Getting started with the
Intel RealSense SDK and MATLAB

Team members:

Aditya Sharma

Mukul Javadekar

Faculty Advisor:

Prof. Murali Subbarao

This is a preliminary draft of the guidebook for working with Intel RealSense camera and MATLAB. This guide consists of three main parts: Installing Intel RealSense SDK 2.0, how to run code in C++ in Visual Studio and how to run the MATLAB code.

You will find different bugs while executing the codes specially on MATLAB. These may be due to the SDK and the sample functions provided by Intel. Please report it to professor as you observe it.

1. INSTALLING INTEL SDK:

1.1 Go to the latest stable release, navigate to the Assets section, download depending on your OS and its version and run Intel.RealSense.SDK.exe:

1.2 For Windows you might get a window stating that: "Windows protected you PC" as shown in Fig1. Just click on "More info" and click on Run Anyway.
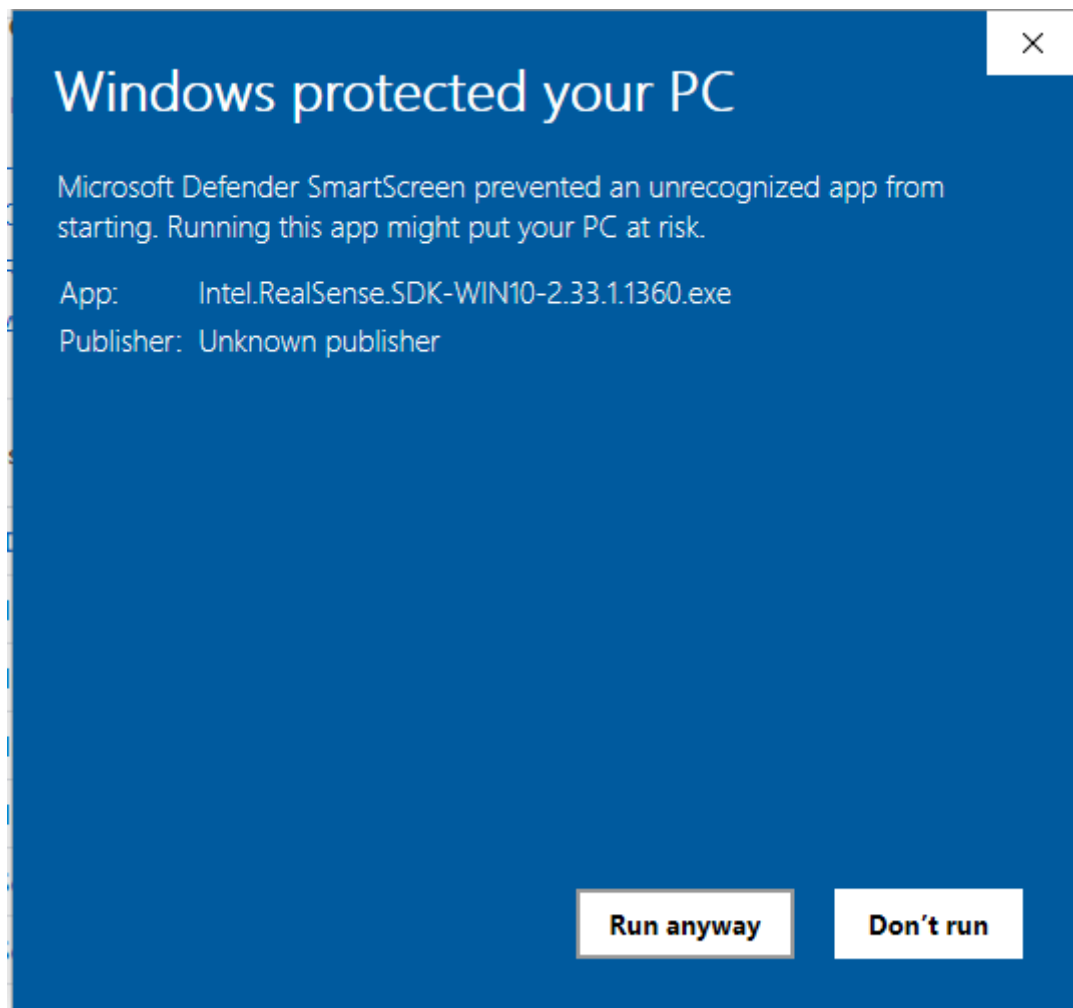

