Indian Journal of Science and Technology, *Vol 8(7), 678–682, April 2015*

ISSN (Print) : 0974-6846
ISSN (Online) : 0974-5645
DOI: 10.17485/ijst/2015/v8i7/69907

An Augmented Reality Application with Leap and Android

V. Meenakshi Sundaram, Shriram K. Vasudevan*, C. Santhosh, R. G. K. Barath Kumar and G. Deepak Kumar

Amrita School of Engineering, Amrita Vishwa Vidyapeetham University, Coimbatore, India;
kv_shriram@cb.amrita.edu, msundarv@gmail.com, sansanthosh651@gmail.com, barath.rgk10@gmail.com, gdeepakkumar@live.com

Abstract

The objective of the work is to use hand gesture interface for augmented reality applications. By introducing hand gestures control through leap motion sensor we improve user experience in AR applications. It is first of its kind in augmented reality to interface Leap with Augmented Reality.

Keywords: Augmented Reality, Hardware, Operations, Software, Virtual Reality

1. Introduction

We aim at providing a system, apparatus, and methods are provided for Aaugmented Reality (AR) glass that accomplishes virtual computing with gesture interface. It helps the users to interact with real world environment using digital information. This means that the view of personal computing would be revolutionized into virtual computing, which is an innovation in the area of computing systems. The applications are built over the android platform. The image tracking algorithm using AR SDK helps to bring the digital content into the real world. The product also comes with a camera for surface tracking. The output is displayed with Field of View for each eye so that user experience stereoscopic view of digital content in the real world. The user interfaces are made with Leap motion technology so that the natural hand gestures are processed into specific commands for interacting with the applications[1].

2. Hardware Components

The glass consists of pair of Transparent OLED with see-through display. It consists of binary layers on both edges of the OLED, which accommodates an emissive and conductive layer. Electrical impulses travel through the conductive layer and produce light at the emissive layer. The narrowness of the screen as well as the clear cathodes within allows the screens to be transparent. Two cameras are placed on either side of Glass for surface tracking. Leap motion sensor is placed on the glass which could track hands at lest of about 1 cm accuracy. The glass constructed will look something like as shown in Figure 1.

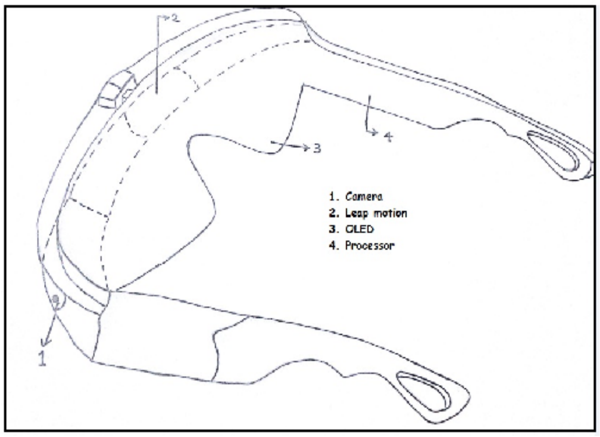

Figure 1. Hardware components.

*Author for correspondence

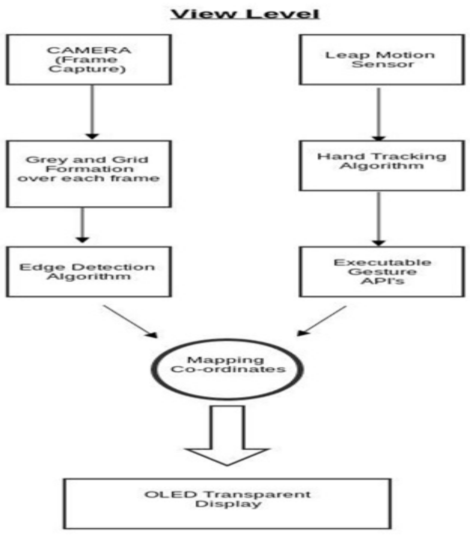

Figure 2. View level.

