Ravi Krishna Tejaswi Duttaluru
2656812

EEC-693/793 - COMPUTER VISION DEPTH CAMERA

Spring 2017

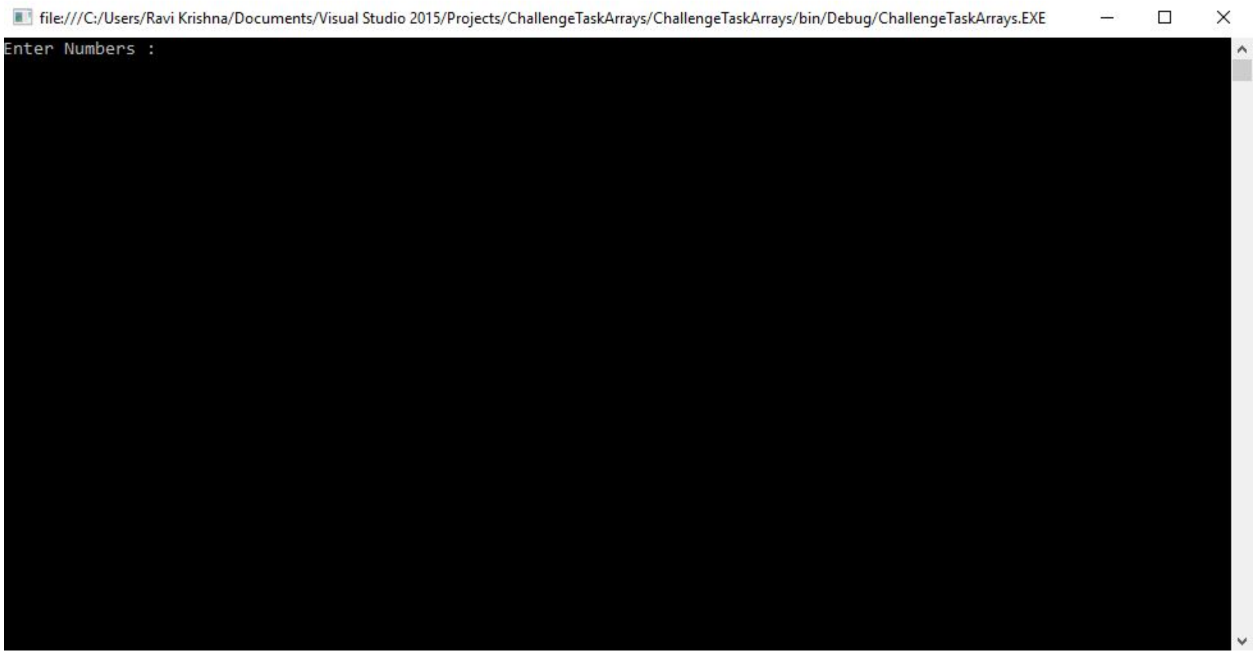
Challenge Task Report

Lecture 6:

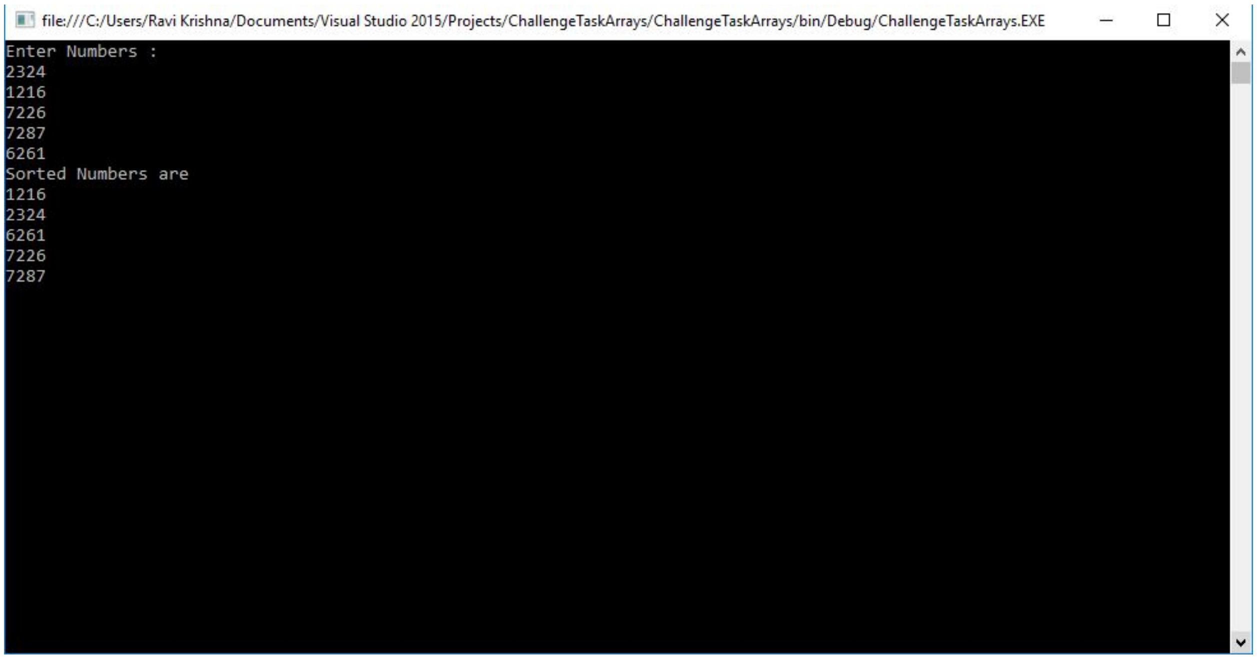
Write a program to sort number of elements in an array using a one-dimensional array. The input of elements should be taken during run time.