Fig1.1

1.3 Intel RealSense SDK 2.0 is distributed under the Apache 2.0 permissive open-source license: You will get a setup wizard (Fig1.2). Click on "Next". After this you need to select which components you want to install. Select Full Installation as shown in Fig1.3. Click Next. Select create a desktop shortcut (Fig1.4). Click Next. Review it before installing (Fig1.5). It takes 1-2 mins for installation. Check Launch Intel RealSense Viewer and click Finish.
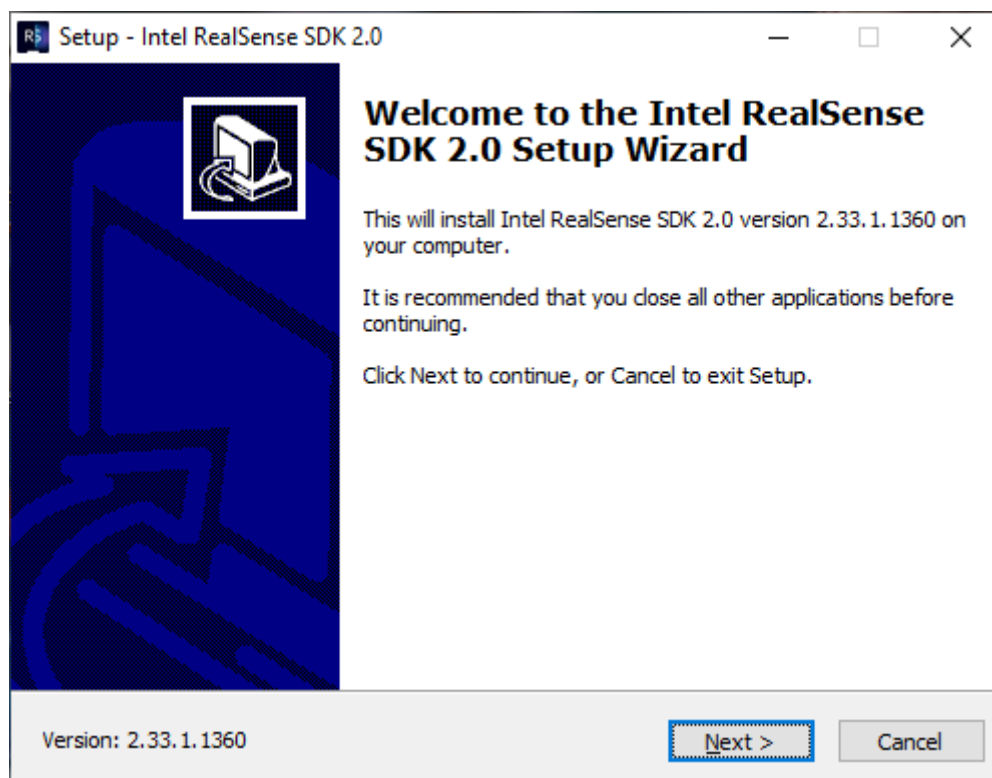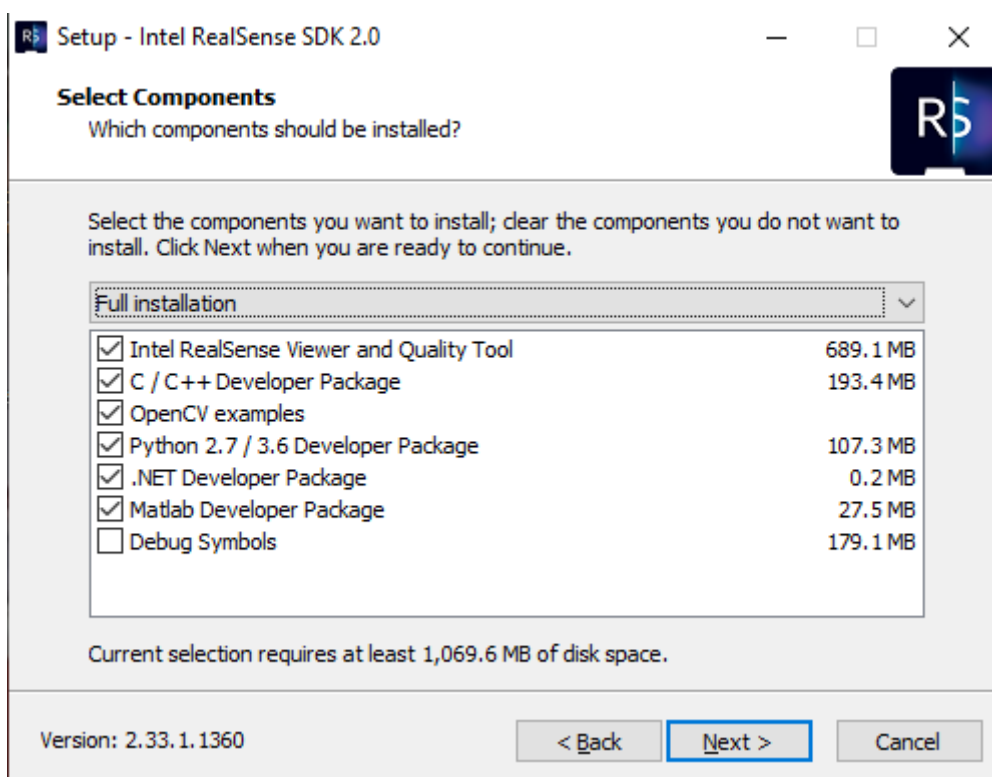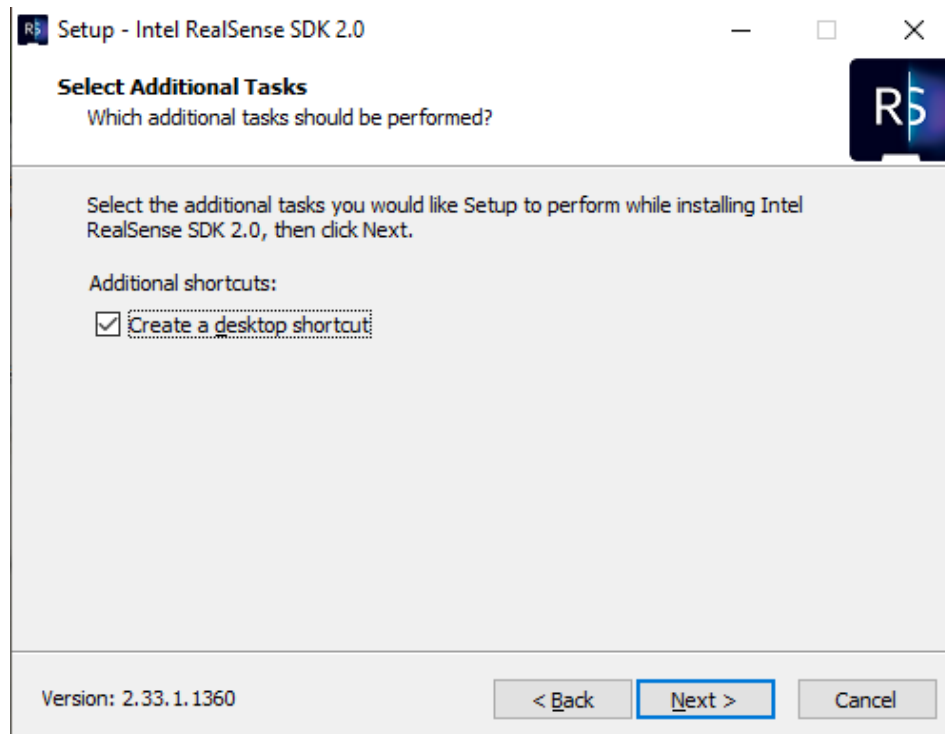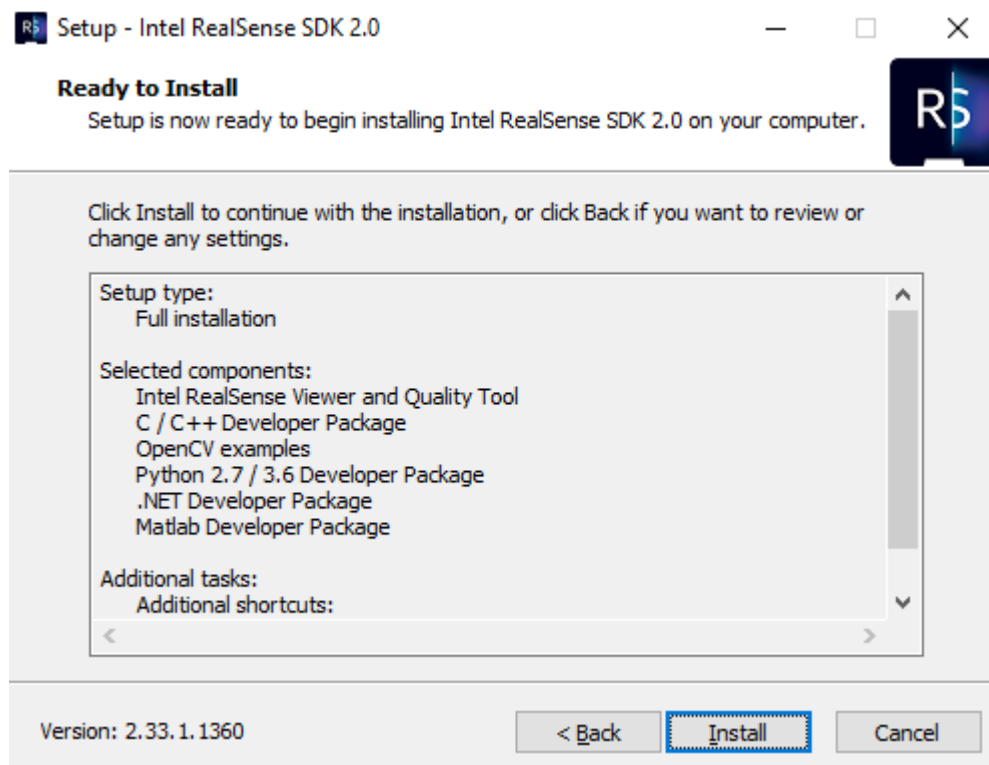
Fig1.2

