Challenge Task 6

Reza Shisheie

2708062

The goal of this task is to successfully add zoom in and zoom out to the app.

Two classes of zoom\_in and zoom\_out are added to the project and their Gesture types are added to “GestureRecognitionEgine.cs”

Zoom in starts once two hands are less than 0.5m distance and end if distance is over 1 m

Zoom out starts when distance is more than 0.75 m and ends when it is less than 0.25 m

Here is a snippet of code for zoom out:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using Microsoft.Kinect;

namespace lecture\_11\_rule\_based

{

public class zoom\_out : GestureBase

{

private double validatePosition;

private double startingPostion;

public zoom\_out() : base(GestureType.zoom\_out) { }

protected override bool ValidateGestureStartCondition(Skeleton skeleton)

{

//Console.WriteLine("ValidateGestureStartCondition");

var handRightPosition = skeleton.Joints[JointType.HandRight].Position;

var handLeftPosition = skeleton.Joints[JointType.HandLeft].Position;

var shoulderRightPosition = skeleton.Joints[JointType.ShoulderRight].Position;

var spinePosition = skeleton.Joints[JointType.Spine].Position;

var hipPosition = skeleton.Joints[JointType.HipCenter].Position;

if ((handRightPosition.Y < shoulderRightPosition.Y) && (handLeftPosition.Y < shoulderRightPosition.Y) &&

(handRightPosition.Y > hipPosition.Y) &&

(handLeftPosition.Y > hipPosition.Y) &&

(Math.Abs(handLeftPosition.X - handRightPosition.X) > 0.75))

{

//Console.WriteLine("ValidateGestureStartCondition --> yes");

validatePosition = Math.Abs(handLeftPosition.X - handRightPosition.X);

startingPostion = Math.Abs(handLeftPosition.X - handRightPosition.X);

return true;

}

return false;

}

protected override bool IsGestureValid(Skeleton skeletonData)

{

//Console.WriteLine("IsGestureValid");

var currentHandRightPoisition = skeletonData.Joints[JointType.HandRight].Position;

var currentHandLeftPoisition = skeletonData.Joints[JointType.HandLeft].Position;

if (validatePosition < Math.Abs(currentHandRightPoisition.X - currentHandLeftPoisition.X))

{

return false;

}

//Console.WriteLine("IsGestureValid-->yes");

validatePosition = Math.Abs(currentHandRightPoisition.X - currentHandLeftPoisition.X);

return true;

}

protected override bool ValidateGestureEndCondition(Skeleton skeleton)

{

var currentHandRightPoisition = skeleton.Joints[JointType.HandRight].Position;

var currentHandLeftPoisition = skeleton.Joints[JointType.HandLeft].Position;

double distance = Math.Abs(currentHandLeftPoisition.X - currentHandRightPoisition.X);

float currentshoulderDiff = GestureHelper.GetJointDistance(skeleton.Joints[JointType.HandRight], skeleton.Joints[JointType.HandLeft]);

//Console.WriteLine("ValidateGestureEndCondition");

//Console.WriteLine(currentshoulderDiff);

if (currentshoulderDiff < 0.25)

return true;

return false;

}

protected override bool ValidateBaseCondition(Skeleton skeleton)

{

var handRightPosition = skeleton.Joints[JointType.HandRight].Position;

var handLeftPosition = skeleton.Joints[JointType.HandLeft].Position;

var shoulderRightPosition = skeleton.Joints[JointType.ShoulderRight].Position;

var spinePosition = skeleton.Joints[JointType.Spine].Position;

var hipPosition = skeleton.Joints[JointType.HipCenter].Position;

if ((handRightPosition.Y < shoulderRightPosition.Y) &&

(handLeftPosition.Y < shoulderRightPosition.Y) &&

(handRightPosition.Y > hipPosition.Y) &&

(handLeftPosition.Y > hipPosition.Y))

{

//shoulderDiff = GestureHelper.GetJointDistance(skeleton.Joints[JointType.HandRight], skeleton.Joints[JointType.ShoulderLeft]);

validatePosition = Math.Abs(handLeftPosition.X - handRightPosition.X);

startingPostion = Math.Abs(handLeftPosition.X - handRightPosition.X);

return true;

}

return false;

}

}

}

Once zoom in or out is detected, the size of the shown photo is changes by 2X or 0.5 X

if (e.gsType.ToString() == "zoom\_in")

