes
```

```
(venv) C:\UdaciteProject\starter\src>python main.py -fdm C:/UdaciteProject/starter/models/face-detection-adas-0001/FP16/face-detection-adas-0001.xml -lrm C:/UdaciteProject/starter/models/landmarks-regression-retail-0009/FP16/landmarks-regression-retail-0009.xml -hpm C:/UdaciteProject/starter/models/head-pose-estimation-adas-0001/FP16/head-pose-estimation-adas-0001.xml -gem C:/UdaciteProject/starter/models/gaze-estimation-adas-0002/FP16/gaze-estimation-adas-0002.xml -inp C:/UdaciteProject/starter/bin/demo.mp4 -d CPU -o C:/UdaciteProject/starter/outcomes_CPU/FP16/ -flags face_frame face_eyes
MFX: Unsupported extension: C:/UdaciteProject/starter/bin/demo.mp4
MFX: Unsupported FourCC: avc1 (0x31637661)
[WinError 183] Cannot create a file when that file already exists: 'C:/UdaciteProject/starter/outcomes_CPU/FP16/'
Total time: 31.845104932785034 s
Total inference time: 31.8 s
Total model load time, 1.1650731563568115 s
FPS: 1.8553459119496856 frames/s
```

## CPU – Model –FP32

```
python main.py -fdm C:/UdaciteProject/starter/models/face-detection-adas-0001/FP32/face-detection-adas-0001.xml
-lrm C:/UdaciteProject/starter/models/landmarks-regression-retail-0009/FP32/landmarks-regression-retail-0009.xml -hpm C:/UdaciteProject/starter/models/head-pose-estimation-adas-0001/FP32/head-pose-estimation-adas-0001.xml
-gem C:/UdaciteProject/starter/models/gaze-estimation-adas-0002/FP32/gaze-estimation-adas-0002.xml
-inp C:/UdaciteProject/starter/bin/demo.mp4 -d CPU -o
C:/UdaciteProject/starter/outcomes_CPU/FP32/ -flags face_frame face_eyes
```

```
(venv) C:\UdaciteProject\starter\src>python main.py -fdm C:/UdaciteProject/starter/models/face-detection-adas-0001/FP32/face-detection-adas-0001.xml -lrm C:/UdaciteProject/starter/models/landmarks-regression-retail-0009/FP32/landmarks-regression-retail-0009.xml -hpm C:/UdaciteProject/starter/models/head-pose-estimation-adas-0001/FP32/head-pose-estimation-adas-0001.xml -gem C:/UdaciteProject/starter/models/gaze-estimation-adas-0002/FP32/gaze-estimation-adas-0002.xml -inp C:/UdaciteProject/starter/bin/demo.mp4 -d CPU -o C:/UdaciteProject/starter/outcomes_CPU/FP32/ -flags face_frame face_eyes
MFX: Unsupported extension: C:/UdaciteProject/starter/bin/demo.mp4
MFX: Unsupported FourCC: avc1 (0x31637661)
[WinError 183] Cannot create a file when that file already exists: 'C:/UdaciteProject/starter/outcomes_CPU/FP32/'
Total time: 31.645941972732544 s
Total inference time: 31.6 s
Total model load time, 0.9303081035614014 s
FPS: 1.8670886075949367 frames/s
```







## Device GPU – Model - FP32-INT8

```
python main.py -fdm C:/UdaciteProject/starter/models/face-detection-adas-0001/FP32-INT8/face-detection-adas-0001.xml  
-lrm C:/UdaciteProject/starter/models/landmarks-regression-retail-0009/FP32-INT8/landmarks-regression-retail-0009.xml  
-hpm C:/UdaciteProject/starter/models/head-pose-estimation-adas-0001/FP32-INT8/head-pose-estimation-adas-0001.xml -gem C:/UdaciteProject/starter/models/gaze-estimation-adas-0002/FP32-INT8/gaze-estimation-adas-0002.xml  
-inp C:/UdaciteProject/starter/bin/demo.mp4 -d GPU -o C:/UdaciteProject/starter/outcomes_GPU/FP32-INT8/ -flags face_frame face_eyes
```

```
(venv) C:\UdaciteProject\starter\src>python main.py -fdm C:/UdaciteProject/starter/models/face-detection-adas-0001/FP32-INT8/face-detection-adas-0001.xml -lrm C:/UdaciteProject/starter/models/landmarks-regression-retail-0009/FP32-INT8/landmarks-regression-retail-0009.xml -hpm C:/UdaciteProject/starter/models/head-pose-estimation-adas-0001/FP32-INT8/head-pose-estimation-adas-0001.xml -gem C:/UdaciteProject/starter/models/gaze-estimation-adas-0002/FP32-INT8/gaze-estimation-adas-0002.xml -inp C:/UdaciteProject/starter/bin/demo.mp4 -d GPU -o C:/UdaciteProject/starter/outcomes_GPU/FP32-INT8/ -flags face_frame face_eyes  
MFX: Unsupported extension: C:/UdaciteProject/starter/bin/demo.mp4  
MFX: Unsupported FourCC: avc1 (0x31637661)  
[WinError 183] Cannot create a file when that file already exists: 'C:/UdaciteProject/starter/outcomes_GPU/FP32-INT8/'  
Total time: 34.146677017211914 s  
Total inference time: 34.1 s  
Total model load time, 47.70037484169006 s  
FPS: 1.7302052785923754 frames/s
```

## GPU – Model – FP16