2.1 Camera

With the help of depth sense view of camera, the captured frames are processed into grey scale and the edges in the frames are recognized. Finally the required information is embedded in user's field of view.

2.2 Leap Motion Sensor

This sensor provides effective control over the applications. The APIs provided by the sensors can be modified with our Android Kernel Driver so that the user could experience effective Interface. Dotted lines indicate the position of the sensor. (www.leapmotion.com can help the reader and researchers to gather more information on the same).

2.3 Transparent OLED

Transparent displays with OLED's are the most commonly produced forms of see-through displays on the market. It provides stereoscopic view of output in front of user eyes.

2.4 Snapdragon Processor

Snapdragon is one of the family members of mobile Systems on a Chip (SoC) by the organization, QUALCOMM. It has multiple topographies like to those of the ARM Cortex-A8 core and it is based on the ARM v7 instruction set, it has much higher performance for multimedia-related SIMD operations.

3. Software Components

Applications are processed over the android platform. The upper level of the software is completely based on the camera and leap motion sensor. The surface is tracked and each region gets separated into horizontal and vertical scan lines of 5 pixels apart. The initial step is to convert the frame into Grey scale so that the dark and white pixel alternatively represents one and zeros. Then the Gaussian derivative is used to detect black/white edges in the image. The Gaussian derivative is based on the convolving the image with a small, separable and integer values filter in horizontal and vertical direction. After the detection of the edges, RANSAC algorithm is used to construct line segments in each region so that the co-ordinate of each pixel could be mapped with any digital content. This provides the digital information to come live in the real world with the same co-ordinate axis.

In the given figure (Figure 2), the camera and leap motion sensor acts as the foremost view level of the invention. The frame captured by the camera is processed into grey scale. The black and white pixel is tracked as grid. Each grid represents the line and the edges are detected. The Leap motion sensor tracks the hand and correspondingly the gesture is detected. The resultant final co-ordinate is mapped with the edges of the grid and the digital information is embedded over the grid layer and the final result is displayed on an OLED.

Leap motion sensor provides various API's for Hand Tracking, each of which are pure java libraries. The mapping co-ordinates of digital content could be made intractable with these API's. The Driver for Leap motion sensor is also available on Android platform, So that the developers can make the application in easier way.

4. Working Application

The applications are developed with the help of Unity 3D framework and final applications are built over the android platform. Using the AR SDK's the virtual objects are placed over the surface with viewpoints and the rendered objects are controlled using leap motion sensor. In the Figure 3, the table is tracked with the help of a webcam and the virtual object (hockey game) is placed and it is controlled using the leap motion sensor. The gesture used here are Pinch and Grab.

1) The Pinch gesture is used to trigger the virtual object on the table.

2) The grab gesture is used to get the control of the striker. Finally, the movement of the striker is controlled with the hands. JozeGuna et al2 Figure 4 has got the way the Android is made fit for Leap. Figure 8 has got the screenshot of the application developed for air hockey.

The bird's eye view of the proposed equipment is presented in Figure 7.

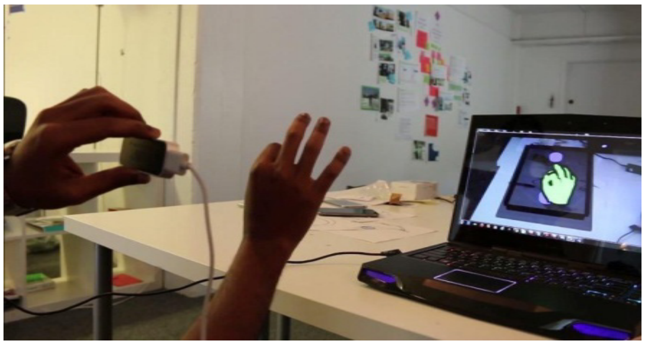

Figure 3. Glow Hockey game using laptop.