Source File: ChallengeTaskArrays

A C# program is written using Console Application in Visual Studio. A for loop has been implemented for 5 variables. The variables are sorted using a temporary variable. The output of the code is shown below.



```
file:///C:/Users/Ravi Krishna/Documents/Visual Studio 2015/Projects/ChallengeTaskArrays/ChallengeTaskArrays/bin/Debug/ChallengeTaskArrays.EXE
Enter Numbers :
```


file:///C:/Users/Ravi Krishna/Documents/Visual Studio 2015/Projects/ChallengeTaskArrays/ChallengeTaskArrays/bin/Debug/ChallengeTaskArrays.EXE

```
Enter Numbers :
2324
1216
7226
7287
6261
Sorted Numbers are
1216
2324
6261
7226
7287
```

<u>Lecture 7:</u>

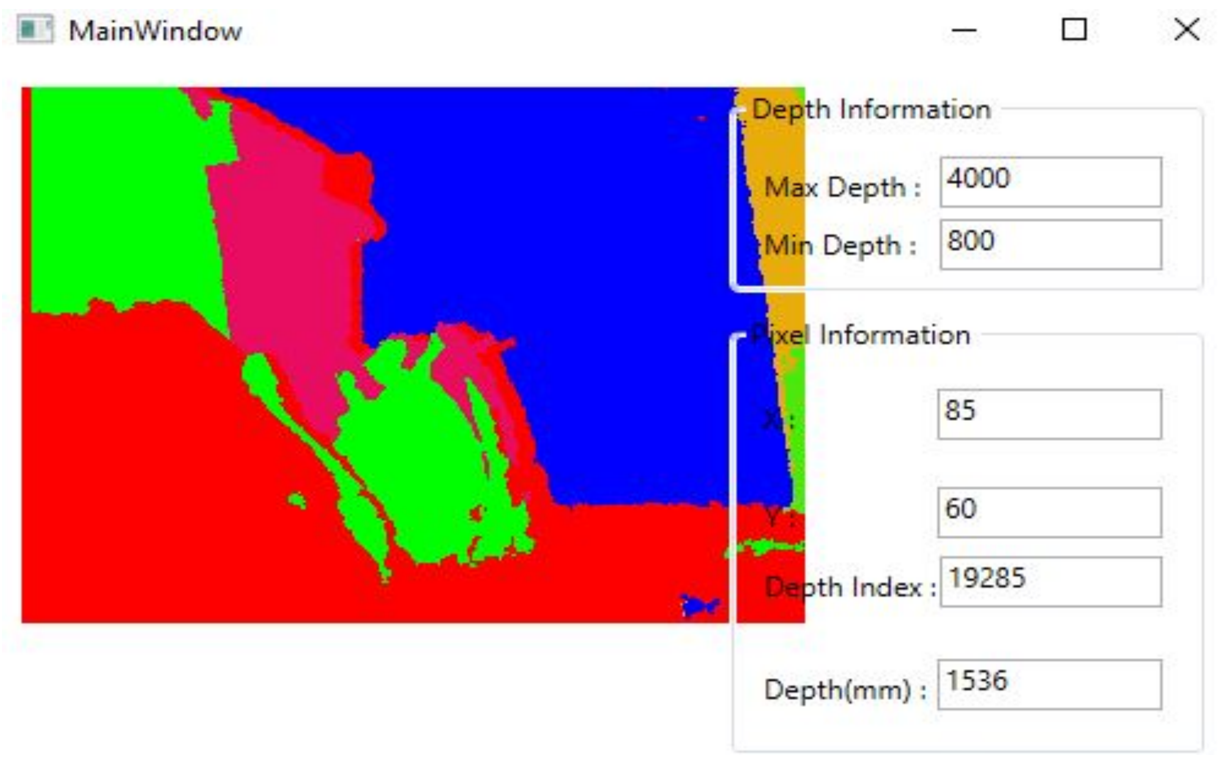
Modify the Depth Cam app such that the gray-scale depth image is changed to a color image, where the color indicates the depth band, for example, red denotes a distance from 3.5m to 4m, orange for 3.0m to 3.5m, etc.

Source File: ChallengeTaskDepthCam

The depth of the image is found out using a method named image1_Mousedown. In this method, the co-ordinates of the point where we click the mouse are recorded and the depth at that point is calculated. The depth at the given point is calculated using the depth frame pixels. In this code, the maximum depth declared is 4000mm and the minimum depth declared is 800mm. Moreover the co-ordinates where the mouse is pointed are also displayed in TextBoxes in the output window. The C# code given for designing the basic Depth Cam app is modified and a method named GeneratedColorBytes is added to convert the gray-scale depth image to a color image.