Fig1.3

Fig1.4

Fig1.5

2. RUNNING SAMPLE CODE IN C++

2.1 After installing SDK, we will look how to run the C++ sample code provided by the Intel RealSense. After installing SDK, you might have got the icon as Intel RealSense viewer (Fig2.1). Connect the camera to your laptop using USB cable. It is a USB 3.1 cable. Double click on the icon to open it.
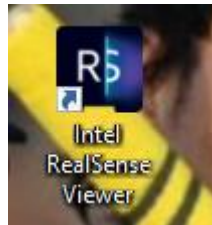

Fig2.1

2.2 After opening the RealSense viewer, on the left side you will see that it is connected and shows some features like Stereo module, RGB camera, Motion Module etc. (Fig2.2).
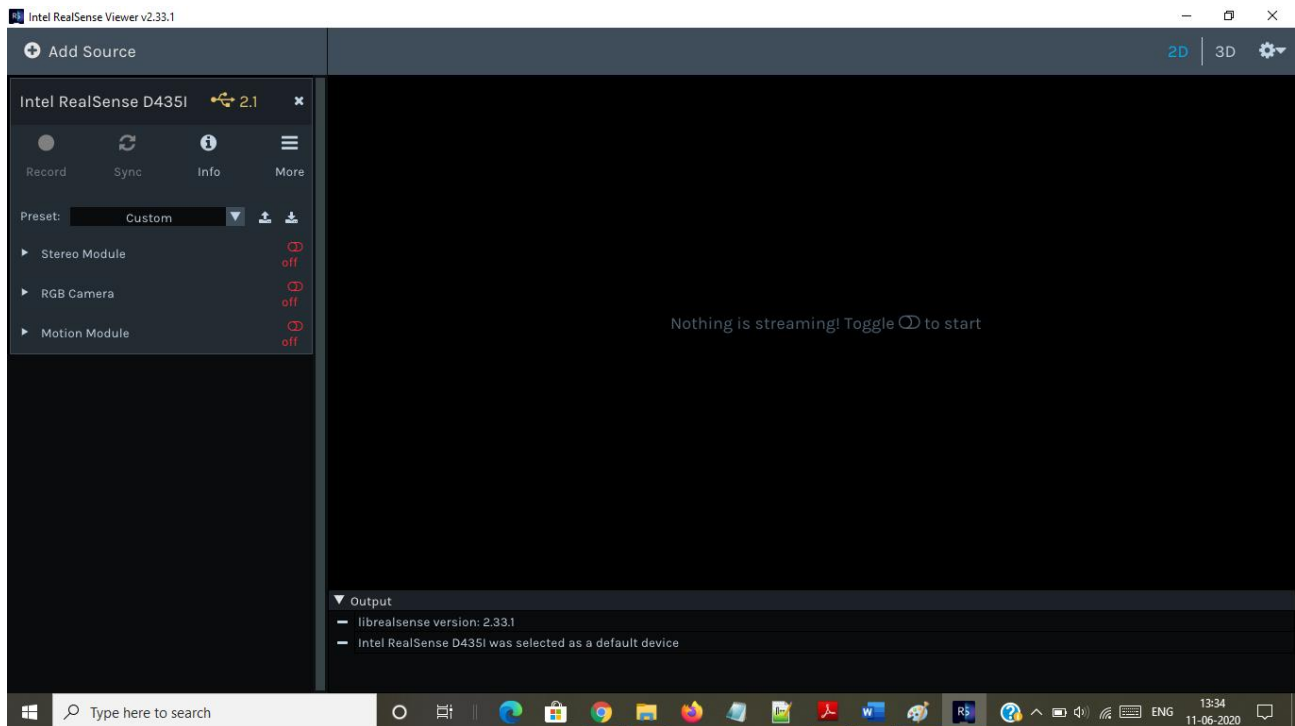
Fig2.2

