Computer Pointer Controller

TODO: Write a short introduction to your project

Project Set Up and Installation
TODO: Explain the setup procedures to run your project. For instance, this can include your project directory structure, the models you need to download and where to place them etc. Also include details about how to install the dependencies your project requires.

This demonstration shows the control of the mouse pointer using the gaze estimation model (gaze-estimation-adas-0002).

The demo relies on the following auxiliary networks:

face-detection-adas-0001 detection networks for finding faces.

head-pose-estimation-adas-0001, which estimates head pose in Tait-Bryan angles, serving as an input for gaze estimation model

landmarks-regression-retail-0009, which estimates coordinates of facial landmarks for detected faces. The keypoints at the center of eyes are used to locate eyes regions required for the gaze estimation model

Other demo objectives are:
Video/Camera as inputs, via OpenCV*
Visualization of gaze estimation results, and, optionally, results of inference on auxiliary models

How It Works
The application reads command-line parameters and loads four networks to the Inference Engine
The application gets a frame from the OpenCV VideoCapture
The application performs inference on auxiliary models to obtain head pose angles and images of eyes regions serving as an input for gaze estimation model
The application performs inference on gaze estimation model using inference results of auxiliary models
The application shows the results

Project directory structure

controller_pointer
- bin
  - demo.mp4
- models
  - **face-detection-adas-0001**
    - INT8
      - face-detection-adas-0001.bin
      - face-detection-adas-0001.xml
    - FP16
      - face-detection-adas-0001.bin
      - face-detection-adas-0001.xml
    - FP32
      - face-detection-adas-0001.bin
      - face-detection-adas-0001.xml
  - **head-pose-estimation-adas-0001**
    - INT8
      - head-pose-estimation-adas-0001.bin
      - head-pose-estimation-adas-0001.xml
    - FP16
      - head-pose-estimation-adas-0001.bin
      - head-pose-estimation-adas-0001.xml
    - FP32
      - head-pose-estimation-adas-0001.bin
      - head-pose-estimation-adas-0001.xml
  - **landmarks-regression-retail-0009**
    - INT8
      - landmarks-regression-retail-0009.bin
      - landmarks-regression-retail-0009.xml

FP16

landmarks-regression-retail-0009.bin
landmarks-regression-retail-0009.xml

FP32

landmarks-regression-retail-0009.bin
landmarks-regression-retail-0009.xml

lgaze-estimation-adas-0002

INT8

gaze-estimation-adas-0002.bin
gaze-estimation-adas-0002.xml

FP16

gaze-estimation-adas-0002.bin
gaze-estimation-adas-0002.xml

FP32

gaze-estimation-adas-0002.bin
gaze-estimation-adas-0002.xml

src

base_pointer.py
face_detection.py
facial_landmarks_detection.py
gaze_estimation.py
head_pose_estimation.py
main.py
input_feeder.py
mouse_controller.py