The colors indicated in the method are Red, Green, Pink, Yellow and Blue. If the depth is less than 1.0m(1000mm), the color indicated is **Red**. If the depth is more than 1.0m(1000mm) and less than 1.5m(1500mm), the color indicated is **Green**. If the depth is more than 1.5m(1500mm) and less than 2.0m(2000mm), the color indicated is **Pink**. If the depth is more than 2.0m(2000mm) and less than 2.5m(2500mm), the color indicated is **Light Green**. If the depth is more than

2.5m(2500mm) and less than 3.0m(3000mm), the color indicated is **Yellow**. If the depth is more than 3.0m(3000mm), the color indicated is **Blue**. The outputs of the code are shown below.



MainWindow

Depth Information

Max Depth : 4000

Min Depth : 800

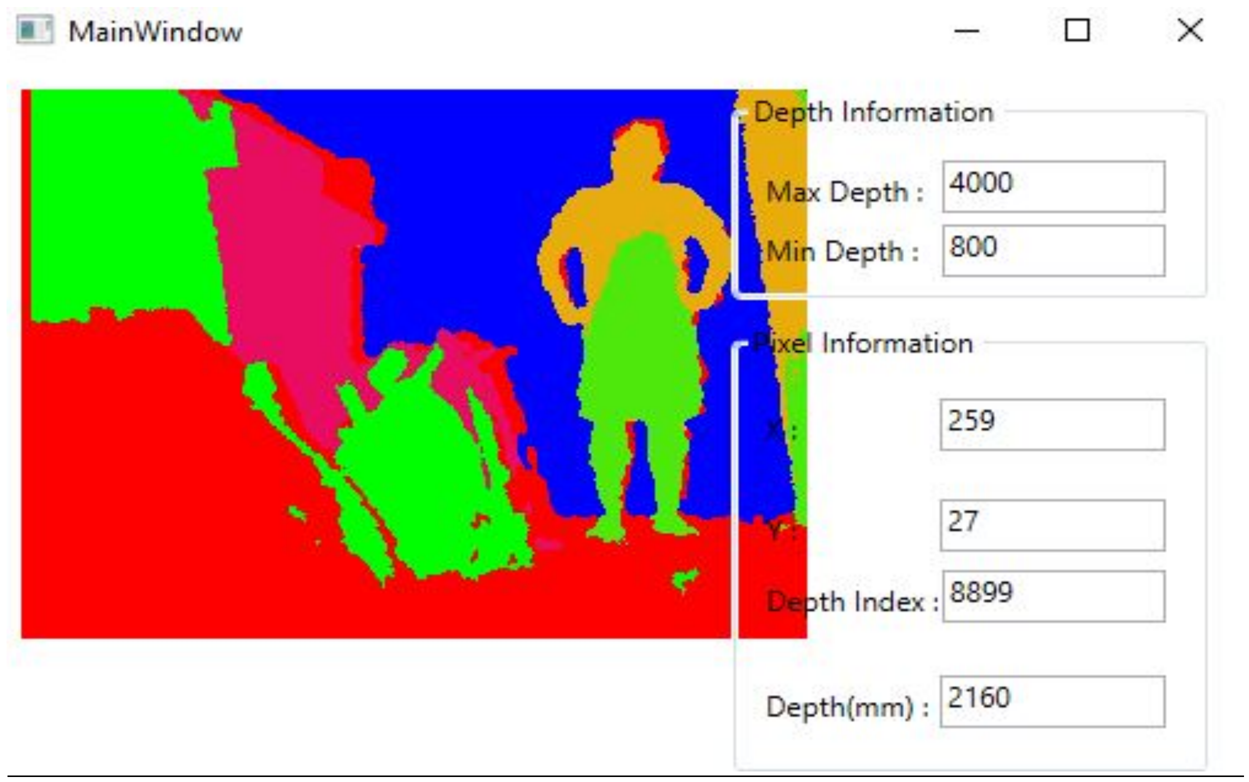
Pixel Information

X : 85

Y : 60

Depth Index : 19285

Depth(mm) : 1536



Depth Information

Max Depth : 4000

Min Depth : 800

Pixel Information

X : 259

Y : 27

Depth Index : 8899

Depth(mm) : 2160

<u>Lecture 8:</u>

Modify the Tracking Hand project to make it a drawing app

- **Shows all traces of the hand movement**
- **Add button to clear traces to make a new drawing**
- **Add a small palette chooser for change the color of the drawing point (an Ellipse)**
- **Note that you must add code such that the button/palette is pushed/selected using the gesture.**