```
python main.py -fdm C:/UdaciteProject/starter/models/face-detection-adas-0001/FP16/face-detection-adas-0001.xml  
-lrm C:/UdaciteProject/starter/models/landmarks-regression-retail-0009/FP16/landmarks-regression-retail-0009.xml -hpm C:/UdaciteProject/starter/models/head-pose-estimation-adas-0001/FP16/head-pose-estimation-adas-0001.xml  
-gem C:/UdaciteProject/starter/models/gaze-estimation-adas-0002/FP16/gaze-estimation-adas-0002.xml  
-inp C:/UdaciteProject/starter/bin/demo.mp4 -d GPU -o C:/UdaciteProject/starter/outcomes_GPU/FP16/ -flags face_frame face_eyes
```

```
(venv) C:\UdaciteProject\starter\src>python main.py -fdm C:/UdaciteProject/starter/models/face-detection-adas-0001/FP16/face-detection-adas-0001.xml -lrm C:/UdaciteProject/starter/models/landmarks-regression-retail-0009/FP16/landmarks-regression-retail-0009.xml -hpm C:/UdaciteProject/starter/models/head-pose-estimation-adas-0001/FP16/head-pose-estimation-adas-0001.xml -gem C:/UdaciteProject/starter/models/gaze-estimation-adas-0002/FP16/gaze-estimation-adas-0002.xml -inp C:/UdaciteProject/starter/bin/demo.mp4 -d GPU -o C:/UdaciteProject/starter/outcomes_GPU/FP16/ -flags face_frame face_eyes  
MFX: Unsupported extension: C:/UdaciteProject/starter/bin/demo.mp4  
MFX: Unsupported FourCC: avc1 (0x31637661)  
[WinError 183] Cannot create a file when that file already exists: 'C:/UdaciteProject/starter/outcomes_GPU/FP16/'  
Total time: 32.57253837585449 s  
Total inference time: 32.6 s  
Total model load time, 34.921903133392334 s  
FPS: 1.8098159509202454 frames/s
```

## GPU – Model – FP32

```
python main.py -fdm C:/UdaciteProject/starter/models/face-detection-adas-0001/FP32/face-detection-adas-0001.xml  
-lrm C:/UdaciteProject/starter/models/landmarks-regression-retail-0009/FP32/landmarks-regression-retail-0009.xml -hpm C:/UdaciteProject/starter/models/head-pose-estimation-adas-0001/FP32/head-pose-estimation-adas-0001.xml
```





```
gem C:/UdacityProject/starter/models/gaze-estimation-adas-0002/FP32/gaze-estimation--  
adas-0002.xml  
-inp C:/UdacityProject/starter/bin/demo.mp4 -d GPU -o  
C:/UdacityProject/starter/outcomes_GPU/FP32/ -flags face_frame face_eyes
```

```
(venv) C:\UdacityProject\starter>python main.py -fdm C:/UdacityProject/starter/models/face-detection-adas-0001/FP32/  
face-detection-adas-0001.xml -lrm C:/UdacityProject/starter/models/landmarks-regression-retail-0009/FP32/landmarks-regre  
ssion-retail-0009.xml -hpm C:/UdacityProject/starter/models/head-pose-estimation-adas-0001/FP32/head-pose-estimation-ada  
s-0001.xml -gem C:/UdacityProject/starter/models/gaze-estimation-adas-0002/FP32/gaze-estimation-adas-0002.xml -inp C:/Ud  
acityProject/starter/bin/demo.mp4 -d GPU -o C:/UdacityProject/starter/outcomes_GPU/FP32/ -flags face_frame face_eyes  
MFX: Unsupported extension: C:/UdacityProject/starter/bin/demo.mp4  
MFX: Unsupported FourCC: avc1 (0x31637661)  
[WinError 183] Cannot create a file when that file already exists: 'C:/UdacityProject/starter/outcomes_GPU/FP32/'  
Total time: 32.82218098640442 s  
Total inference time: 32.8 s  
Total model load time, 33.834617137908936 s  
FPS: 1.7987804878048783 frames/s
```

## Command Line Arguments for Running the app

Following are command line arguments that can use for while running the main.py file python main.py:-

1. -h (required) : Get the information about all the command line arguments
2. -fdm (required) : Specify the path of Face Detection model's xml file
3. lrm : Specify path of xml file of landmark regression model
4. -hpm (required) : Specify the path of Head Pose Estimation model's xml file
5. -gem (required) : Specify the path of Gaze Estimation model's xml file
6. -inp (required) : Specify the path of input video file or enter cam for taking input video from webcam
7. -prob (optional) : Specify the probability threshold for face detection model to detect the face accurately from video frame.
8. -flags (optional): Specify the flags from face\_frame, face\_eyes frame, hp, ge if you want to visualize the output of corresponding models of each frame (write flags with space separation. Ex:- -flags fd fld hp).
9. - d (Optional): Specify Device for inference, the device can be CPU, GPU, FPGU, MYRID
10. - o : Specify path of output folder where we will store results