2.3 Stereo Module shows the Depth stream of the image. RGB camera shows the RGB color stream and Motion Module displays the Accel stream and Gyro stream. You need to toggle these so that the camera starts streaming the image (Fig2.3). Stream it for 3-4 seconds then close it. Now open the icon on the desktop which says Examples for Intel RealSense SDK. For that you need to install Visual Studio as well – link: https://visualstudio.microsoft.com/downloads/
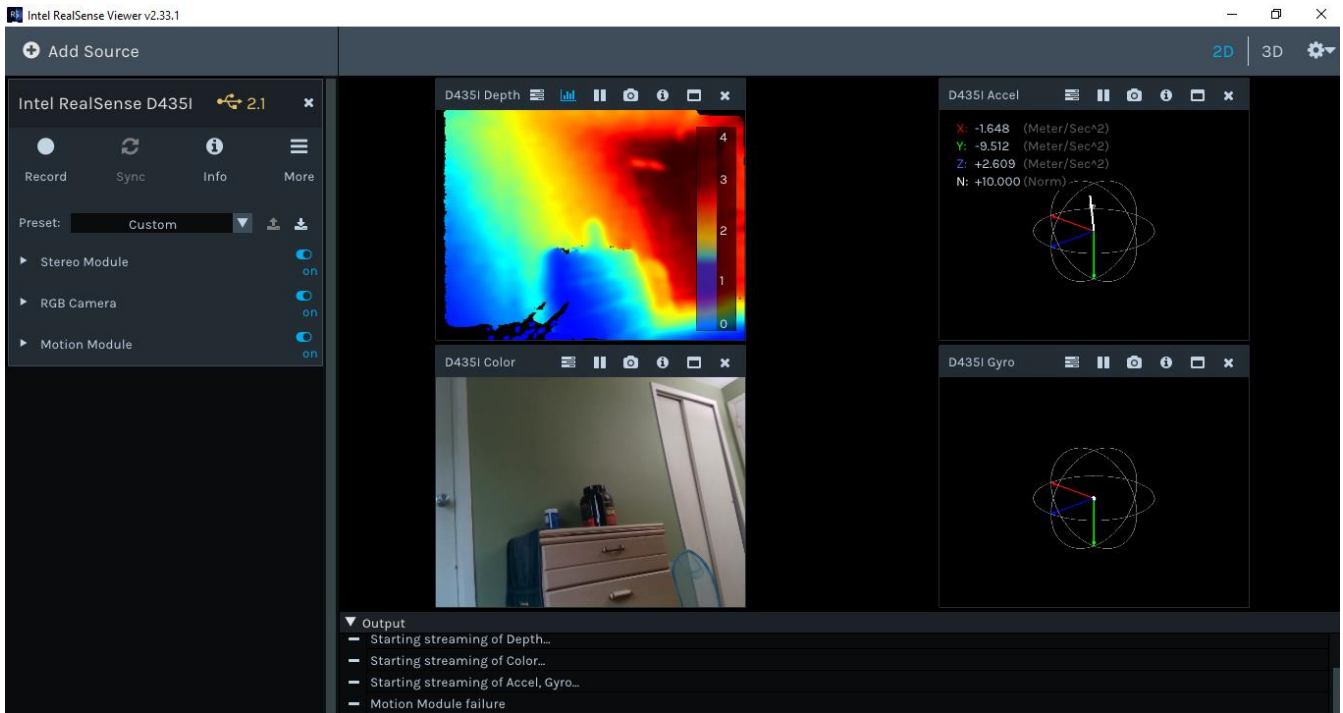
Fig2.3

2.4 After installing the latest version of visual studio open the icon which states examples for intel RealSense and press F5. It builds, compiles and runs the code align.exe. Building will generate and run align.exe application in the backend and you will be able to see output (Fig2.4) which displays the alignment of Depth and RGB image provided the slider down because of which you can adjust the depth and RGB streaming of an image.
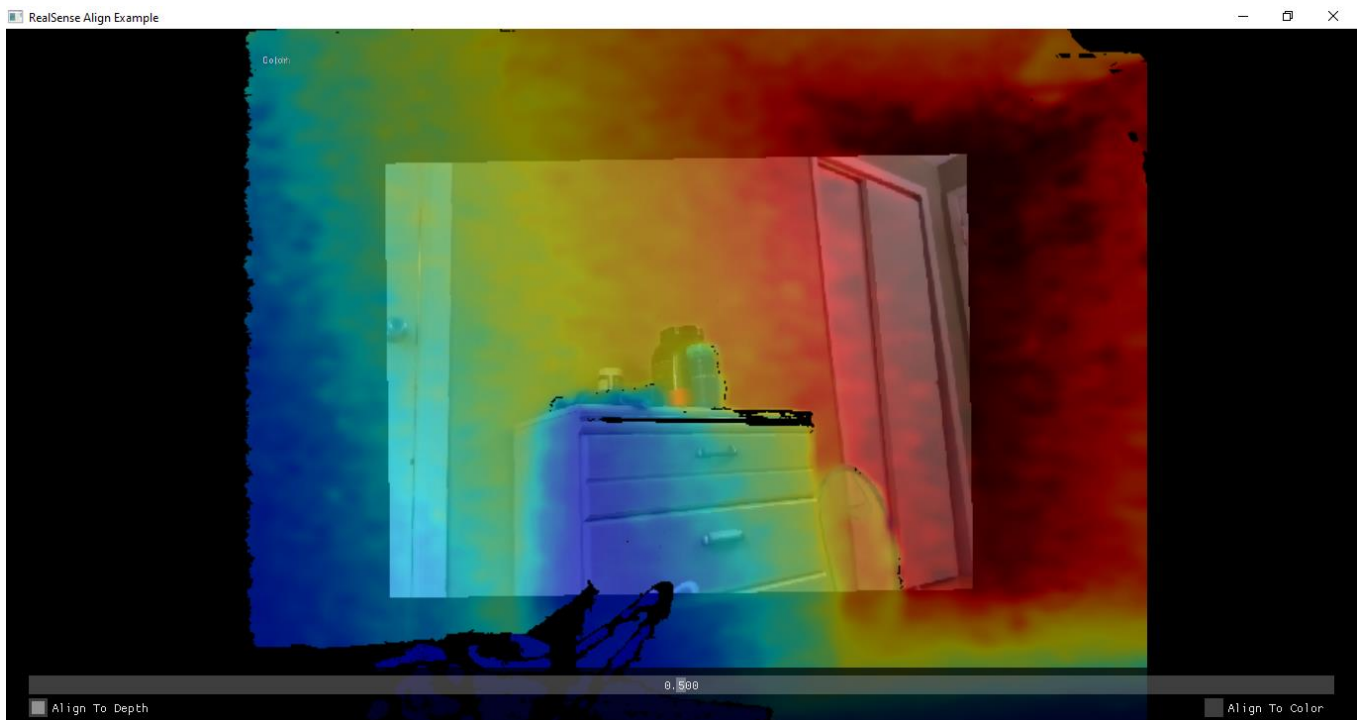