Source File: ChallengeTaskPaintApp

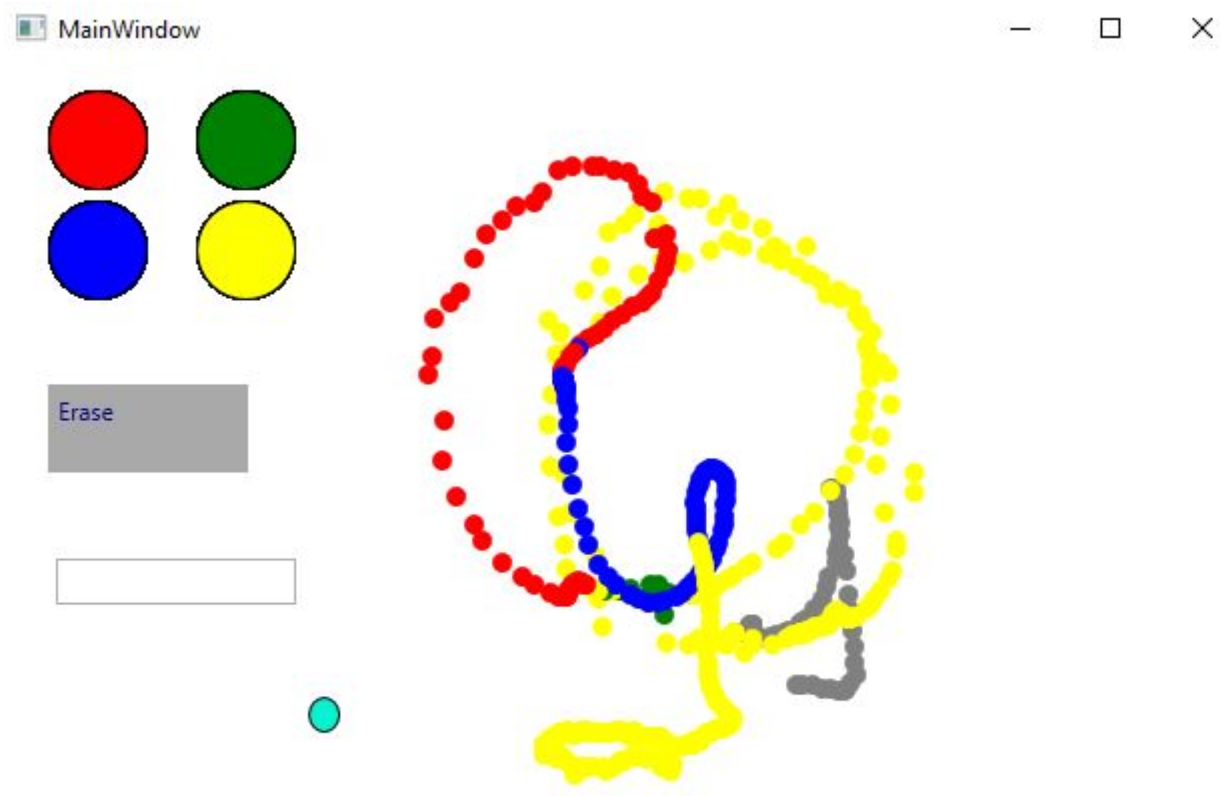
A C# code is written and the methods skeletonframeready, MapJointsWithUIElements, scaleposition are added. An integer is declared as ch=0which is used to represent our choice of colors. If ch=1, the color to be chosen is Red. If ch=2, the color to be chosen is Green and so on. Four ellipses are taken in the canvas and the colors are Red, Green, Blue and Yellow. A button is considered for clearing the canvas. The co-ordinates for these four ellipses and the clear button are recorded. When the code is being run, the left hand position will be moving. When the left hand position reaches the exact co-ordinates of a color or the erase button, the canvas either changes the color or it gets cleared when the co-ordinates of the erase button are matched. The command we use for clearing canvas is

Canvas.Children.Clear();

But, in my code, I renamed the canvas as paintapp. So my command is modified as

Paintapp.Children.Clear():

The output of the code is shown below.



Lecture 9:

Modify the Tracking Skeleton project so that you can measure the angle formed between left/right arm and torso.

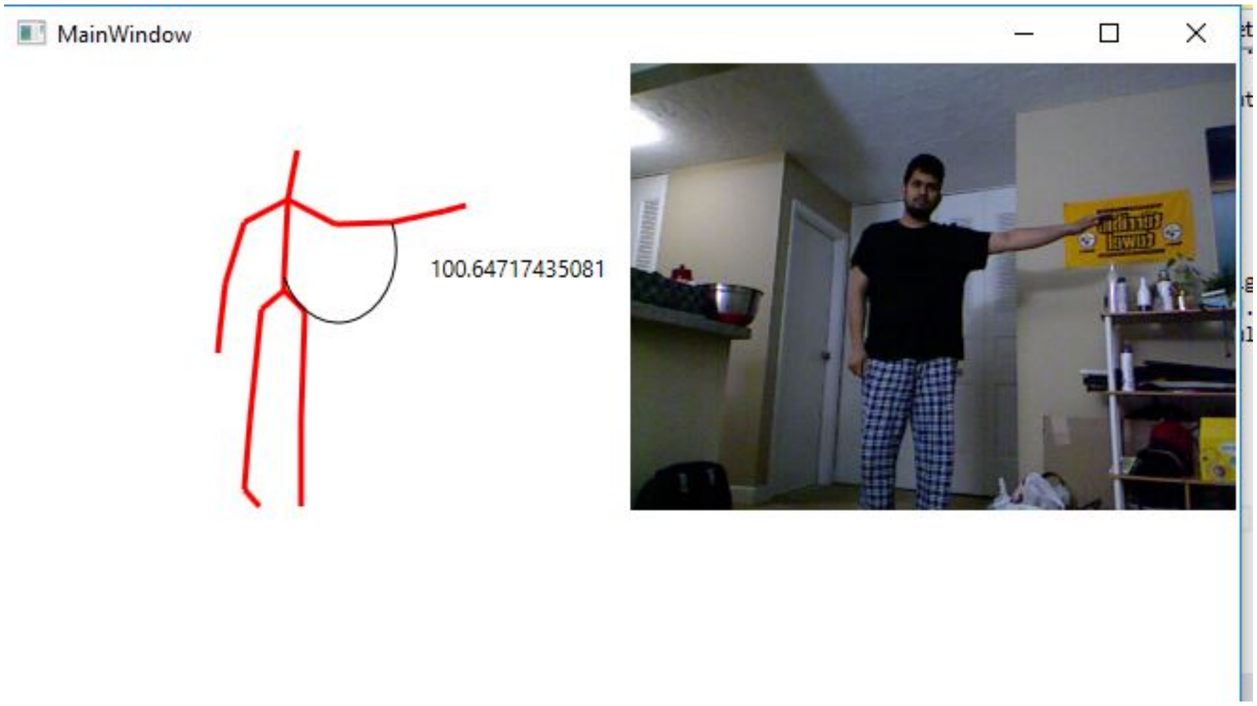
- **Draw an arc between the arm and torso.**
- **Display the angle value with a textbox on top of the arc.**

Source File: ChallengeTaskHandAngle

The angle between the spine and the elbow is found out in this code. For that 3 vectors are taken at right elbow, hip and right shoulder. Now the distances between (right shoulder and hip) and (right shoulder and right elbow) are calculated and noted as vectors b1 and b2. Now the angle is found out between b1 and b2. That gives the angle between right hand and torso. The angle found out with this formula is displayed in a textbox. As per the requirement, the textbox is supposed to be dynamic. In order to make it dynamic, these two lines are added to the code.

Canvas.SetLeft(textblock, handPt.X + 20);
Canvas.SetTop(textblock, handPt.Y + 20);

The output of the code is shown below.



100.64717435081

Lecture 12:

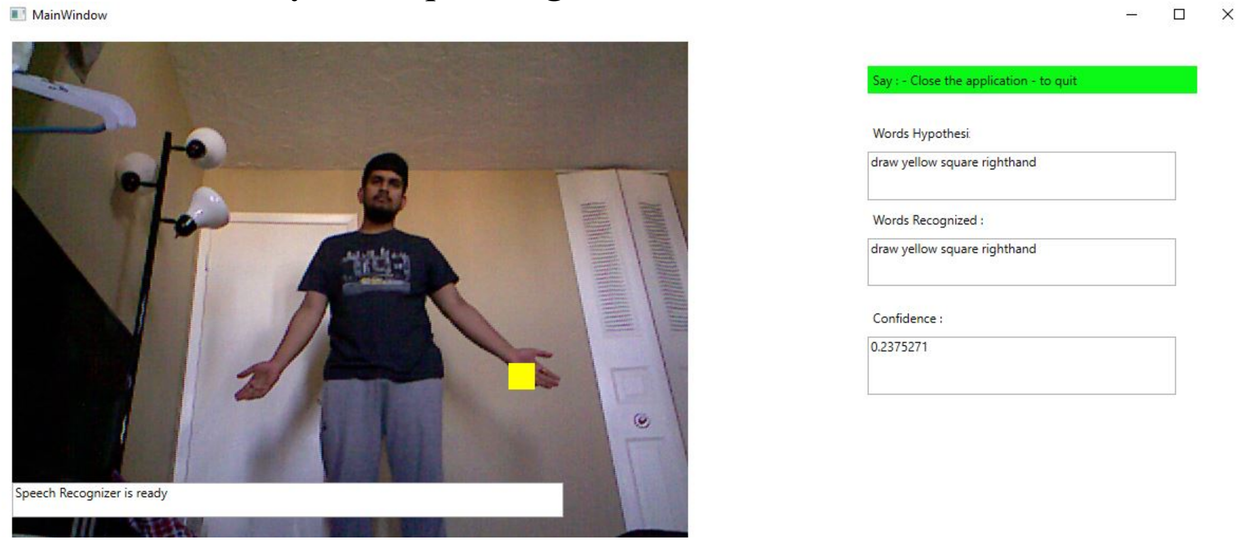
Improve the DrawShapeFromSpeech project in the following ways:

- **Enable both color image and skeleton data streams**
- **Display color image frames (but not the skeleton)**
- **Modify the grammar such that you can add a particular shape to a particular joint location**
  - **E.g., draw a red circle at the right hand**
- **Enable drawing by right (or left) hand, using the color and shape you specified in voice command**

Source File: ChallengeTaskDrawShapeFromSpeech

For this project, first the Speech references are added along with the Kinect references to enable the audio stream array. In the code we consider different shapes, colors and joints. The grammar of the code is explained below. First of all, I have split the command into four words. The first word is "Draw" which is constant. The second word is a color object. Here a color within the declared colors is said. The third word is a shape string. Here we say a desired shape within the declared shapes. The fourth word is a joint. I have declared three joints for my code. They are Head, Left Hand and Right Hand. Finally, it recognizes the four words and produces the outputs. For displaying the image without the skeleton, we have to super impose the image frame over the canvas frame with same resolution of 640X480 and same positioning of both frames. The outputs of the code are shown below.

Command: Draw yellow square righthand.



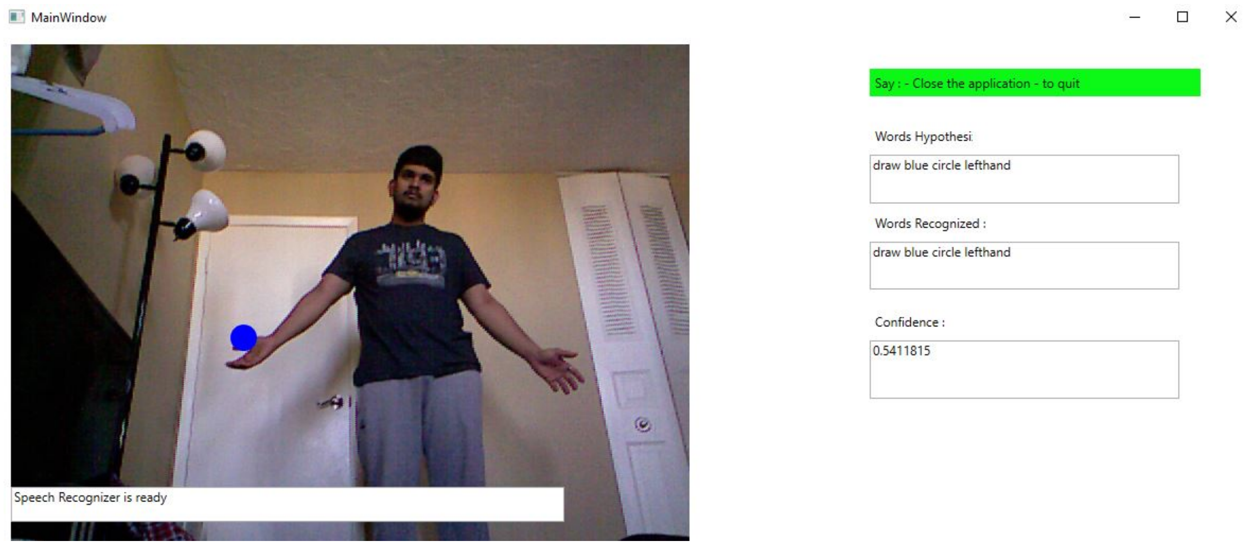
Speech Recognizer is ready

draw yellow square righthand

draw yellow square righthand

0.2375271

Command: Draw blue circle lefthand.


MainWindow

Speech Recognizer is ready

Say : - Close the application - to quit

Words Hypothesi:

draw blue circle lefthand

Words Recognized :

draw blue circle lefthand

Confidence :

0.5411815

<u>Lecture 13:</u>

Improve the basic Unity project (for object rotation and translation) in the following ways:

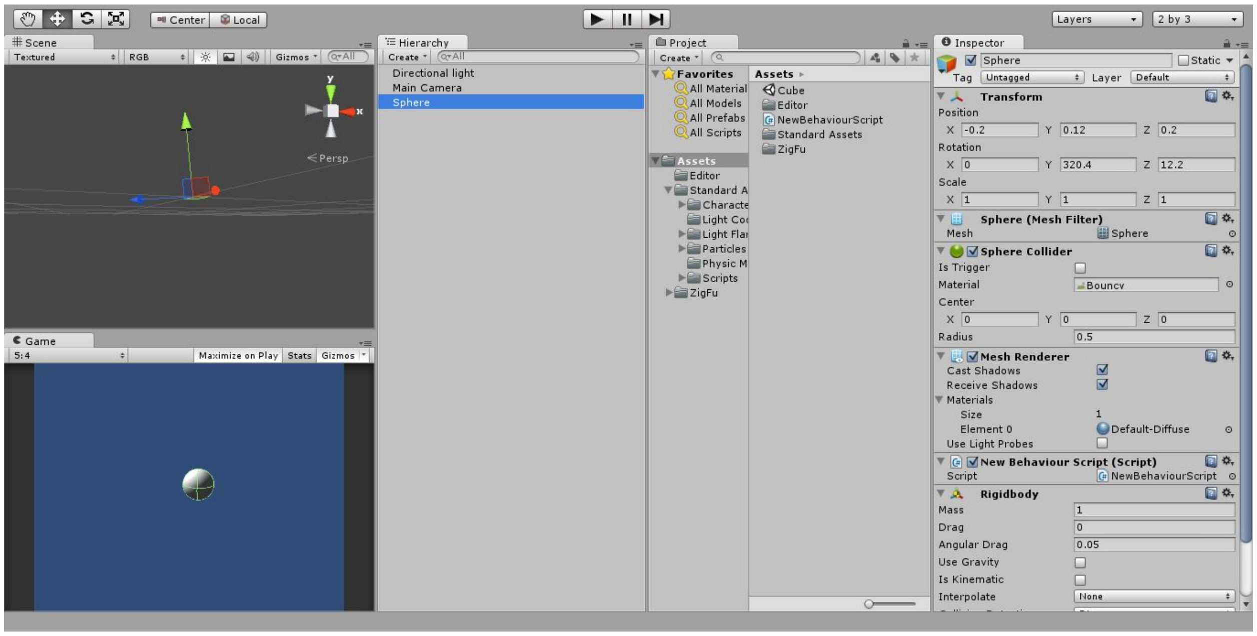
- **Add boundary check programmably for the game object when it reaches the boundary of the game view.**
- **Change direction of the movement for the game object programmably each time it reaches the boundary so that the game object bounces back and forth between the boundaries of the game view.**