```
usage: main.py [-h] -fdm FACEDETECTIONMODEL -lrm LANDMARKREGRESSIONMODEL -hpm HEADPOSEESTIMATIONMODEL -gem GAZEESTIMATIONMODEL -inp INPUT
               [-flags PREVIEWFLAGS [PREVIEWFLAGS ...]] [-prob PROB_THRESHOLD]
               [-d DEVICE] [-o OUTPUT_PATH]

optional arguments:
  -h, --help            show this help message and exit
  -fdm FACEDETECTIONMODEL, --faceDetectionModel FACEDETECTIONMODEL
                        Specify path of xml file of face detection model
  -lrm LANDMARKREGRESSIONMODEL, --landmarkRegressionModel LANDMARKREGRESSIONMODEL
                        Specify path of xml file of landmark regression model
  -hpm HEADPOSEESTIMATIONMODEL, --headPoseEstimationModel HEADPOSEESTIMATIONMODEL
                        Specify path of xml file of Head Pose Estimation model
  -gem GAZEESTIMATIONMODEL, --gazeEstimationModel GAZEESTIMATIONMODEL
                        Specify path of xml file of Gaze Estimation model
  -inp INPUT, --input INPUT
                        Specify path of input Video file or cam for webcam
  -flags PREVIEWFLAGS [PREVIEWFLAGS ...], --previewFlags PREVIEWFLAGS [PREVIEWFLAGS ...]
                        Specify flag from ff, fl like -flags ff fl(Space
                        separated if multiple values)face_frame for
                        faceDetectionModel, face_eyes for
                        landmarkRegressionModel
  -prob PROB_THRESHOLD, --prob_threshold PROB_THRESHOLD
                        Specify probability threshold for face detection model
  -d DEVICE, --device DEVICE
                        Specify Device for inferenceIt can be CPU, GPU, FPGU,
                        MYRID
  -o OUTPUT_PATH, --output_path OUTPUT_PATH
```