Fig2.4

2.5 To run individual code in visual studio, double click on that particular project, go to Project tab, click on "Set as Startup Project" and then press F5 to run. Before doing that make sure you build the project and then set as startup project as building project will create an .exe application file which will run after you press F5.

3. RUNNING SAMPLE CODE IN MATLAB

3.1 Install MATLAB latest version. Install Image Processing Toolkit while installing MATLAB. After installing MATLAB, again follow the steps which you followed for C++. Open MATLAB, click on Open traverse to the path where the examples are saved i.e. "C:\Program Files (x86)\Intel RealSense SDK 2.0\matlab\+realsense" and select pointcloud example (Fig3.1).

3.2 As stated above, start SDK and stream data, close it and then run MATLAB pointcloud example. You will be asked to select path. Select add to path or where the MATLAB examples are saved (Fig3.2). Mostly you will have to select Add to path as you already have specified the path while creating the project. You will get the output as shown in Fig3.3.

3.3 To get the depth values and co-ordinates of the pointcloud or image, you can just write that data into a csv file by adding one small code in the code (Fig3.4)

3.4 To get the RGB value of the pixel of the image write just simple code as shown in Fig (3.5)

3.4 After running this, try to run other examples in the sample list. Now, we want to align the depth and RGB image as we did in C++ code.

3.5 To do this run the test1alignnew.m example. Create a new script by clicking on New and selecting New script or simply click on + sign besides pointcloud example to get new script tab (Fig3.6).
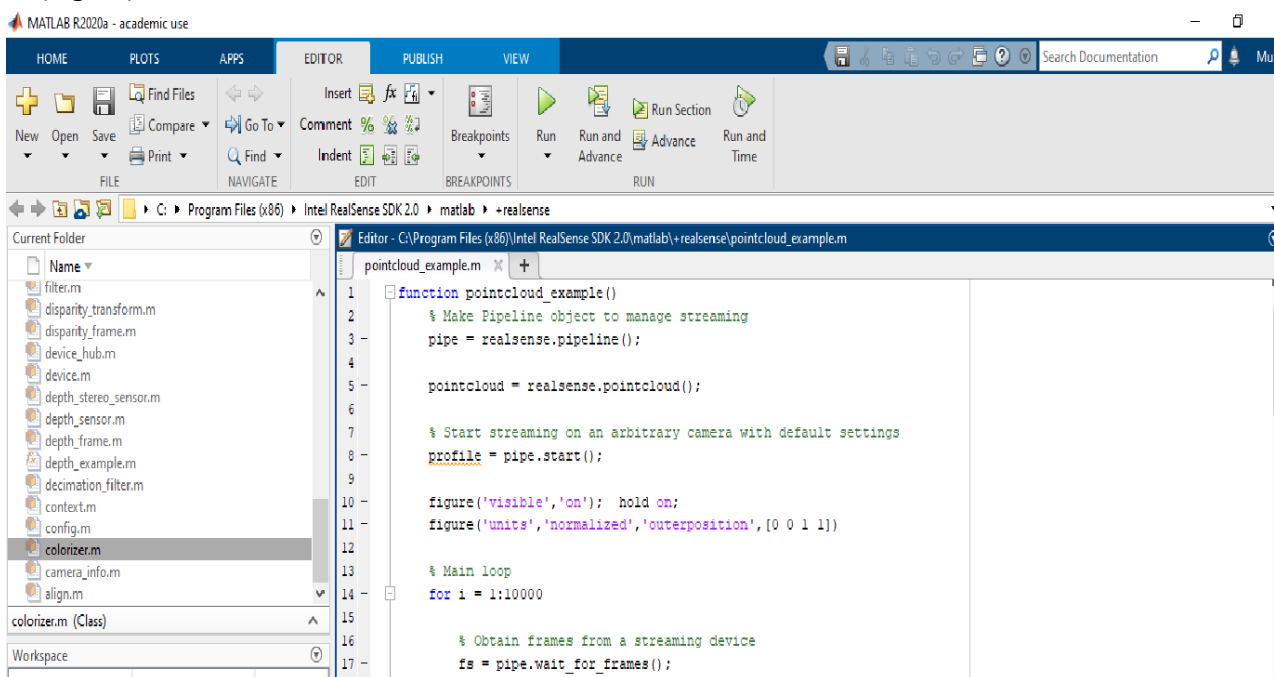

