**Chapter Four**

**SYSTEM DESIGN AND IMPLEMENTATION**

**4.1 Introduction**

In this chapter, I highlighted some of the system algorithm and pseudocode. User interfaces and also the tools that help in the implementation of the system. And discussion followed at the end.

**4.2 System Design**

The design phase of any system is very important, vital and crucial because the success of any system depends largely on its design specifications. In this phase, the final specifications are used for translating the model into a design of the desired system and modules are being defined showing their relationships to one another in a way known as a structural chart using structured tools. The reason for the design phase is to specify a particular software system that will meet the requirements gathered at the analysis phase. Structured design divides a program into smaller, independent modules. They are arranged orderly in a hierarchy that shows a model of the application area which is organized in a top-down manner.

The concept of modification thus comes from structured design which is an attempt to reduce complexity and make a problem manageable by sub-dividing it into smaller segments.

**4.3 Physical Design**

1. **Input Design**

This is an interface between the user and the system that allows the user to enter data. Image input is generally done through either the webcam or file upload.

1. **Output Design**

This serves as an interface between the user and the system that provide report to the user.

**4.4 Application Algorithm**

In mathematics and computer science, an algorithm is an effective method expressed as a finite list of well-defined instructions for calculating a function. Algorithms are used for calculation, data processing, and automated reasoning.

Starting from an initial state and initial input (perhaps null), the instructions describe a computation that, when executed, will proceed through a finite number of well-defined successive states, eventually producing "output" and terminating at a final ending state. The transition from one state to the next is not necessarily deterministic; some algorithms, known as randomized algorithms, incorporate random input.