## Benchmarks

I ran the model inference on CPU and GPU device on local machine given same input video and same virtual environment. Listed below are hardware versions:

The screenshot displays the Intel® UHD Graphics Control Panel window. The title bar reads "Intel® UHD Graphics Control Panel" and "Options and Support". The left sidebar contains a navigation menu with "Information Center" (selected), "Hot Key Manager", "Preferences", "Profiles", and "Support". Below the menu is a "Select Option" dropdown set to "System Information". The main content area, titled "Intel(R) UHD Graphics 630", displays system information in a two-column table.

Property	Value
Report Date:	Wednesday, July 8, 2020
Report Time [hh:mm:ss]:	7:48:25 PM
Driver Version:	26.20.100.6911
Operating System:	Windows* 10 Home (10.0.19041)
Physical Memory:	8081 MB
Vendor ID:	8086
Device ID:	3E9B
Device Revision:	00
Graphics Output Protocol (GOP) Version:	9.0.1074
Current Resolution:	1920 x 1080
Processor:	Intel(R) Core(TM) i5-8300H CPU @ 2.30GHz
Processor Speed:	2304 MHz
Processor Graphics in Use:	Intel(R) UHD Graphics 630
Shader Version:	5.1
OpenGL* Version:	4.6
OpenCL* Version:	2.1
Vulkan* Version:	1.1.103
CUI Appx Version:	3.3.0.0
Microsoft DirectX*	12.0
Runtime Version:	12.0
Hardware-Supported Version:	12.0



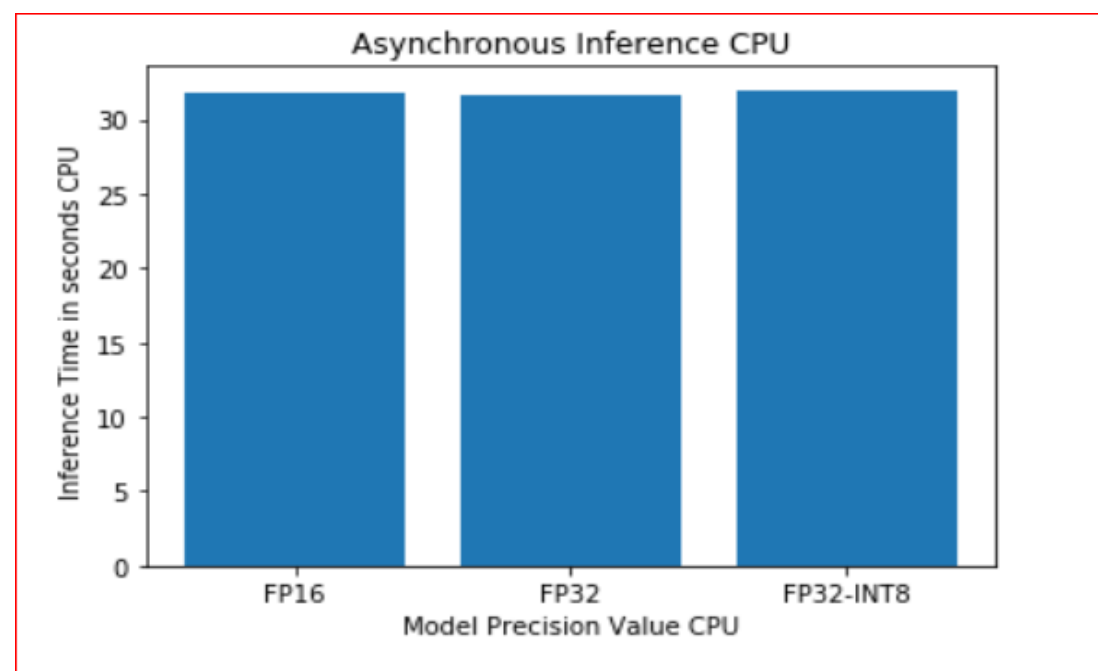
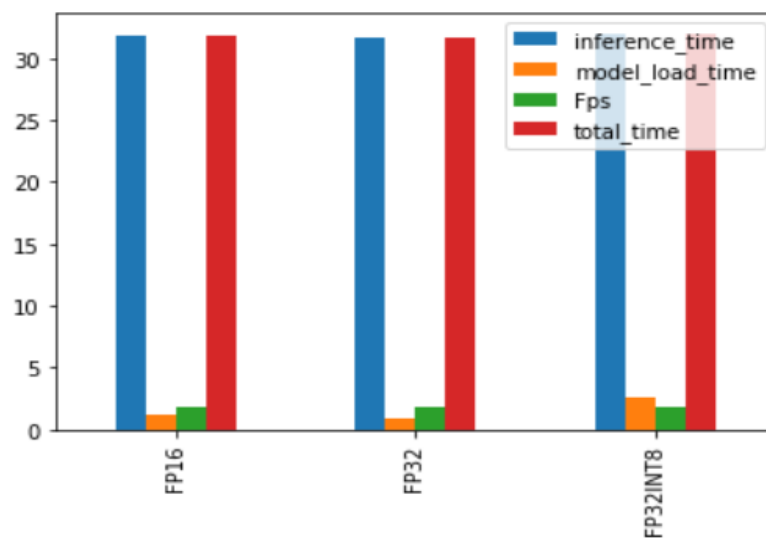


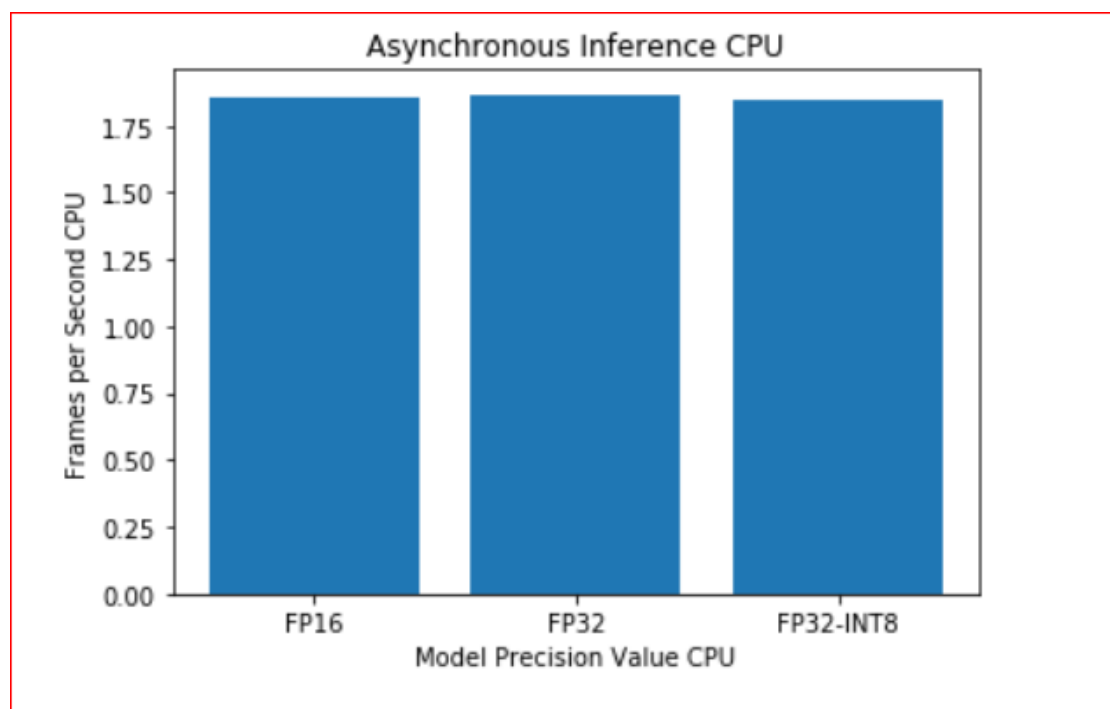
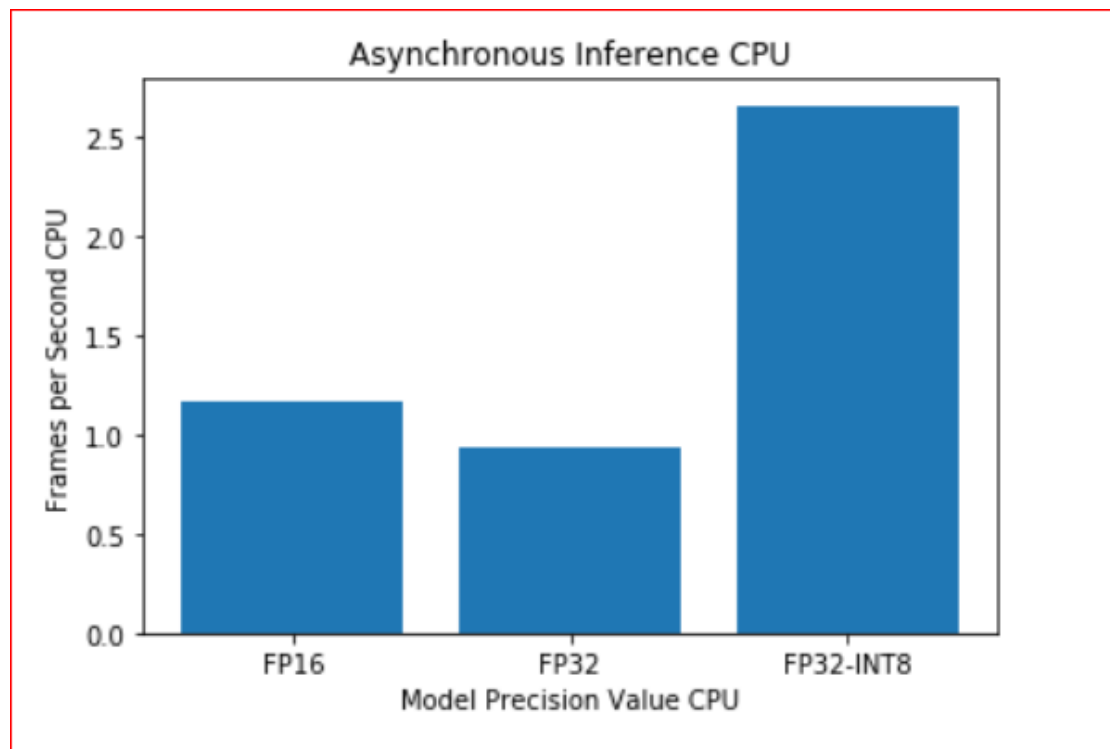
Due to non-availability of FPGA and VPU in local machine, I did not run inference for these device types

## Benchmark results of the model. CPU

	inference_time	model_load_time	Fps	total_time
FP16	31.8	1.165073	1.855346	31.845105
FP32	31.6	0.930308	1.867089	31.645942
FP32INT8	32.0	2.662999	1.843750	32.022906

