**Kinect + Camera Acquisition:**

1. The code gives the user a time of ‘c3’ frames at a rate of 30 frames/sec to settle for the pose, and then records the 3D body coordinates.
2. The 3D coordinates input from the Kinect are displayed as a point cloud, using the openGL library. They are saved in csv format in the file- ‘kinecdata.csv’, which has the locations of all the 25 3D coordinates in order.
3. Initially, the Kinect for Windows v2 SDK has to be installed (before connecting Kinect with PC for the first time).
4. The drivers of Kinect would auto-install when it is connected to the PC for the first time after the installation of SDKs.
5. The steps for setup with the visual studio are given in the following links:

* Initial setup of the Kinect- <http://www.cs.princeton.edu/~edwardz/tutorials/kinect2/kinect0.html>
* Glew- <http://www.cs.princeton.edu/~edwardz/tutorials/kinect2/kinect3.html>
* Body tracking explanation- <http://www.cs.princeton.edu/~edwardz/tutorials/kinect2/kinect4.html>

1. The attached source.cpp file is the one which would replace the already existing source file in 3D\_1 Visual Studio project.

**OpenCV (building applications inside visual studio):**

* Linking with the visual studio: <http://docs.opencv.org/3.0-beta/doc/tutorials/introduction/windows_visual_studio_Opencv/windows_visual_studio_Opencv.html>
* Installing the opencv: <http://docs.opencv.org/3.0-beta/doc/tutorials/introduction/windows_install/windows_install.html#windowssetpathandenviromentvariable>

**Qt GUI:**

1. The folder included is a Qt project. It should be opened using an installed Qt creator if any editing in the code is to be done.
2. The data is stored in the file ‘**outt2.csv**’ in the people\_all folder, and the format is same as that captured by the Kinect folder.
3. For direct use of the app, open the ‘Deployment’ folder and click on ‘GUI\_ImageViewer.exe’.
4. If more images are to be added to the database, they have to be renamed in the same manner as they are present in the people\_all folder, i.e., the name should be of the form ‘im0xxx.jpg’ (starting from im0100.jpg).
5. The variables- *lastimg* and *numimg* in source.cppwould be changed in accordance with any modifications in the present database.
6. The outputs stored from the GUI can be converted to input mat file ‘labels.mat ’ so that it can be directly used for training by the Ramanan codes.
7. Run the script ‘dataform.m’ in matlab. It makes the above conversion.
8. Now, this people\_all folder can be used as it is to replace the one already present in Ramanan code folders.

**Ramanan 2006:**

1. The two papers from January and December account for training and testing of an algorithm which is being used to estimate human body poses from static 2D images, from the PARSE dataset.
2. The file parsePerson.m is the one which is run for testing and the query image can be changed by editing the name in this file only.
3. The attempt to code the training part is included in ‘gradientAscent.m’ and the accompanying functions.

**Ramanan 2011:**

1. The code has the readme included for the learning as well as the demonstration part.

**Matlab (for compiling C files in MATLAB R2012a):**

1. First uninstall the Microsoft Visual Studio C++ 2010 (x86 and x64) redistributable compilers.
2. Download and install the Windows v 7.1 SDK.
3. On the command window-

* Enter ‘mex -setup’
* Select the compiler installed above, and carry on with the compilation of the C files using the mex command.