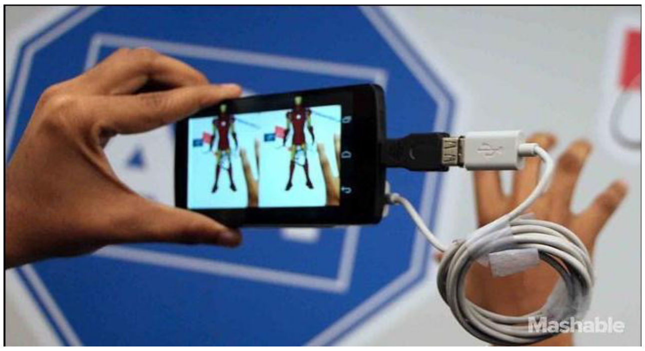

Figure 4. Leap motion with android.

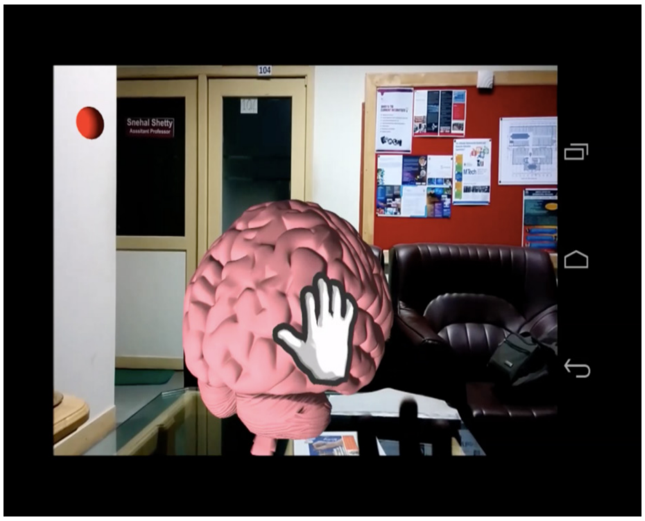

Figure 5. Medical application [Nexus 5 screenshot].

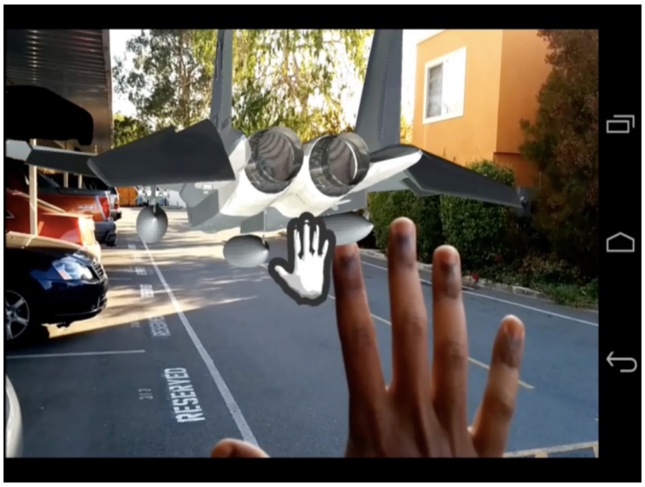

Figure 6. Outdoor application [Nexus 5 screenshot].

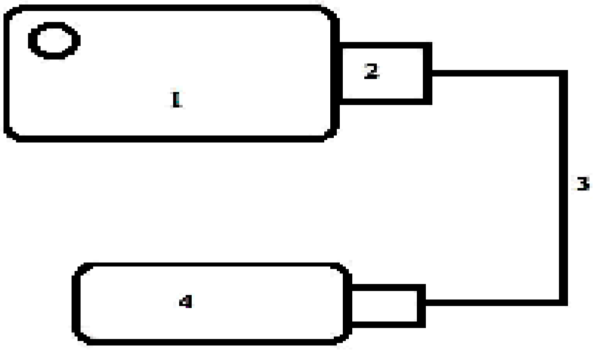

Figure 7. Birds view of the product design.