**step 1:** Open the webcam or accept the image as a file.

**Step 2:** Restart**/**Start reading time.

**Step 3:** Read image frame from webcam

**Step 4:** Use OpenCV library to find the face in the frame and if there is no face detected go to Step 2.

**Step 5:** Use ageNet Convolutional Neural Networks (forward) to estimate the age.

**Step 6:** Use genderNet Convolutional Neural Networks (forward) to estimate the Gender.

**Step 7:** Display age and Gender.

**Step 8:** Display accuracy of both age and Gender.

**Step 9** Display time taken to estimate age and Gender.

**Step 10:** Go to Step 1 for another frame or quit.

**4.4.1 Pseudocode**

while camera is open or image file imported

start timer

get frame

if no frame

break

get box over the subject face

if no box detected //meaning that no face detected

continue

for every face in the box

extract face features

insert the face features as input of caffemodel

//for gender

use caffemodel to estimate the gender of face

display estimated gender

display the accuracy of the estimated gender

//for age

use caffemodel to estimate the age of face

display estimated age

display the accuracy of the estimated age

display time taken to estimate

if webcam is open

continue

else

break

**4.5 User Interface**

**4.5.1 Splash Screen**

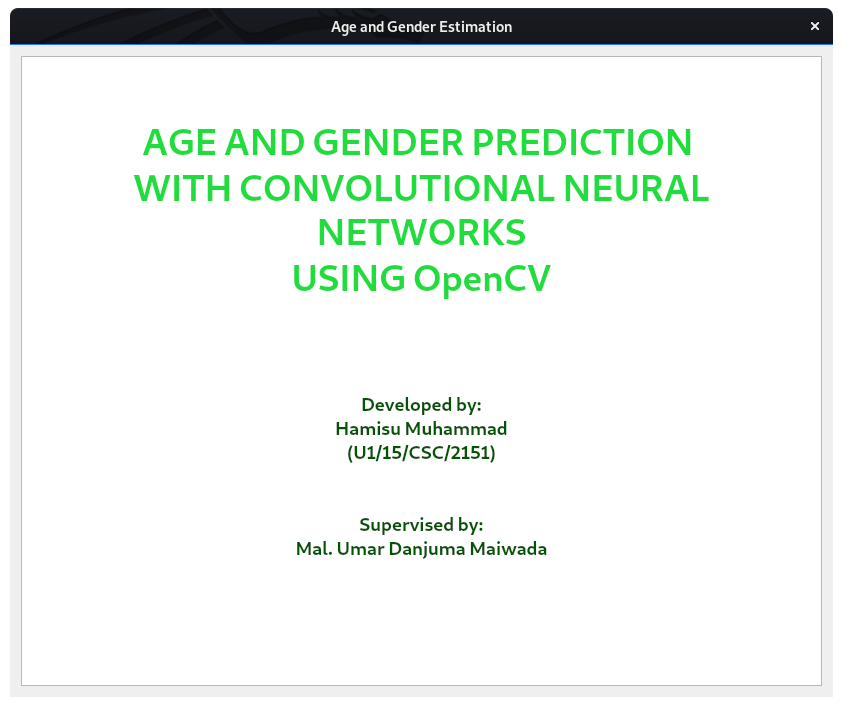
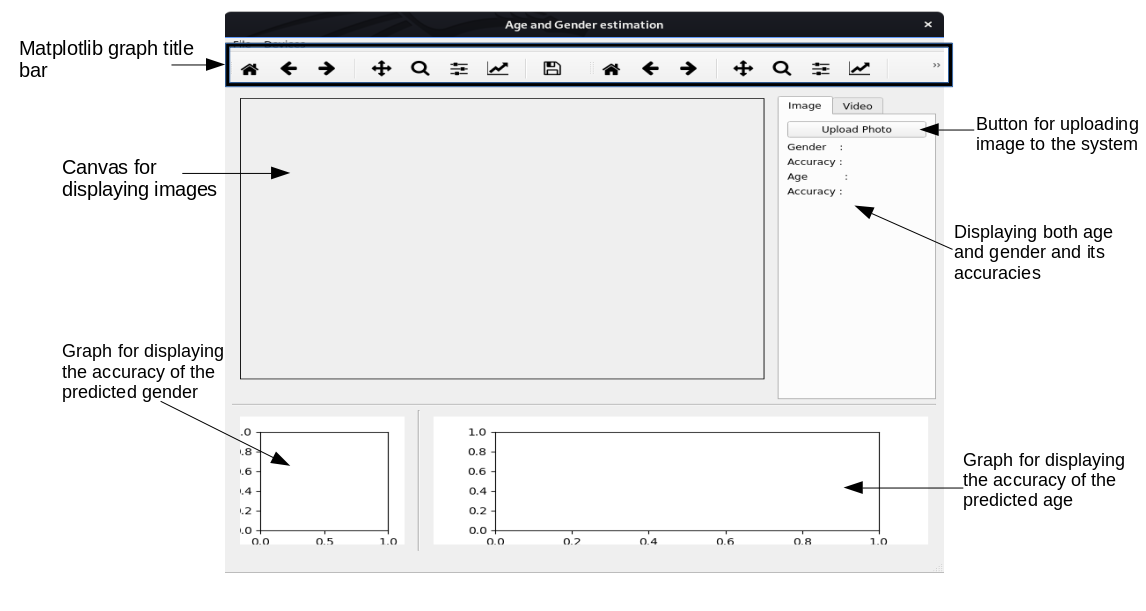
****