Source File: ChallengeTaskCube

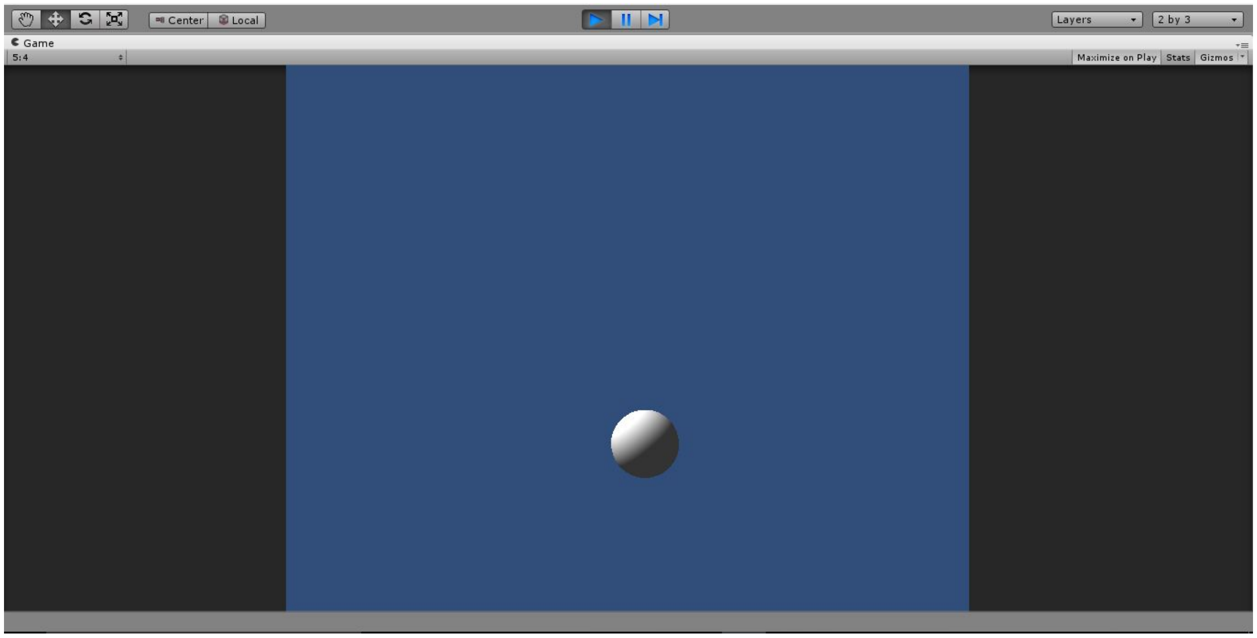
The view of the Main Camera is changed from Perspective view to Orthographic view and the size is set to 4. The view of the camera is adjusted such that the view acts as boundaries.

A sphere is added to the game view. A sphere collider and a rigid body is added to the inspector of the sphere. Make sure to add a Bouncy material in the sphere collider. Make sure to uncheck the 'Use Gravity' button in the rigid body. This is to be done because as per rules of gravity, a ball cannot bounce back without hitting a surface and the bouncing property is not same for all the materials. So, when we uncheck the gravity, the movement speed and bouncing of the ball completely depends on the code we added to the sphere.

A C# script is added to the sphere. In the script, a Boolean variable is declared which is used to verify the conditions for rotation, translation and position. In the code, the positions of x, y and z coordinates are modified with respect to time. The position of the sphere is give in both positive and negative axes. The outputs of the code are shown below.



```
Sphere                      Static
Tag  Untagged      Layer  Default
Transform
Position
  X -0.2      Y 0.12      Z 0.2
Rotation
  X 0         Y 320.4     Z 12.2
Scale
  X 1         Y 1         Z 1
Sphere (Mesh Filter)
  Mesh                   Sphere
Sphere Collider
Is Trigger               ☐
Material                 Bouncv
Center
  X 0         Y 0         Z 0
Radius                   0.5
Mesh Renderer
  Cast Shadows           ✓
  Receive Shadows        ✓
  Materials
    Size                 1
    Element 0            Default-Diffuse
  Use Light Probes       ☐
New Behaviour Script (Script)
  Script                 NewBehaviourScript
Rigidbody
Mass                     1
Drag                     0
Angular Drag             0.05
Use Gravity              ☐
Is Kinematic             ☐
Interpolate              None
```



<u>Lecture 15:</u>

Improve the Gesture Recognition Project in the following ways:

- **Add recognition of two more gestures: (1) Right hand is raised; (2) Left hand is raised**
- **Add the GestureRecognitionEngine.cs to a Unity + Kinect app, and add visual feedback on the gestures recognized.**