5. Mode of Operations

The best suit of operation is by connecting the leap motion sensor to the android mobiles (nexus 5) using the USB to micro USB connector and the applications which are build using the Unity 3D framework can be deployed and controlled using the hands. One can have Figure 5 as reference to understand the following architectural points of the product.

1) Nexus 5 Smart Android phone is used to process all the applications and also receives the data from leap motion sensor.

2) USB to Micro USB connector which helps to send data from sensor to mobile.

3) USB cable.

4) Leap Motion Sensor.

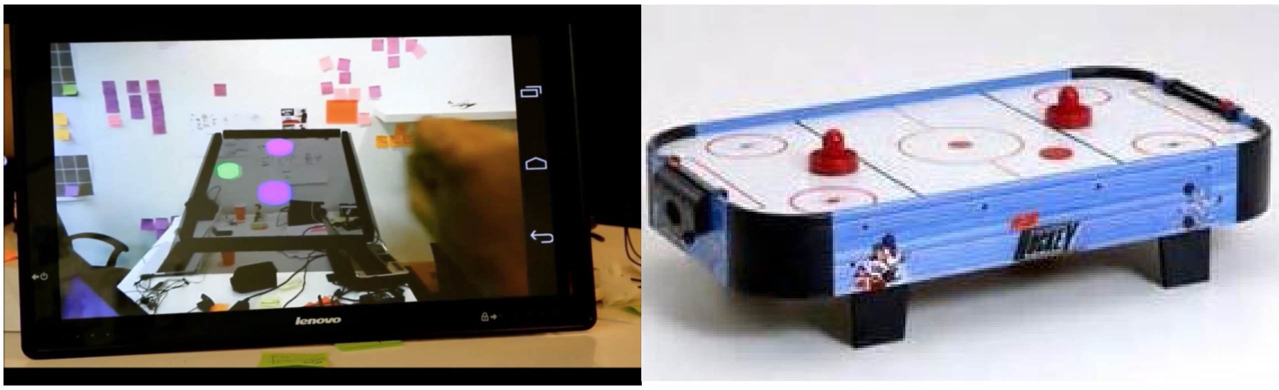

Figure 8. Air hockey or Glow hockey, AR air hockey.

6. Sample AR Game

To get an idea what the app is all about, the following few lines are drafted. We have built the App which replicates the game shown in Figure 6. The basic components involved in this game are, a. Paddle, b. Coin and c. Board. These are technically referred from the app point of view as Unity Game Objects.

The following code is a unity JavaScript code that is being added to the game scene to control the paddle object present.