README

requirements.txt

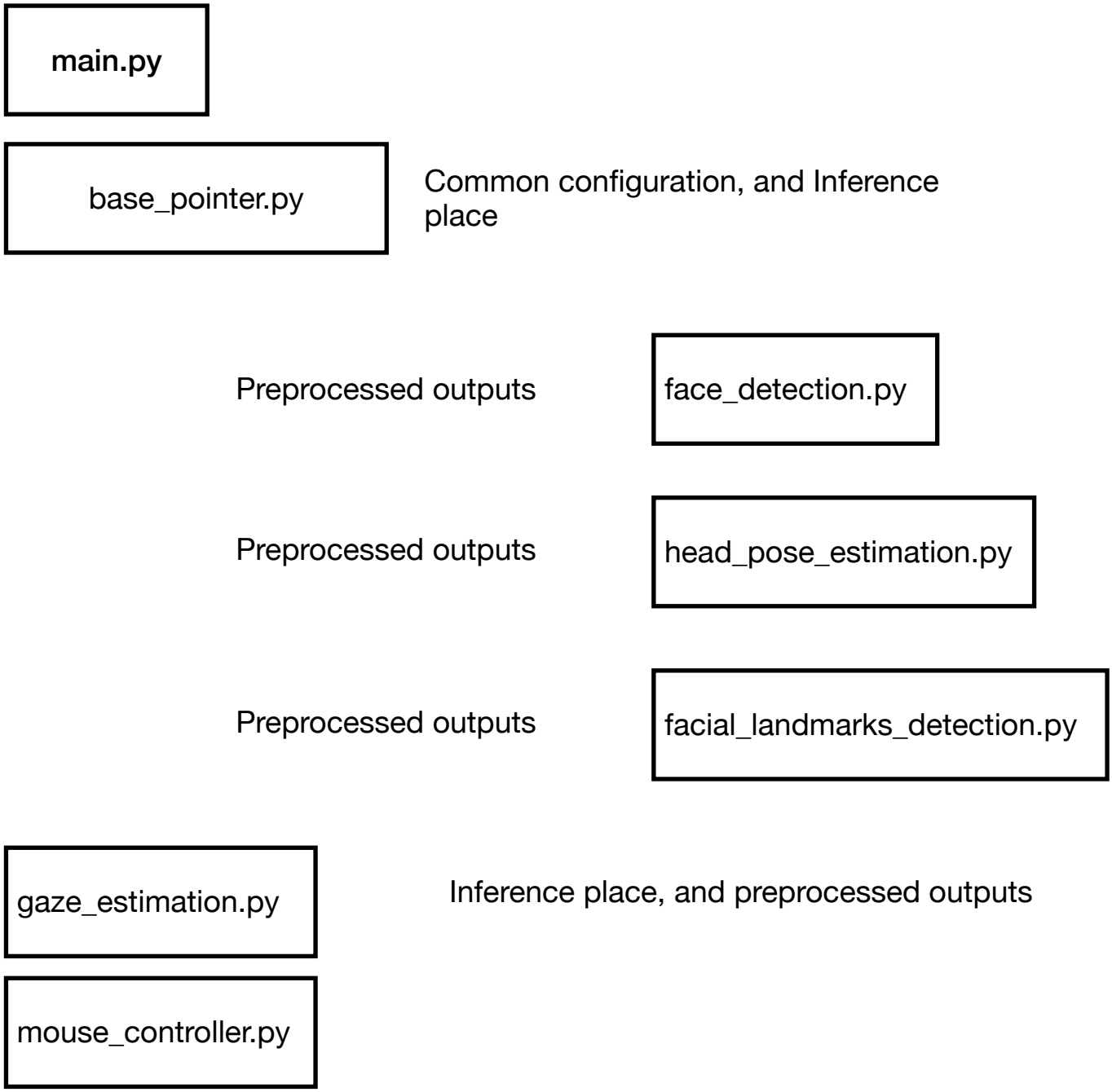
Dependencies
Working with openvino_2020.3.194
pc_test ~
$ brew install python3