{

Console.WriteLine("zoom\_in");

BitmapImage myBitmapImage = new BitmapImage();

myBitmapImage.BeginInit();

myBitmapImage.UriSource = new Uri(@"D:\lecture\_2\_button.png");

myBitmapImage.DecodePixelWidth = 1000;

myBitmapImage.EndInit();

image2.Height = 300;

image2.Width = 300;

image2.Source = myBitmapImage;

}

if (e.gsType.ToString() == "zoom\_out")

{

Console.WriteLine("zoom\_out");

BitmapImage myBitmapImage = new BitmapImage();

myBitmapImage.BeginInit();

myBitmapImage.UriSource = new Uri(@"D:\lecture\_2\_button.png");

myBitmapImage.DecodePixelWidth = 1000;

myBitmapImage.EndInit();

image2.Height = 50;

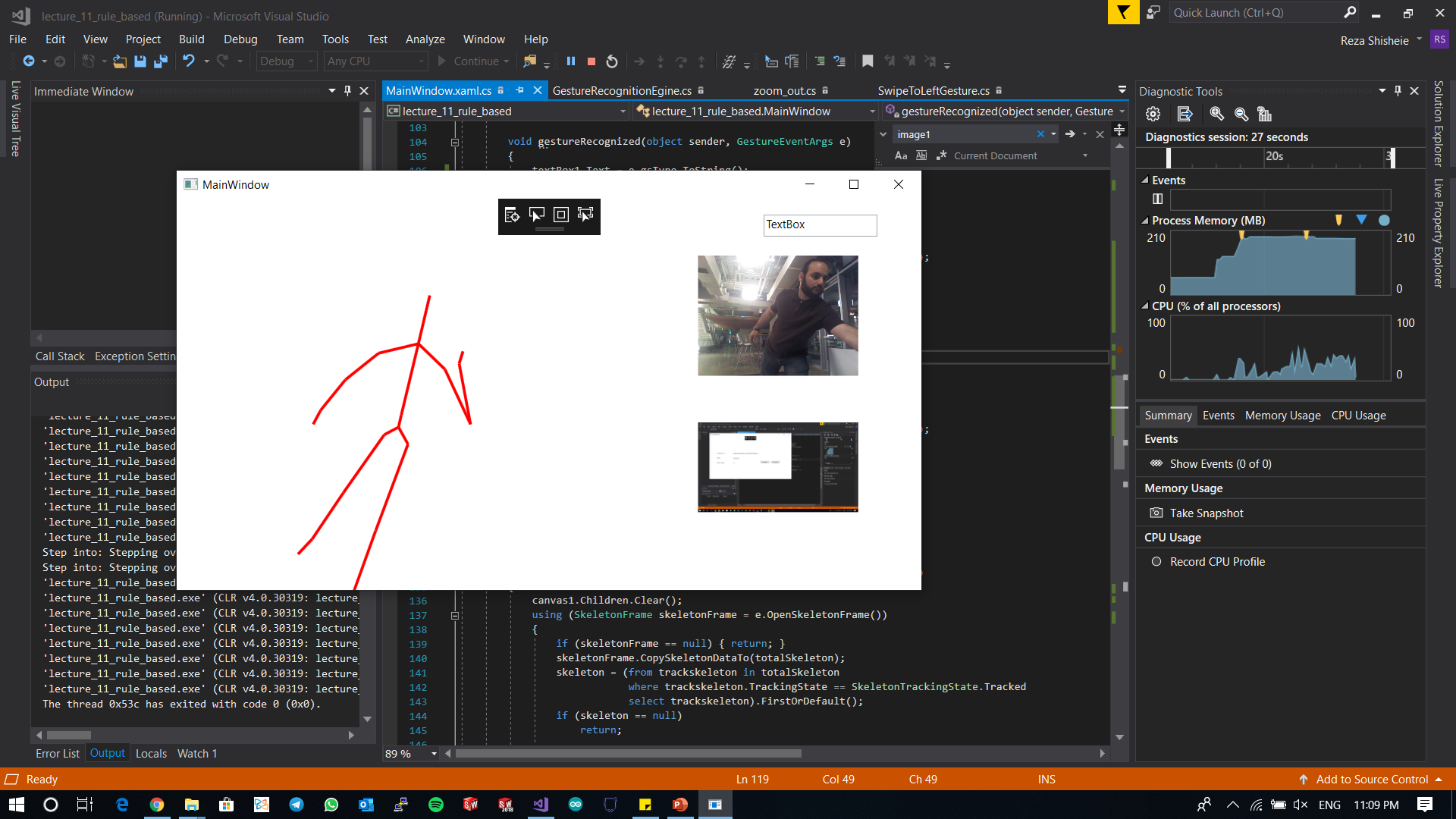
image2.Width = 50;

image2.Source = myBitmapImage;