[C1]The function start will be called as soon as the script is executed. And the initial position of the paddle will be stored which also creates an object for leap controller. [C2] The function change when called will find the difference between the current and previous position of the hand. [C3] Hand Get Front Hand function is used to obtain the hand which is near to the LEAP. In order to check whether the hand is detected by the LEAP or not, we have made the color of paddle to change. If the color of the paddle is red it means hand is not detected and if the color of paddle turns blue, hand is detected. [C5] Update function is called once per frame i.e., this function will be called in regular time interval. [C6] first the current frame is obtained from the controller object. [C7] Then the front hand position is obtained and also the if no hand is detected the program is not allowed to continue. [C8] Change in hand position is calculated and the paddles current position is obtained [C9]. From [C10] we get the new position based on the change in hand position. Paddle is translated depending on the change in hand position. Paddle is restricted from leaving the board. Board's width is set as 6 and length as 8 in this case. [C11] In order to keep the paddle within the limits of the board the following conditions have been used. The conditions are that the paddle can move in the positive y direction by 3 units and the negative y direction by 1 unit since the length of the board is 8 and one half of the board is 4. Similarly the direction in the x axis has been restricted by 3 units in both positive and negative directions. The following snippet of the code is enough for the understanding of above conveyed points.

```
usingUnityEngine;
usingSystem.Collections;
using Leap;
public class paddlecontroller : MonoBehaviour
{
publicGameObjectboard,paddle ;
Controller m_leapController;
[SerializeField] Transform m_parent; public Vector3 inipos;
public static boolisSelected;
public Vector3 lasth;
publicboolinicheck ;
//[C1], Cited in text in the previous paragraph.
void Start ()
m_leapController = new Controller();
inipos = paddle.transform.localPosition; //Paddle object's initial position
inicheck = true;
//[C2] Cited in text in the previous paragraph.
Vector3 change(Vector3 handPos)
if (inicheck)
inicheck= false;
return new Vector3(0,0,0)      }
Vector3 ohandPos = lasth; lasth = handPos;
return new Vector3( (handPos.x - ohandPos.x)*1.5f,
(handPos.y - ohandPos.y) * 1.5f,
handPos.z - ohandPos.z) ;
//[C3] Cited in text in the previous paragraph.
```

```
Hand GetFrontHand(Frame f)
floatzComp = -float.MaxValue; Hand candidate = null; if
(f.Hands.Count == 0)
          {
                    //Paddle becomes red if hand is not
detected
                    GameObject pad = GameObject.
FindGameObjectWithTag ("paddle");
                    pad.renderer.material.color = Color.
red;
          }
          else
          {
                    //Paddle becomes blue if detected
                    GameObject pad = GameObject.
FindGameObjectsWithTag ("paddle");
                    pad.renderer.material.color = Color.blue;
          }
          //[C4] Cited in text in the previous paragraph
          for(int i = 0; i<f.Hands.Count; ++i)
                    {
                    if (f.Hands[i].PalmPosition.
ToUnityScaled().y >zComp)
                    candidate = f.Hands[i]; zComp=f.
Hands[i].PalmPosition.ToUnityScaled().y;
          }
if (inicheck)
lasth = candidate.PalmPosition.ToUnityScaled() ;
return candidate;
}
//[C5] Cited in text in the previous paragraph
void Update ()
{
Frame f = m_leapController.Frame();//[C6]
Hand h = GetFrontHand(f);//[C7]
if (h == null) return;
Vector3 handPos = h.PalmPosition.ToUnityScaled();
Vector3 ch = change (handPos);//[C8]
GameObject paddle = GameObject.
FindGameObjectWithTag ("paddle");
Vector3 curpos = paddle.transform.localPosition;//[C9]
Vector3 newpos = new Vector3 (curpos.x + ch.x, curpos.y +
ch.z, curpos.z);//[C10]
  //[C11]
if (Mathf.RoundToInt (curpos.y + ch.z) >= inipos.y - 1
&&        Mathf.RoundToInt (curpos.y + ch.z) <= inipos.y +
3
&&        Mathf.RoundToInt (curpos.x + ch.x) >= inipos.x -
3
&&        Mathf.RoundToInt (curpos.x + ch.x) <= inipos.x
+3) {
          paddle.transform.localPosition = newpos;
```

7. Conclusion

Currently all AR applications are using touch as primary source for interaction which reduces immersive user experience. By introducing hand gestures control through leap motion sensor we improve user experience in AR applications. We are the first in the world of augmented reality to interface Leap with Augmented Reality. It has now opened doors and avenues for lot of other applications in the same gaming sector and other arenas as well.

8. References

1. Hull JJ, Erol B, Graham J, Ke Q, Kishi H, Moraleda J, Van Olst DG. Paper-based augmented reality. 17th International Conference on Artificial Reality and Tele existence; 2007.
2. Guna J, Jakus G, Pogacnik M, Tomazic S, Sodnik J. An analysis of the precision and reliability of the leap motion sensor and its suitability for static and dynamic tracking. Sensors. 2014; 14(2):3702–3720.