Figure 4.1 Splash screen for stating the system

**4.5.2 Main Menu**

Figure 4.2 system menu for image estimation

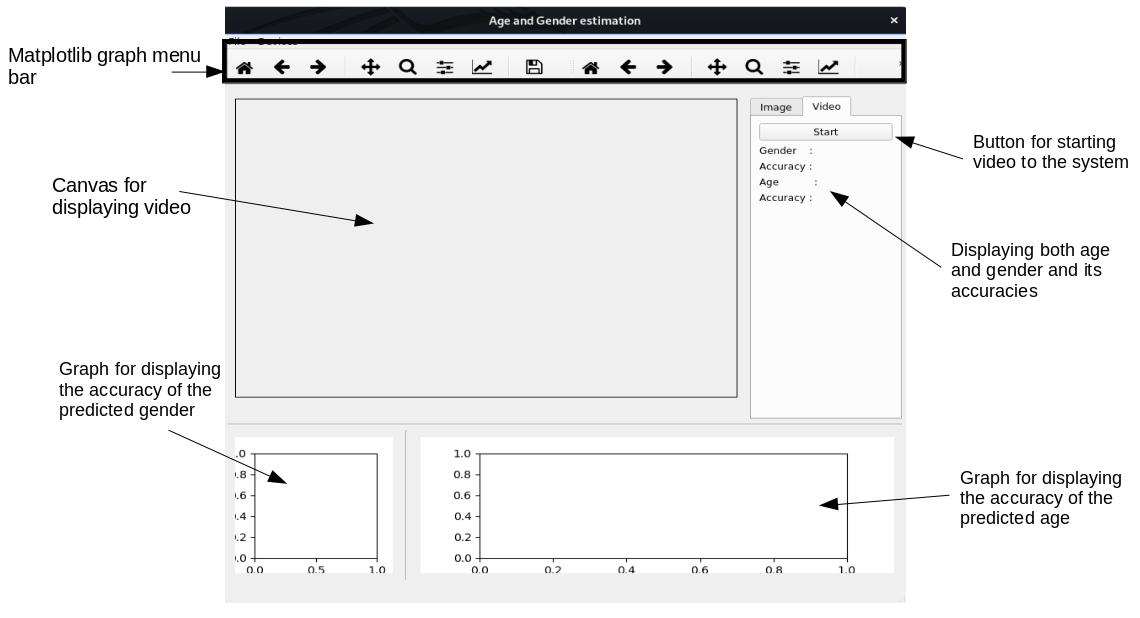
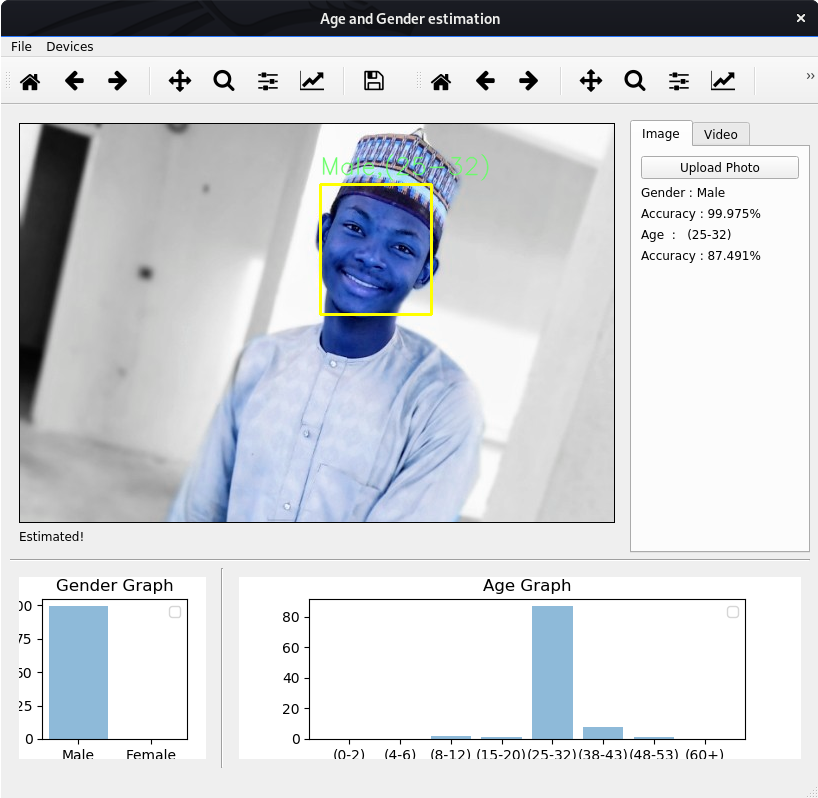
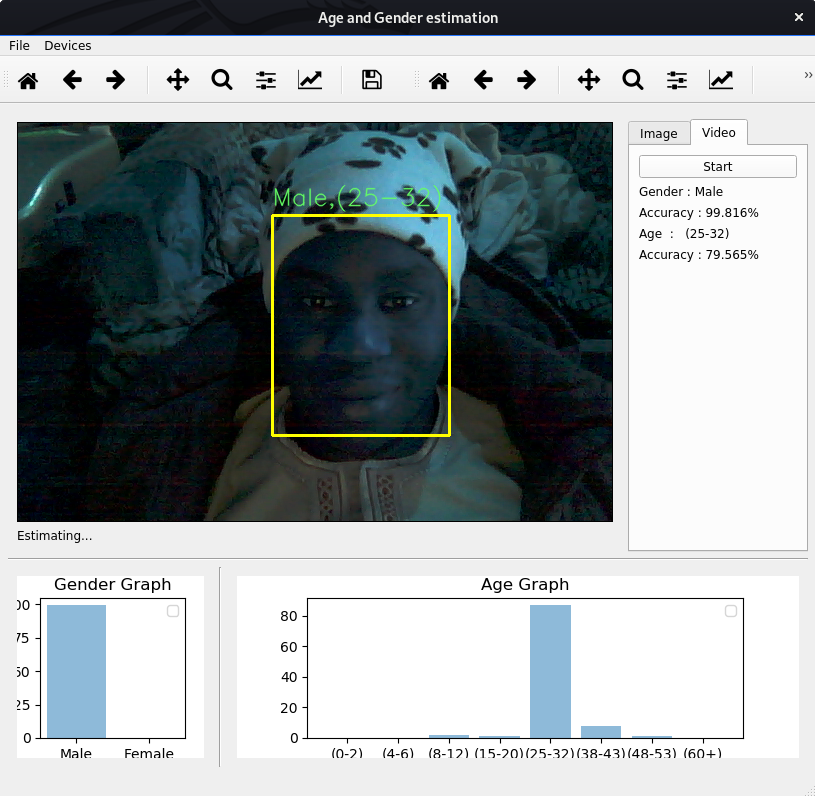
****

Figure. 4.3 system menu for real-time estimation

Figure 4.4 Testing the system with image

Figure 4.5 Real-time estimation

**4.6 System Requirements**

These are set of tools needed by the system to operate, as its design. The proper operation of the newly designed system depends on these requirements. These requirements are in two classes, **hardware** and **software** requirement.

* **Minimum Hardware requirement**

This is the physical component needed by the system to operate. The software require moderate hardware to manage multiple processing. This software requires a minimum of:

* + - Build-in camera of a laptop (used as image input device)
    - Pentium processor 2.27 GHz
    - 4 GB of RAM
    - 100 GB of storage
* **Minimum Software requirement**

This is the non-physical component needed by the system to make it operable.

* + - **Windows operating systems:** Windows 7, windows 8, windows 10 or latter version with Python and other dependencies installed.
    - **Linux Operating System:** Ubuntu, Parrot, Kali, etc.

**4.7 Implementation Tools**

Below are some of the tools used to run the project design, which includes:

* **Python Programming Language**

*Python* is a programming language with extensive supported packages and modules. It is developed by Guido van Rossum. It is derived from many other languages such as ABC, modula-3, C, C++, Algol-68, smallTalk, Unix shell and other scripting languages. *Python* also provides interfaces to all major commercial databases (Swaroop, 2003). *Python* consists of a broad standard library. This feature enables the exploration and access to various file types such as XML, HTML, WAV, CSV files. IDLE is a simple *Python* Integrated Development Environment (IDE) available for Windows, Linux, and Mac OS X (Lent, 2013). All commands can be typed, saved and run in *Python* IDLE iterative shell. As such, *Python* is chosen as programming language throughout the project.