Fig3.1
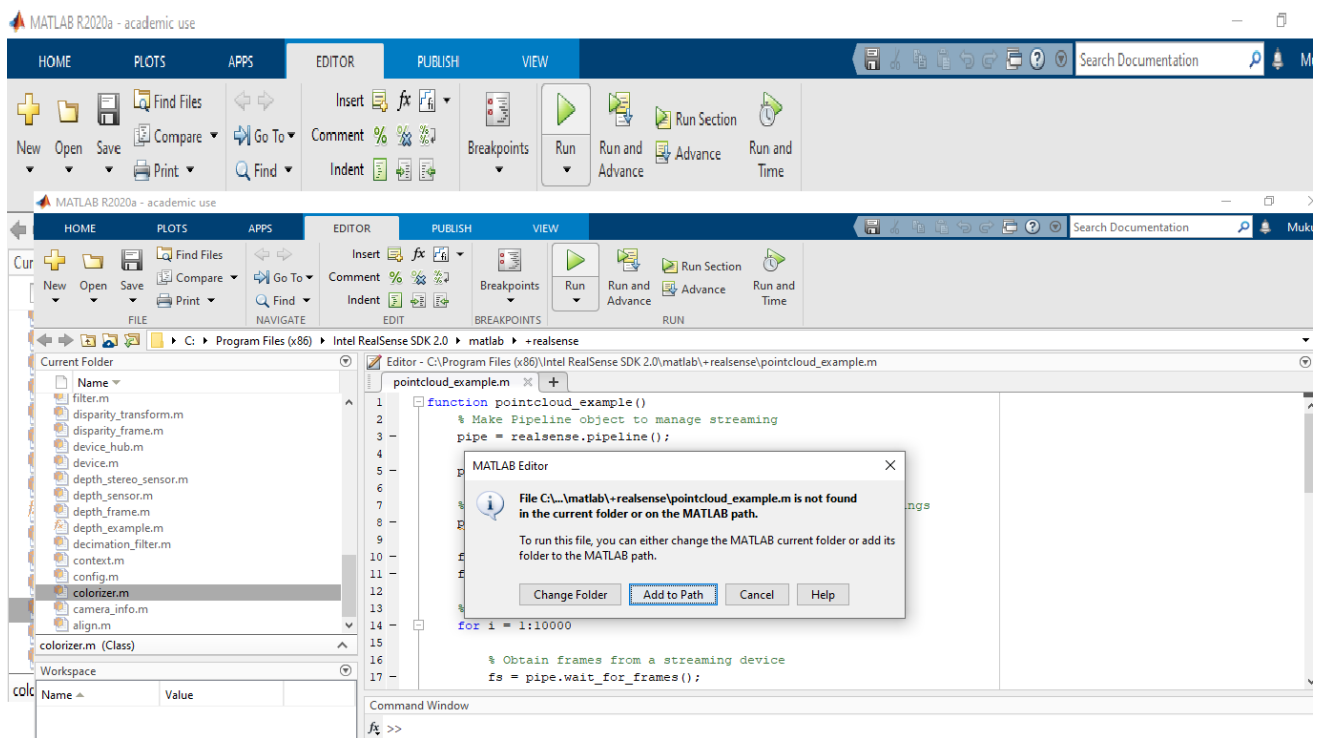
Fig3.2
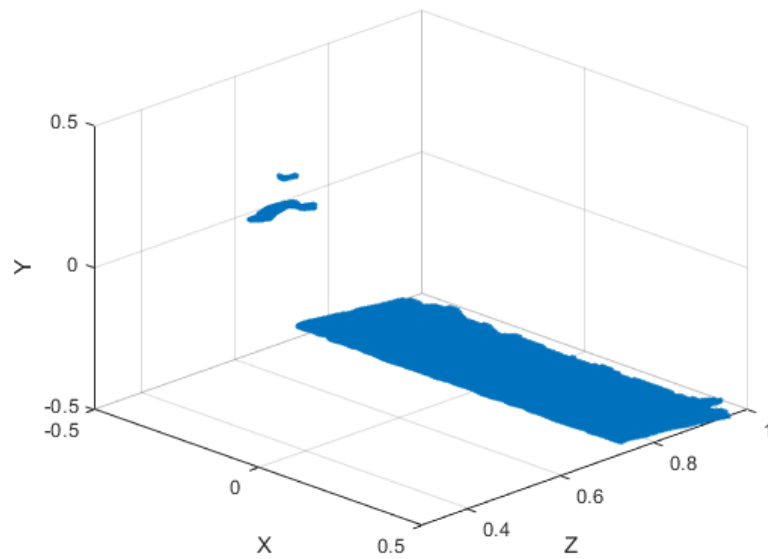
Fig3.3 Pointcloud example

```
points = pointcloud.calculate(depth);

% Adjust frame CS to matlab CS
vertices = points.get_vertices();
X = vertices(:,1,1);
Y = vertices(:,2,1);
Z = vertices(:,3,1);
depthData = [X, Y, Z];
writematrix(depthData, 'depthData.csv');
```

Fig3.4

```
img = permute(reshape(colordata',[3,color.get_width(),color.get_height()]),[3 2 1]);
r = img(:, :, 1);
g = img(:, :, 2);
b = img(:, :, 3);

writematrix(r,"R.csv");
writematrix(g,"G.csv");
writematrix(b,"B.csv");
```

Fig3.5
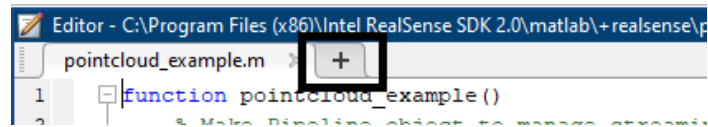
Fig3.6

3.6 Write the following code:

Fig3.7

Fig3.8
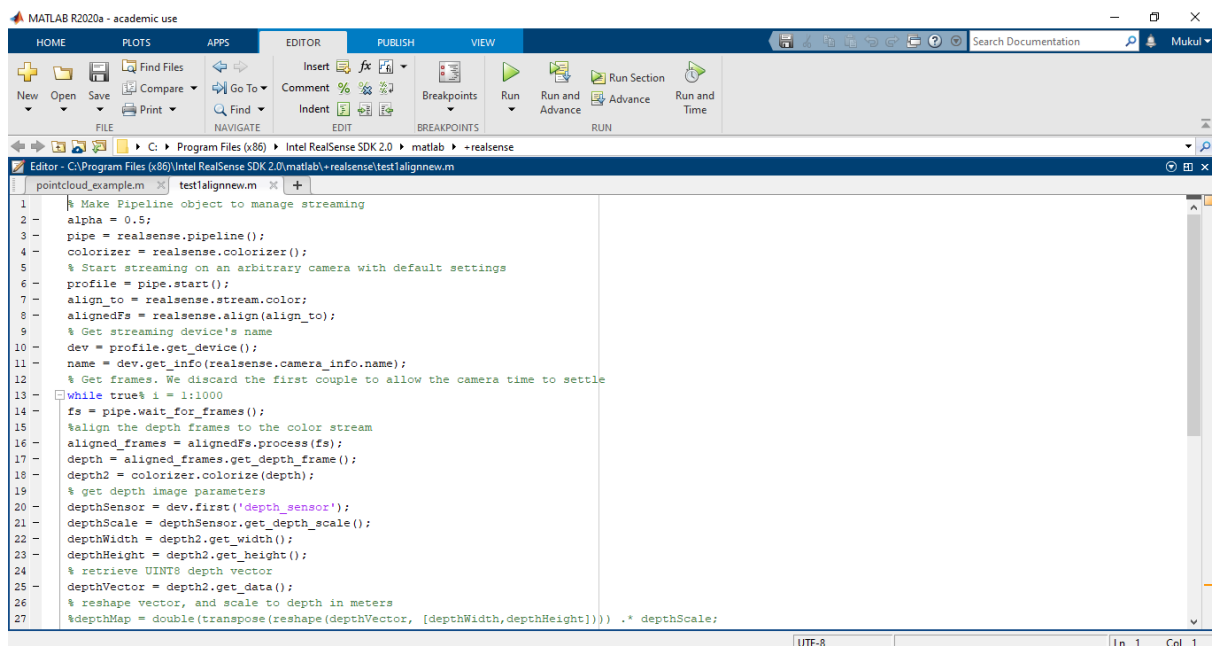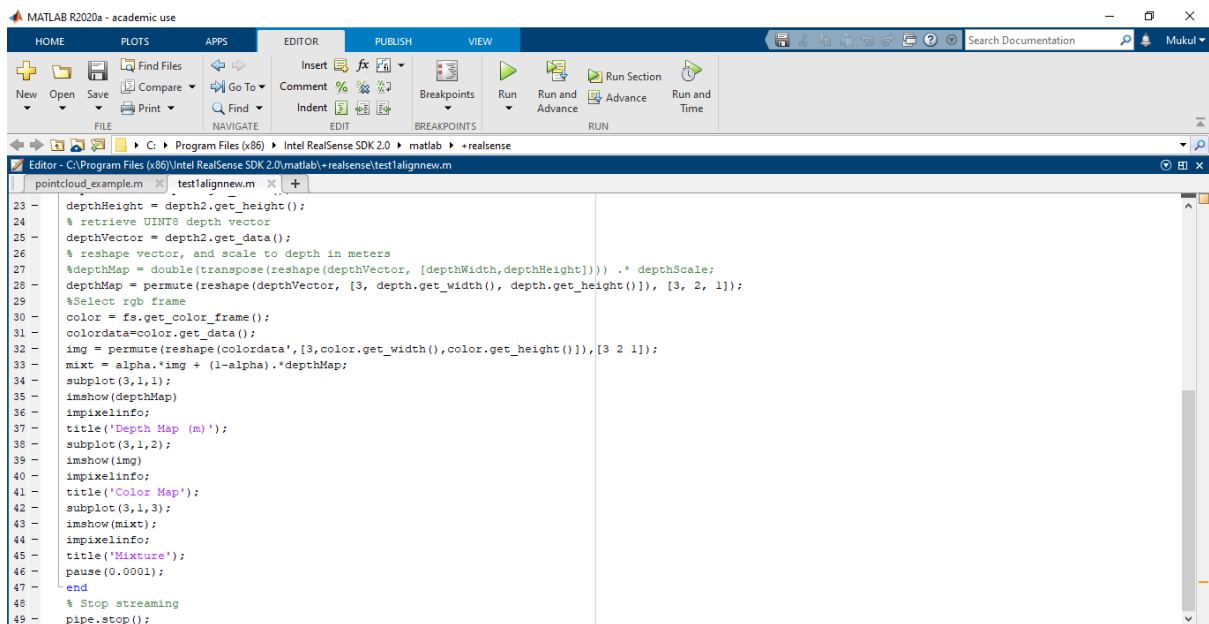
3.7 Now run this example and see the results (Fig3.9)

Fig3.9

3.8 We can also write the RGB values of the image into a file by again adding a small line to the code same as what we did in pointcloud example (Fig3.10).

```
img = permute(reshape(colordata',[3,color.get_width(),color.get_height()]),[3 2 1]);
writematrix(img,"RGBData.csv")
```

Fig3.10

Here, you just need to mention the image name and the filename where you want to write data into.

3.9 Sometimes you might get error like in fig3.11


```
Command Window
  Error using librealsense_mex
  Frame didn't arrive within 15000

  Error in realsense.pipeline/wait_for_frames (line 40)
              out = realsense.librealsense_mex('rs2::pipeline', 'wait_for_frames', this.objectHandle);

  Error in realsense.pointcloud_example (line 17)
          fs = pipe.wait_for_frames();
```
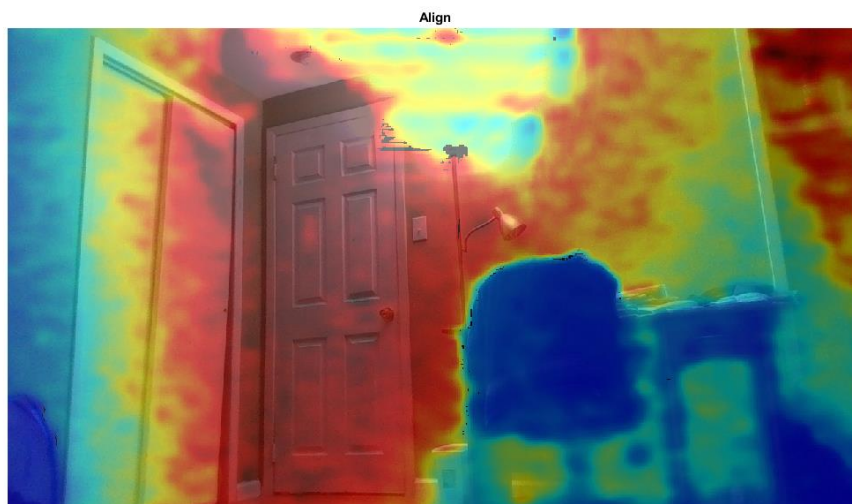
Fig3.11

But that does not hamper your result; it runs successfully. It will specifically come for Pointcloud example; however, it is a possibility that it is a bug from SDK and MATLAB functions which are given as samples from Intel.