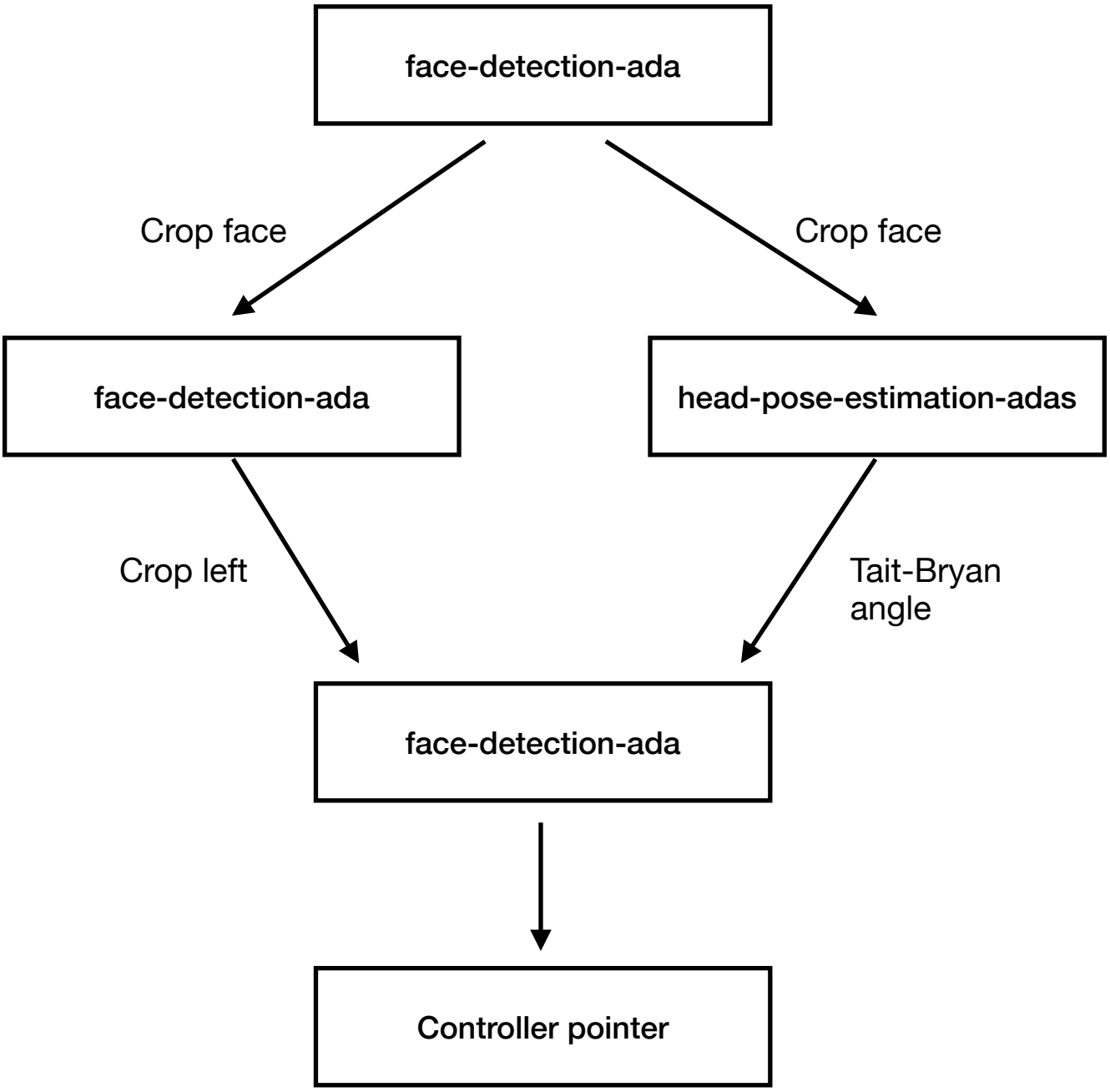
$ nano .bash_profile

export PATH="/usr/local/opt/python/libexec/bin:/usr/local/sbin:
$PATH"
export VIRTUALENVWRAPPER_PYTHON=/usr/local/bin/python3
export VIRTUALENVWRAPPER_VIRTUALENV=/usr/local/bin/
virtualenv
export WORKON_HOME=$HOME/.virtualenvs
export PROJECT_HOME=$HOME/Devel
source /usr/local/bin/virtualenvwrapper.sh
source /opt/intel/openvino/bin/setupvars.sh

$ workon open
(open) pc_test ~
$ Pip3 install pyautogui
$ pip3 install virtualenv virtualenvwrapper

$ cd /opt/intel/openvino/install_dependencies
(open) pc_test install_dependencies
$ sudo -E ./install_openvino_dependencies.sh


main.py
base_pointer.py
Common configuration, and Inference place
Preprocessed outputs
face_detection.py
Preprocessed outputs
head_pose_estimation.py
Preprocessed outputs
facial_landmarks_detection.py
gaze_estimation.py
Inference place, and preprocessed outputs
mouse_controller.py


face-detection-ada
Crop face
Crop face
face-detection-ada
head-pose-estimation-adas
Crop left
Tait-Bryan
angle
face-detection-ada
Controller pointer

Demo

TODO: Explain how to run a basic demo of your model
Running the application with an empty list of options yields an error message.

Step 1 : Download this project.

Step 2 : Initialize the openVINO environment using the following command
source /opt/intel/openvino/bin/setupvars.sh -pyver 3.5

Step 3 : Download the required models by using OpenVINO model downloader (On Mac), and the demo structure.

Face Detection model
curl -O https://download.01.org/opencv/2020/openvinotoolkit/2020.3/open_model_zoo/models_bin/1/face-detection-adas-0001/

Facial Landmark detection model
curl -O https://download.01.org/opencv/2020/openvinotoolkit/2020.3/open_model_zoo/models_bin/1/landmarks-regression-retail-0009/

Head pose Detection model
curl -O https://download.01.org/opencv/2020/openvinotoolkit/2020.3/open_model_zoo/models_bin/1/head-pose-estimation-adas-0001/

Gaze Estimation model
curl -O https://download.01.org/opencv/2020/openvinotoolkit/2020.3/open_model_zoo/models_bin/1/gaze-estimation-adas-0002/

To run the demo, you can use public or pre-trained and optimized gaze-estimation-adas-0002 model, and the auxiliary models. To download the pre-trained models, use the OpenVINO Model Downloader or go to https://download.01.org/opencv/

For example, to do inference on a CPU, run the following command:

python3 /controller_pointer/src/main.py -m1 Desktop/project/controller_pointer/models/face-detection-adas-0001/INT8/face-detection-adas-0001.xml -m2 Desktop/project/controller_pointer/models/head-pose-estimation-adas-0001/INT8/head-pose-estimation-adas-0001.xml -m3 Desktop/project/controller_pointer/models/landmarks-regression-retail-0009/INT8/landmarks-regression-retail-0009.xml -m4 Desktop/project/controller_pointer/models/gaze-estimation-adas-0002/INT8/gaze-estimation-adas-0002.xml -i Desktop/project/starter/bin/demo.mp4 -t video