```
[8]: result_transposed.plot.bar();
```





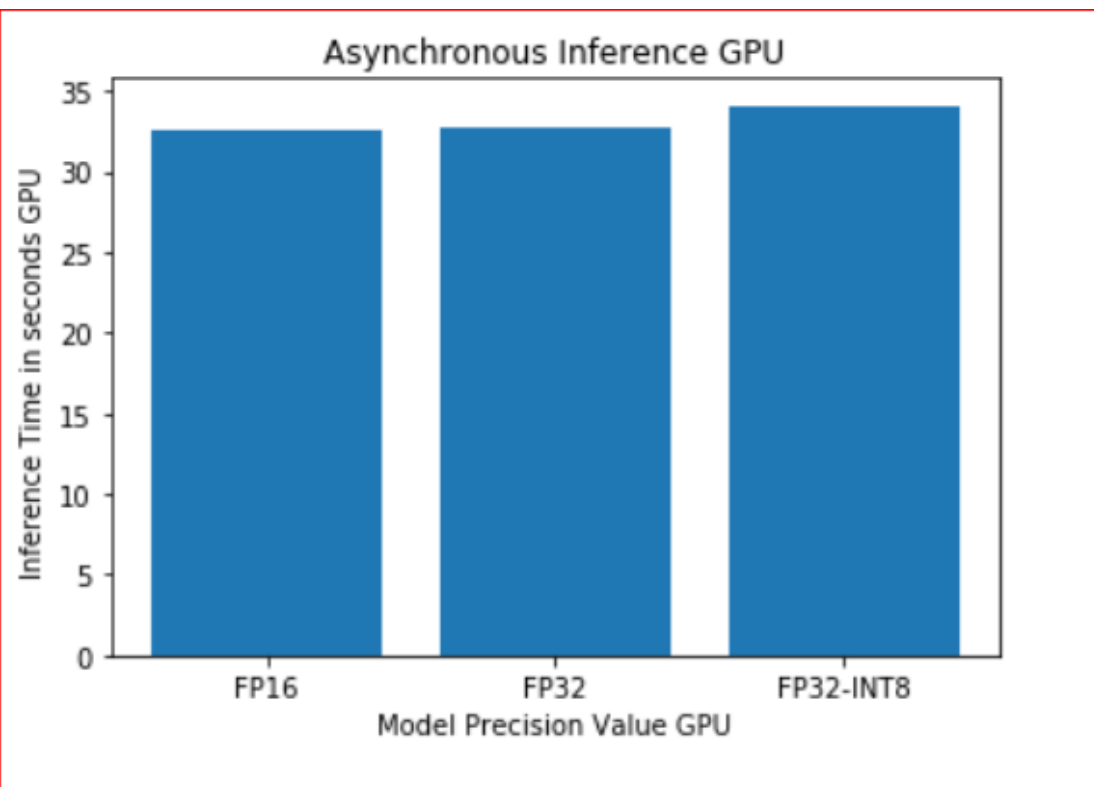
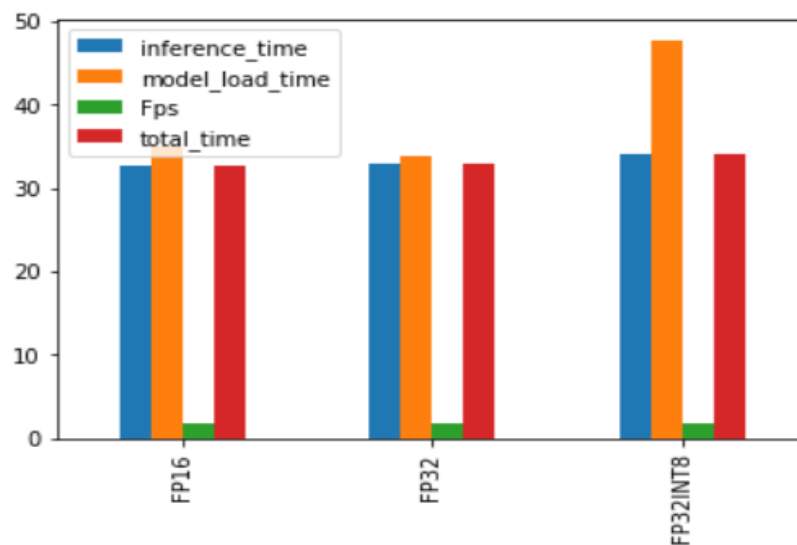


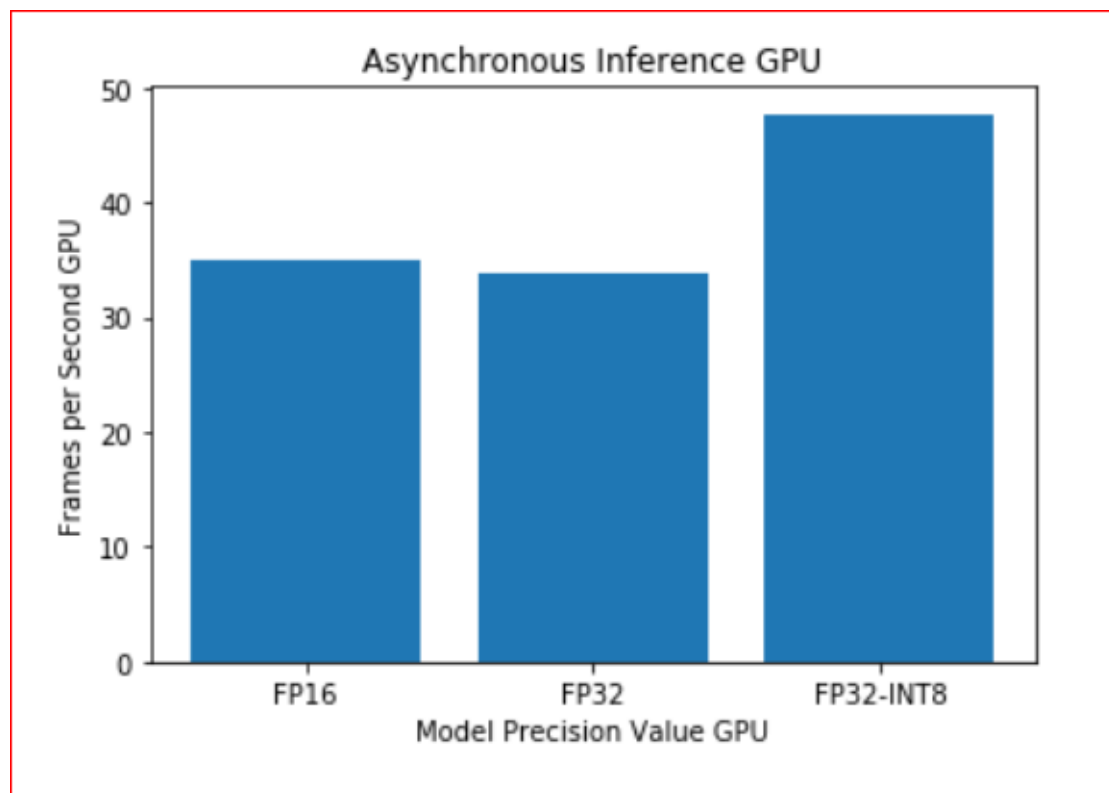
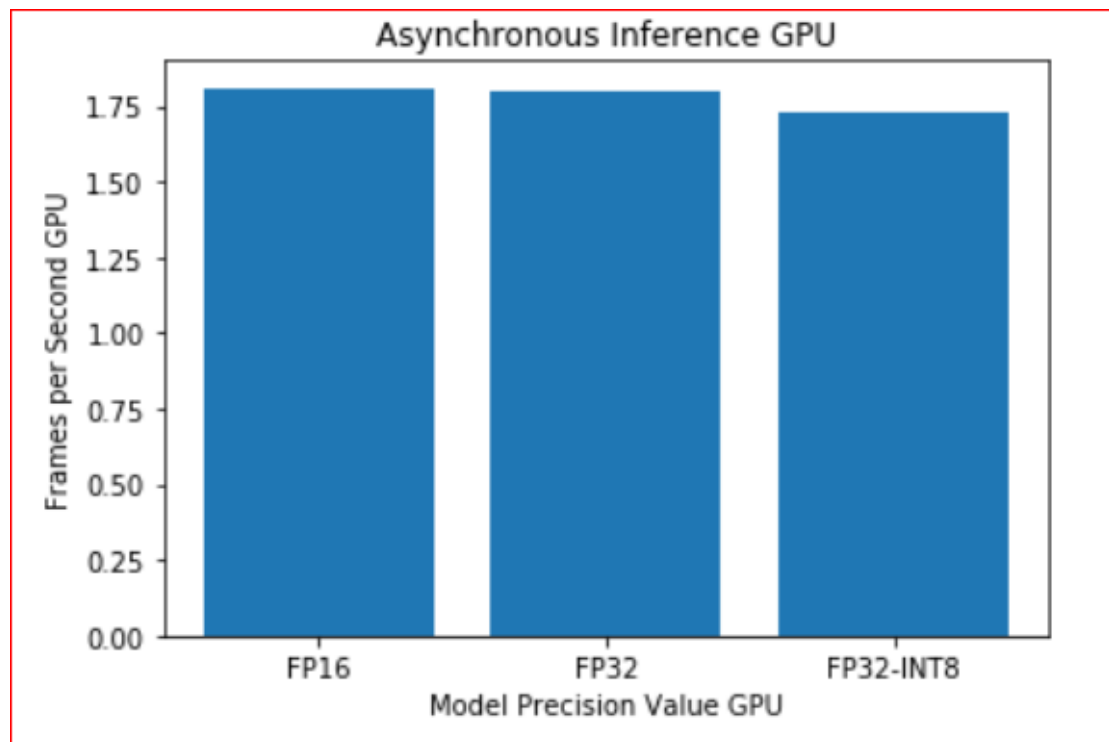
## Benchmark results of the model. GPU

[18]:

	inference_time	model_load_time	Fps	total_time
<b>FP16</b>	32.6	34.921903	1.809816	32.572538
<b>FP32</b>	32.8	33.834617	1.798780	32.822181
<b>FP32INT8</b>	34.1	47.700375	1.730205	34.146677

[19]: `result_transposed.plot.bar();`







Some of the observations from benchmark results:

- Model load time significantly increases when device is switch from CPU to GPU as expected. Model takes considerably longer to load in GPU as we analyzed from prior project study.