In C++ code the value of α can be changed by sliding the bar while for MATLAB we have specified the value of α as 0.5.

4. APPENDIX

4.1 MATLAB code for point cloud example:

```matlab
function pointcloud_example()
    % Make Pipeline object to manage streaming
    pipe = realsense.pipeline();
    pointcloud = realsense.pointcloud();
    % Start streaming on an arbitrary camera with default settings
    profile = pipe.start();
    figure('visible','on');  hold on;
    figure('units','normalized','outerposition',[0 0 1 1])
    % Main loop
    for i = 1:5
        % Obtain frames from a streaming device
        fs = pipe.wait_for_frames();
        % Select depth frame
        depth = fs.get_depth_frame();
        %color = fs.get_color_frame();
        % Produce pointcloud
        if (depth.logical())% && color.logical())
            %pointcloud.map_to(color);
            points = pointcloud.calculate(depth);
            % Adjust frame CS to matlab CS
            vertices = points.get_vertices();
            X = vertices(:,1,1);
            Y = vertices(:,2,1);
            Z = vertices(:,3,1);
            %depthData = [X, Y, Z];
            % writematrix(depthData, 'depthData.csv');
            plot3(X,Z,-Y,'.');
            grid on
            hold off;
            view([45 30]);
            xlim([-0.5 0.5])
            ylim([0.3 1])
            zlim([-0.5 0.5])
            xlabel('X');
            ylabel('Z');
```

```matlab
            zlabel('Y');
            pause(0.01);
        end
       % pcshow(vertices); Toolbox required
    end
     % Stop streaming
    pipe.stop();

end
```

4.2 MATLAB code for Align:

```matlab
%Make Pipeline object to manage streaming
alpha = 0.5;
pointcloud = realsense.pointcloud();
pipe = realsense.pipeline();
colorizer = realsense.colorizer();
% Start streaming on an arbitrary camera with default settings
profile = pipe.start();
align_to = realsense.stream.color;
alignedFs = realsense.align(align_to);
% Get streaming device's name
dev = profile.get_device();
name = dev.get_info(realsense.camera_info.name);
% Get frames. We discard the first couple to allow the camera time to
settle
for i = 1:5
    fs = pipe.wait_for_frames();
    %align the depth frames to the color stream
    aligned_frames = alignedFs.process(fs);
    depth = aligned_frames.get_depth_frame();
    depth2 = colorizer.colorize(depth);
    %pointcloud.map_to(color);
    points = pointcloud.calculate(depth);
    %Adjust frame CS to matlab CS
    vertices = points.get_vertices();
    X = vertices(:,1);
    Y = vertices(:,2);
    Z = vertices(:,3);
    depthData = [X, Y, Z];
    % get depth image parameters
    depthSensor = dev.first('depth_sensor');
    depthScale = depthSensor.get_depth_scale();
    depthWidth = depth2.get_width();
    depthHeight = depth2.get_height();
    % retrieve UINT8 depth vector
    depthVector = depth2.get_data();
    % reshape vector, and scale to depth in meters
    depthMap = permute(reshape(depthVector, [3, depth.get_width(),
depth.get_height()]), [3, 2, 1]);
    %Select rgb frame
    color = fs.get_color_frame();
    colordata=color.get_data();
    img =
permute(reshape(colordata',[3,color.get_width(),color.get_height()]),[3 2
1]);
    r = img(:, :, 1);
    g = img(:, :, 2);
    b = img(:, :, 3);
```

```matlab
        mixt = alpha.*img + (1-alpha).*depthMap;
        imshow(mixt)
        title("Align");
        pause(0.01);
end
%Stop streaming
pipe.stop();
writematrix(X,"Xcoord.csv");
writematrix(Y,"Ycoord.csv");
writematrix(Z,"Zcoord.csv");
writematrix(r,"Red.csv");
writematrix(g,"Green.csv");
writematrix(b,"Blue.csv");
```

4.3 MATLAB code for UI slider:

```matlab
alpha = 0.5;
pipe = realsense.pipeline();
colorizer = realsense.colorizer();
profile = pipe.start();
align_to = realsense.stream.color;
alignedFs = realsense.align(align_to);
%Initialiazing UI slider
fig = uifigure;
im = uiimage(fig);
sld = uislider(fig,'Value',0.5, 'ValueChangedFcn', @(sld, event)
updatesld(sld, alpha));
sld.Limits = [0 1];
while(True)
    fs = pipe.wait_for_frames();
    aligned_frames = alignedFs.process(fs);
    depth = aligned_frames.get_depth_frame();
    depth2 = colorizer.colorize(depth);
    depthSensor = dev.first('depth_sensor');
    depthScale = depthSensor.get_depth_scale();
    depthWidth = depth2.get_width();
    depthHeight = depth2.get_height();
    depthVector = depth2.get_data();
    depthMap = permute(reshape(depthVector, [3, depth.get_width(),
depth.get_height()]), [3, 2, 1]);
    color = fs.get_color_frame();
    colordata=color.get_data();
    img =
permute(reshape(colordata',[3,color.get_width(),color.get_height()]),[3 2
1]);
    mixt = alpha.*img + (1-alpha).*depthMap;
    im.ImageSource = mixt;
    pause(0.01);
end
% Stop streaming
pipe.stop();
writematrix(r,"R.csv");
writematrix(g,"G.csv");
writematrix(b,"B.csv");
function updatesld(sld, alpha)
alpha = sld.Value;
%disp(alpha);
end
```