Documentation
TODO: Include any documentation that users might need to better understand your project code. For instance, this is a good place to explain the command line arguments that your project supports.

python3 /controller_pointer/src/main.py --help

optional arguments:
```
  -h, --help            show this help message and exit
  -m1 MODEL1, --model1 MODEL1
                        Path to an xml file with a trained model1.
  -m2 MODEL2, --model2 MODEL2
                        Path to an xml file with a trained model2.
  -m3 MODEL3, --model3 MODEL3
                        Path to an xml file with a trained model3.
  -m4 MODEL4, --model4 MODEL4
                        Path to an xml file with a trained model4.
  -i INPUT_FILE, --input_file INPUT_FILE
                        Path to video file
  -t INPUT_TYPE, --input_type INPUT_TYPE
                        video, cam
  -d DEVICE, --device DEVICE
                        Specify the target device to infer on: CPU
  -o OUTPUT_INTERMEDIATE_MODEL, --output_intermediate_model
OUTPUT_INTERMEDIATE_MODEL
                        Outputs of intermediate models
  -pt PROB_THRESHOLD, --prob_threshold PROB_THRESHOLD
                        Probability threshold for detections filtering(0.5 by
                        default)
```

Benchmarks
TODO: Include the benchmark results of running your model on multiple hardwares and multiple model precisions. Your benchmarks can include: model loading time, input/output processing time, model inference time etc.
On CPU

	FP16 facial detection	FP32 facial detection	INT8 facial detection
Inference time	25.31	26.99	56.23
	FP16 landmark detection	FP32 landmark detection	INT8 landmark detection
Inference time	0.60	0.58	1.23
	FP16 head pose detection	FP32 head pose detection	INT8 head pose detection
Inference time	1.78	1.83	3.20
	FP16 gaze estimation	FP32 gaze estimation	INT8 gaze estimation
Inference time	5.37	5.45	9.19
	FP16 facial detection	FP32 facial detection	INT8 facial detection
loading time	0.62	0.75	1.61
	FP16 landmark detection	FP32 landmark detection	INT8 landmark detection
loading time	0.16	0.15	0.45
	FP16 head pose detection	FP32 head pose detection	INT8 head pose detection
loading time	0.17	0.16	0.46
	FP16 gaze estimation	FP32 gaze estimation	INT8 gaze estimation
loading time	0.20	0.19	0.56

Results
TODO: Discuss the benchmark results and explain why you are getting the results you are getting. For instance, explain why there is difference in inference time for FP32, FP16 and INT8 models.

The expectation was that for a lower degree of precision, such as in the case of INT8, the inference and loading time would be less than those of FP32 and FP16, however I observe that they are greater. This allows me to

conclude that the results depend on the architecture of the processors, since if they are designed to perform 16 and 32-bit operations, then performing 8-bit operations could take additional time since the processor would have to organize the entrances to carry out the operations according to the characteristics of each of the architectures. (16, 32, 64 bit).
The comparison between the accuracies for FP16 and FP32 in relation to inference and load times, I observe that they are close, with a tendency to be slightly higher for FP32, which I consider is expected since the time to execute the operations should be more extensive. However, the inference and loading times are very similar. In conclusion, the recommendation given to us that it is necessary to test performance and make optimizations on architectures in a real environment (as is the case with the Intel cloud) is necessary since this allows us to have a greater degree of certainty about the results that we could obtain in the final assembly for each processor architecture.

Stand Out Suggestions
This is where you can provide information about the stand out suggestions that you have attempted.

Async Inference
If you have used Async Inference in your code, benchmark the results and explain its effects on power and performance of your project.

The proposed demo is based on synchronous interference.

Edge Cases
There will be certain situations that will break your inference flow. For instance, lighting changes or multiple people in the frame. Explain some of the edge cases you encountered in your project and how you solved them to make your project more robust.

In this demo you could experience interruption in your flow for cases such as where more than one person appears, since this was built only for the inference of a single person. For this case to be added, it is suggested separately to cut out each of the faces of the people and to carry out the inference of the other networks separately.

For those cases where there are variations in illumination, the inference flux could also be stopped since these effects have not been considered in this demo. As a result, the inference will possibly give an unexpected result, as it can happen in cases where there are shadows. One way to consider it is possible to improve the results in this case, could be the addition of multiple images with variations and lighting effects on each one, in order to re-train the models, however this possibly increases the times load, and inference affected performance.