- Model load time is reduced when preicison is changed from FP32 to FP16 to INT8. This is because as weight are quantized and reduced in size then load time is also reduced.
- Model inference time is slightly improved when changing precision from FP32 to FP16 and from FP32 to INT8. This is expected as accuracy decrease performance improves. There is slight increase in inference time when changing precision from FP16 to INT8.
- Model inference time is increased when changing device from CPU to GPU. Here we can get better performance only when batch inferencing is done on GPU to leverage full compute capacity of GPU.
- Frames per second is pretty consistent irrespective of precision but it slightly reduces as device is changed from CPU to GPU. Here again fps can be increased on GPU device if batch inferencing is done.
- For `Inference Time` and `FPS`, `FP32` give slightly better results. There is not much difference for this three different models for this two parameters.
- I have tested model for Asynchronous Inference and Synchronous Inference, Asynchronous Inference has better results it has slight improvement in `inference time` and `FPS`

## Edge Cases

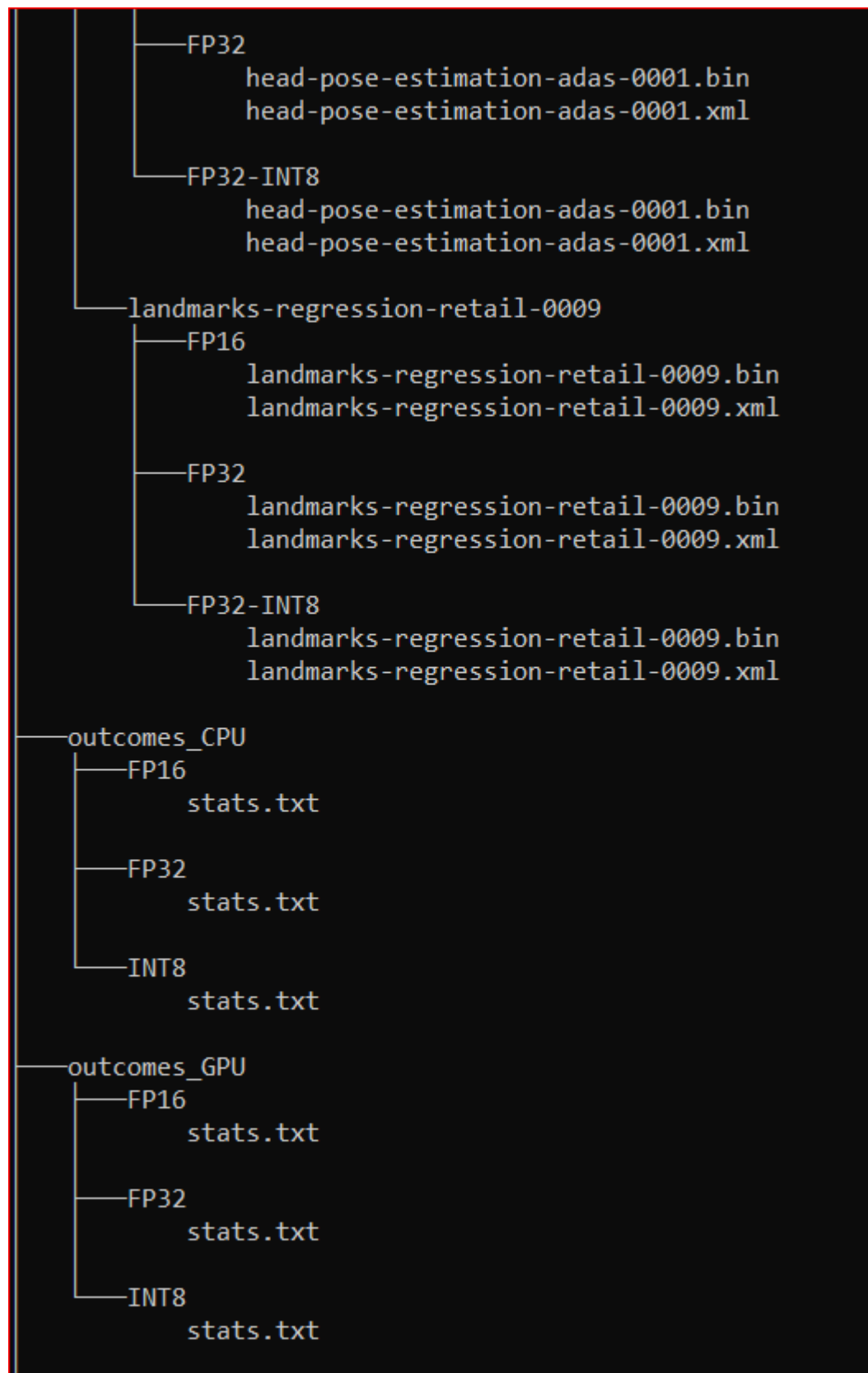
1. The lighting matters greatly for the video feed, so sometimes models does not clearly views the gaze.
2. If for some reason model can not detect the face then it throws off track and pointer continues in same direction.
3. If there are more than one face detected in the frame then model takes the first detected face for control the mouse pointer.