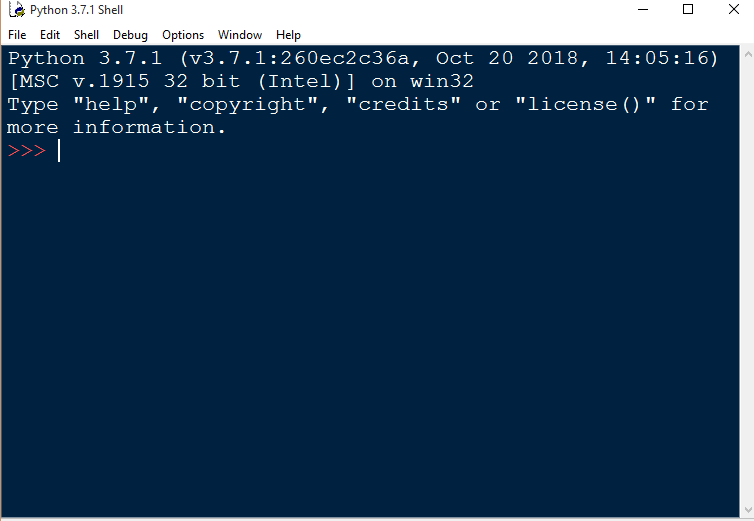
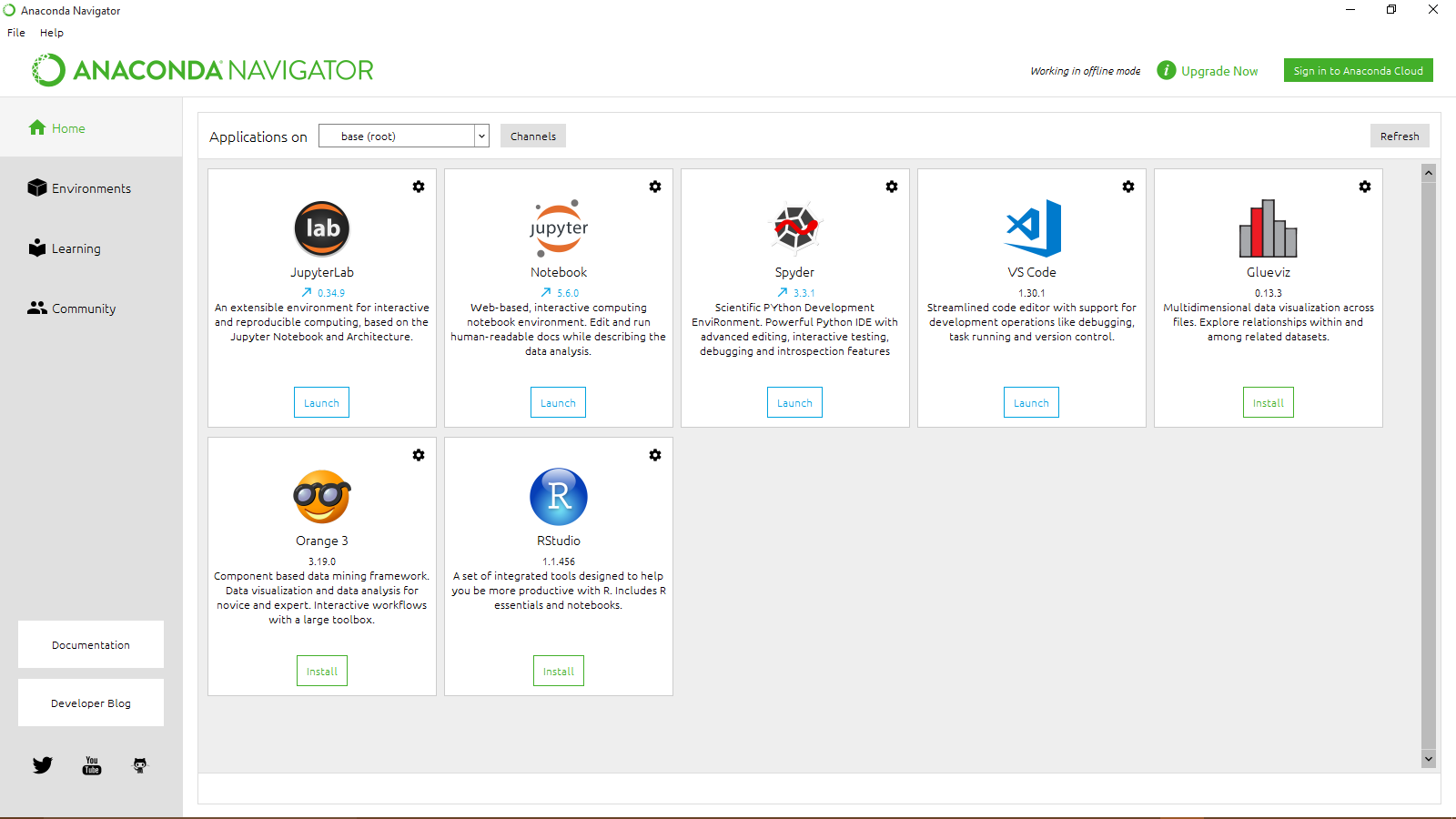


Figure 4.6 Python IDLE (Python shell)

* **Anaconda (Python) Distribution**

**Anaconda** is a free and open-source distribution of the *Python* and *R* programming languages for scientific computing (data science, machine learning applications, large-scale data processing, predictive analytics, etc.), that aims to simplify package management and deployment. Package versions are managed by the package management system *conda*. The Anaconda distribution is used by over 15 million users and includes more than 1500 popular data-science packages suitable for Windows, Linux, and MacOS.