Source File: ChallengeTaskGestureRecognition

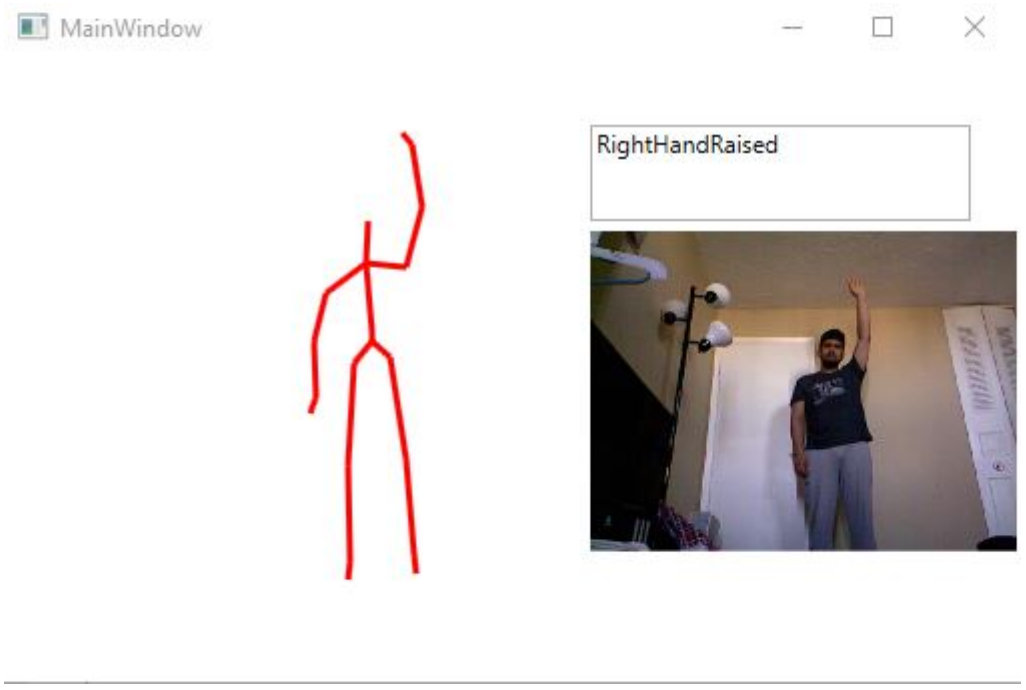
In this project, the given C# code is modified. In this code, a skeleton is displayed and its movements are tracked by the Kinect sensor. Whenever a joint is moved, it detects the movement of the joints and displays it in the Output window. So, when the right elbow and right wrist are raised up, it shows “Right Hand Raised”. If the left wrist and left elbow are raised, it displays “Left Hand Raised”. If the left wrist and right wrist are joined together, it detects that you are clapping your hands by joining your two wrists and the message is displayed as “Hands Clapping”. If all the right elbow, right wrist, left elbow, left wrist are raised, it detects the movement of all four joints and displays “Two Hands Raised” in the TextBox.

The basic code given in the lecture has two gestures pre-defined. They are “Two Hands Raised” and “Hands Clapping”. Two more gestures are added to the given code. They are “Right Hand Raise” and “Left Hand Raise”. The code for adding right hand raise and left hand raise gestures is shown below.

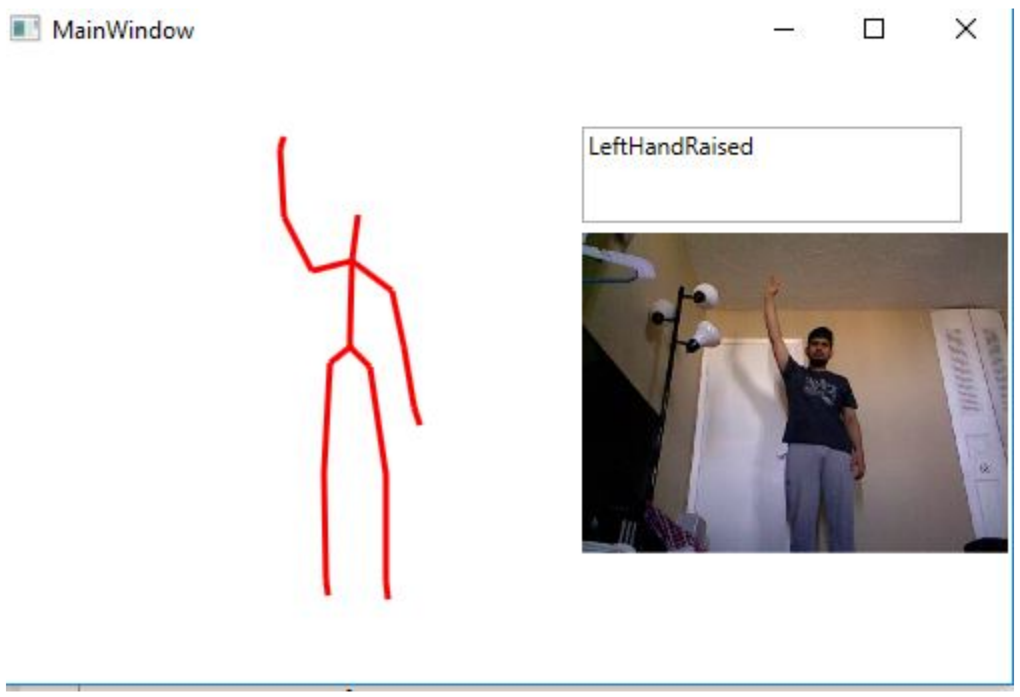
```
case GestureType.RightHandRaised:
        this.MatchRightHandRaisedGesture(this.Skeleton);
        break;

case GestureType.LeftHandRaised:
        this.MatchLeftHandRaisedGesture(this.Skeleton);
        break;
```

The outputs of the code are shown below.



RightHandRaised



LeftHandRaised