```
C:\UdacityProject\starter>tree /F
Folder PATH listing for volume Windows
Volume serial number is EA48-9AD9
C:..
  Commend Run the project.txt
  Intel(R) UHD Graphics 630 Sunday, July 5, 2020.txt
  P05 Computer Pointer Controller.pdf
  requirements.txt
  Analysis Project
    OpenVino Compute Pointer Controller .html
    OpenVino Compute Pointer Controller .ipynb
  bin
    demo.mp4
    output_video.mp4
  models
    face-detection-adas-0001
      FP16
        face-detection-adas-0001.bin
        face-detection-adas-0001.xml
      FP32
        face-detection-adas-0001.bin
        face-detection-adas-0001.xml
      FP32-INT8
        face-detection-adas-0001.bin
        face-detection-adas-0001.xml
    gaze-estimation-adas-0002
      FP16
        gaze-estimation-adas-0002.bin
        gaze-estimation-adas-0002.xml
      FP32
        gaze-estimation-adas-0002.bin
        gaze-estimation-adas-0002.xml
      FP32-INT8
        gaze-estimation-adas-0002.bin
        gaze-estimation-adas-0002.xml
    head-pose-estimation-adas-0001
      FP16
        head-pose-estimation-adas-0001.bin
        head-pose-estimation-adas-0001.xml
```









```
└── src
    ├── face_detection_model.py
    ├── gaze_estimation_model.py
    ├── head_pose_estimation_model.py
    ├── input_feeder.py
    ├── landmark_detection_model.py
    ├── main.py
    ├── model.py
    ├── mouse_controller.py
    └── output_video.mp4
└── __pycache__
    ├── face_detection_model.cpython-36.pyc
    ├── gaze_estimation_model.cpython-36.pyc
    ├── head_pose_estimation_model.cpython-36.pyc
    ├── input_feeder.cpython-36.pyc
    ├── landmark_detection_model.cpython-36.pyc
    ├── model.cpython-36.pyc
    └── mouse_controller.cpython-36.pyc

C:\UdaciteProject\starter>
```

I hope to be home to the project requirements despite the valuable information that we learned from the lessons and also the project, but we do not know the exact correct result

Help resources Forums :<https://knowledge.udacity.com>  
<https://github.com/>

**I wish success to all.**

Marwan Saeed Alsharabbi