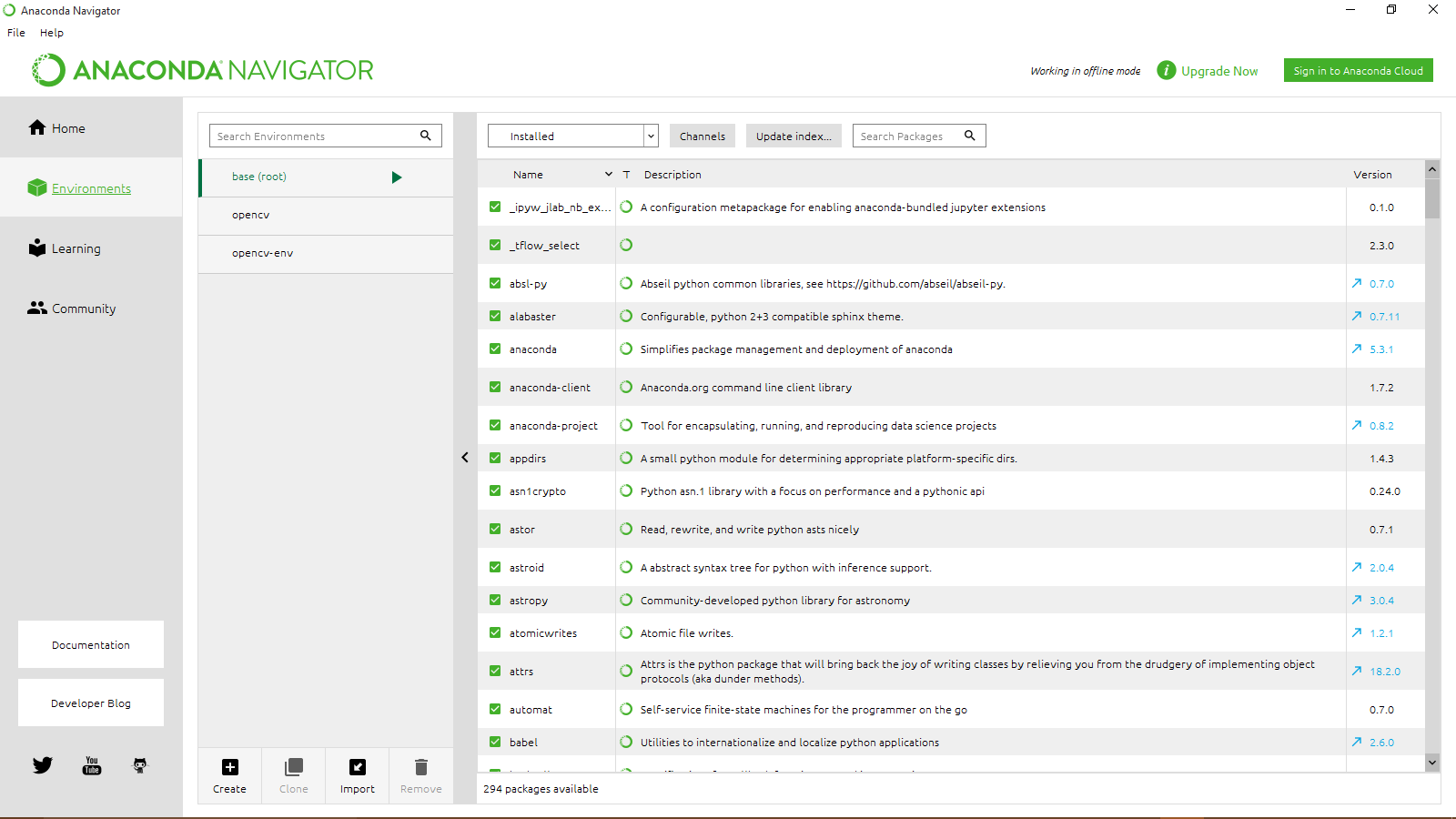


Figure 4.8 Anaconda Navigator List of Packages

* **Open Source Computer Vision Library (OpenCV)**

**OpenCV (Open Source Computer Vision Library)** is a library of programming functions mainly aimed at real time computer vision, developed by Intel. The library is cross-platform. It focuses mainly on real-time image processing. Python, Ruby and Java (using JavaCV) have been developed to encourage adoption by a wider audience. However, since version 2.0, OpenCV includes both its traditional C interface as well as a new C++ interface. This new interface seeks to reduce the number of lines of code necessary to code up vision functionality as well as reduce common programming errors such as memory leaks (through automatic data allocation and deallocation) that can arise when using OpenCV in C. Most of the new developments and algorithms in OpenCV are now developed in the C++ interface. Unfortunately, it is much more difficult to provide wrappers in other languages to C++ code as opposed to C code; therefore the other language wrappers are generally lacking some of the newer OpenCV 2.0 features.

**4.8 Discussion**

During the implementation of the estimation system, the input was derived from the laptop built-in camera. This had affected the output of the estimation system.