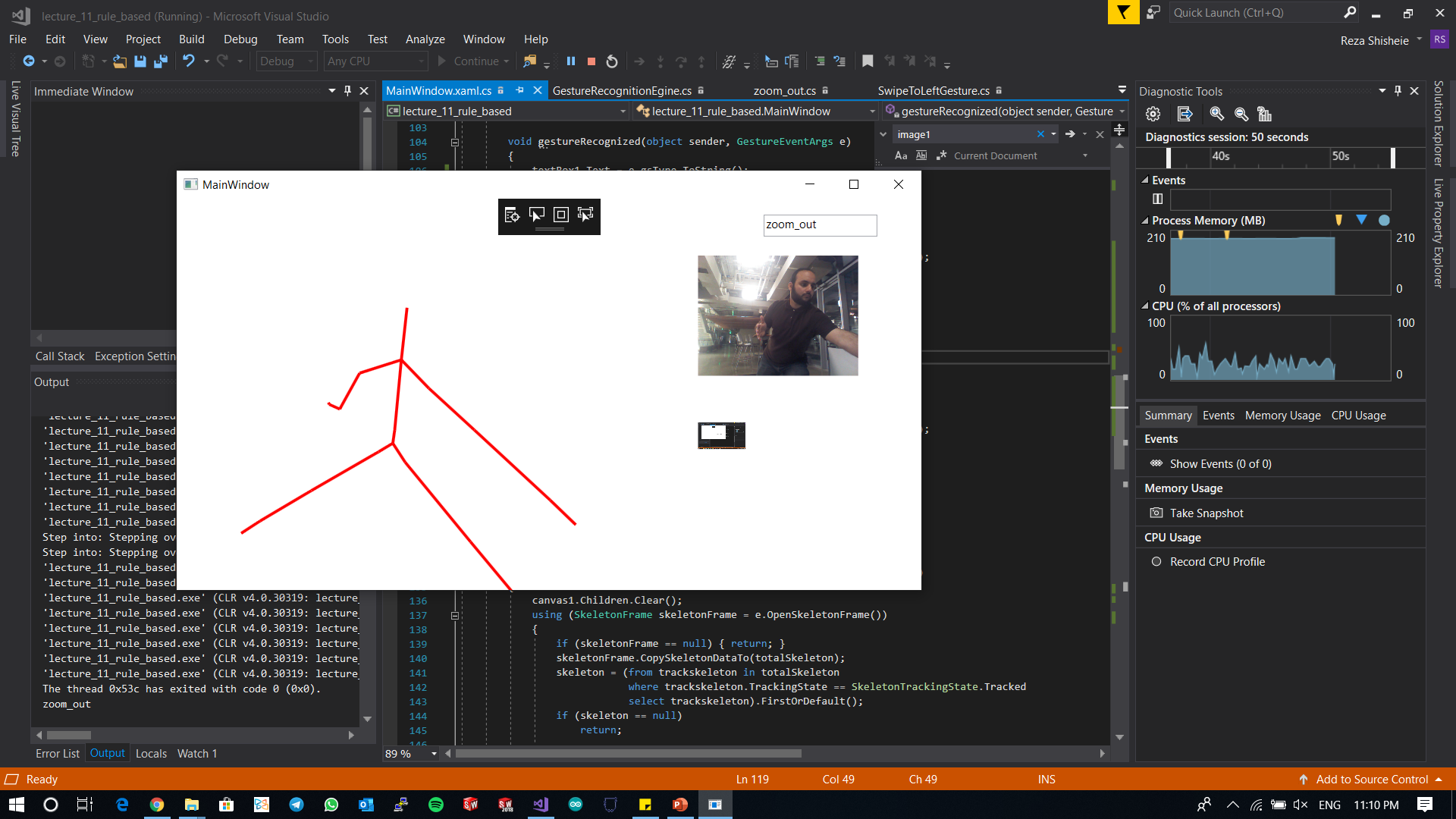
}

Here are screenshots of app:

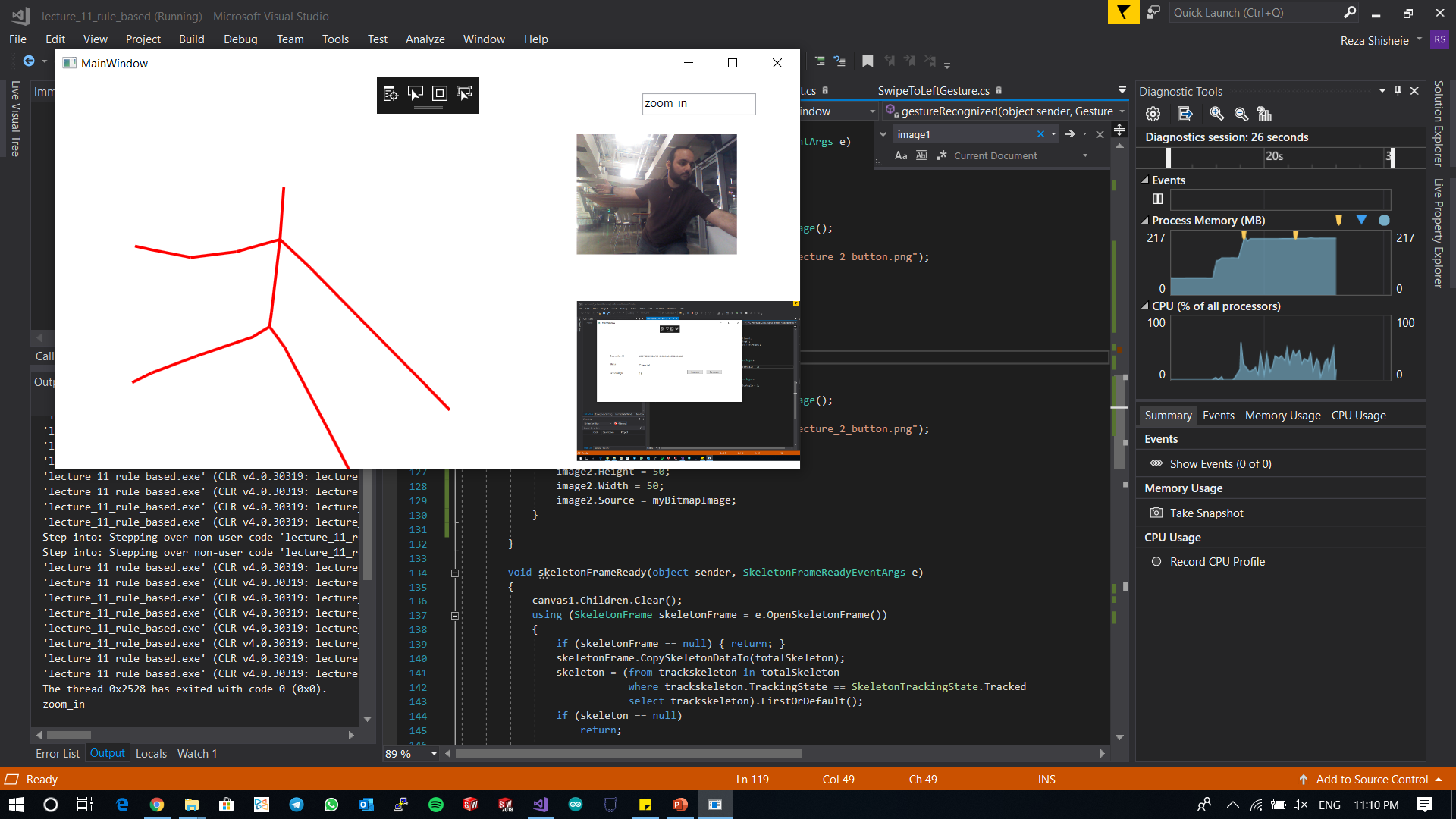
This is when no zoom in or out is detected:



After zoom in:



After zoom out:



Here are two screen shots of the app: