Head Scope

Application for Head Mounted Display

TaeHun Kim
Information System in HYU
2011004426
chutbaksa@gmail.com

Duk Jin Yoon
Information System in HYU
2011004534
yoondukjin@outlook.com

SeungSeon Shin
Information System in HYU
2011004512
vgb1873@gmail.com

Seunggyu Jin
Information System in HYU
2013012740
bernardjin@naver.com

Contents

Role Assignment

Keywords - Head Mounted, Telepresence, Virtual Reality, Google Cardboard

Table 1: Role Assignment

Roles	Name	Task Description and Etc.
User	Kim	Assuming himself as a software user. Discuss about softwares weak point, strong point.
Customer	Shin	Discuss about whole financial costs.
Software Developer	Yoon	Develop entire software. Consider about the program.
Developer manager	Jin	Lay out our rough sketch. Managing our plan.

1   Abstract

Screen displays are primarily advancing to a sharper resolution with larger screens to increase the satisfaction of the consumers. However, in spite of all the upgraded specifications, screen displays still remains as a frame. It is a frame that will only let us view the other world; and we wanted to change this view. Our team wanted to let the viewers to be inside the other world. This principal is called, Virtual Reality. With VR(Virtual Reality), we are able to let users to feel the present in the other world. Our team will implement the VR through Google Cardboard and an Android Smartphone. The Google Cardboard will be used as a viewing tool with asymmetric biconvex lenses, and the Smartphone will be used as a display screen with an application that creates a 3D reality when viewed through the Google Cardboard.

2   Introduction

Films. Film is an incredible medium that is designed with a group of rectangle pictures that are played in a sequence. Though these pictures it can tell us stories in different ways. Film is like a window that lets you see the other world. However, our team wanted something else, we wanted something that allows you to be in the other world. It is called Virtual Reality. Virtual Reality is a machine. But through this machine it allows us to have access to another world and feel present in the world that you are inside. Our team will use the Google Cardboard, which is a box built in the shape of a snorkeling glass. Inside this Google Cardboard there are two lenses that are made of asymmetric biconvex lenses that allows one to experience a 3D reality. Along with the Google Cardboard, we will insert an android smartphone in the form of landscape and build an application that will change 2D to 3D. Inside this application we will install two functions GoogleMap and Cinema Film. Inside this Google Cardboard, it will let you feel like it is real life, and let you feel the

presence of the people you are with. That is why we we chose GoogleMap and Cinema Film. With GoogleMap, it allows users to experience a full 360-degree view of the designated location. It will give them a detailed view and feel as if they are in that location. The Cinema Film will be built so that it allows the users to feel as if they were in the presence of the film. With Google Cardboard, it will let us see not a view into the world, but the whole world stretched in a rectangle. It is a form of media that can change peoples perception of each other. Our team believes that Virtual Reality has the potential that can change the world. We become more empathetic, and we become more connected. Ultimately, we become more human.

3 Android OS based Application

Being a mobile operating system, android OS is a modified version of Linux, originally developed by a start-up, Android, Inc. As Google entered mobile market, it purchased Android and in a bid to encourage independent development works, it released the developer tools under the open source Apache License. The permissive licensing allows the OS and related software to be modified and distributed by enthusiastic developers, network operators and device manufacturers.

3.1 Android Versions

Table 2: Android Version as of May 23, 2015

Version	Code Name
1.5	CupCake.
1.6	Donut.
2.0-2.1	Eclair
2.2-2.2.3	Froyo
2.3-2.3.7	Gingerbread
3.0-3.2.6	Honeycomb
4.0-4.0.4	Ice Cream Sandwich
4.1-4.3.1	Jelly Bean
4.4-4.4.4, 4.4W-4.4W.2	KitKat
5.0-5.1.1	Lollipop

3.2 Technology Features

Android Studio

Android Studio is an integrated development environment(IDE) for developing on the Android platform. It has an installer that sets up the correct Android SDK and a development environment with an optimized emulator and a set of code templates. The build system can generate multiple apks for debug and release. The Lint tools is used to deal with performance, usability, version compatibility, and other issues.

Application Interface and H/W Support

Based on Direct Manipulation, the on screen objects have been programmed to respond to real world actions like swiping, touching etc. Boasting of a fast and responsive fluidic touch screen, the OS supports various dedicated hardware like proximity sensors, gyroscopes, magnetometer and accelerometer etc. The Home Screen is analogue to the Desktop in a Windows OS.

Memory Management

The OS supports multi-threading but depending on the instant memory availability, it can kill application so as to reduce overloading. The RAM management is such so that power consumption is at minimum. As far as third party applications are considered, the SDK provides with ample library entities such as Services, Background Tasks and Foreground Tasks for working with application lifetime.

Security and Privacy

Though the OS is immune to normal user usage, but the security flaws can be exploited, as done by the open source community, to get ROOT access (can be used for malicious purposes by crackers) and modify device capabilities. Except that, the device owners applications are mostly run in an isolated area of OS called sandbox which restricts access to the system resources and hardware unless the user explicitly gives the access permissions during installation itself. Hence, the app gains access to /data partition through this method and the /data partition only. The newest Android OS versions have enhanced security features such as malware scanners built into system to keep a tab on malicious software downloaded through Google Play or any other third party application. Newer applications now rely on OAUTH 2.0 for secure access to internet.

Network Connectivity

The OS supports a full range of connectivity solutions ranging from Bluetooth to ZigBee (through accessory support) and from 2G to LTE support. It supports data packet transmissions through GPRS/EDGE support. Internet can also be accessed through Wi-Fi, WiMAX and shared among other devices through tethering (both over Wi-Fi and USB) support. PC communication is established through device management software using USB and Bluetooth. HTTP service is supported and through use of Google APIs, the phone is an effective GPS enabled device.

3.3 HeadScope

The App "HeadScope" is virtual reality based application which would allow users to view images, watch videos, and even play games in a 3D mode. Depending on their current viewing angle, users are able to turn 360 degree that allows a 360 degree view.

4 Requirement

## 4.1 Requirement for a Head Mounted Display

The "Head Mounted Display" is a virtual reality platform that is used to mount with a smartphone.
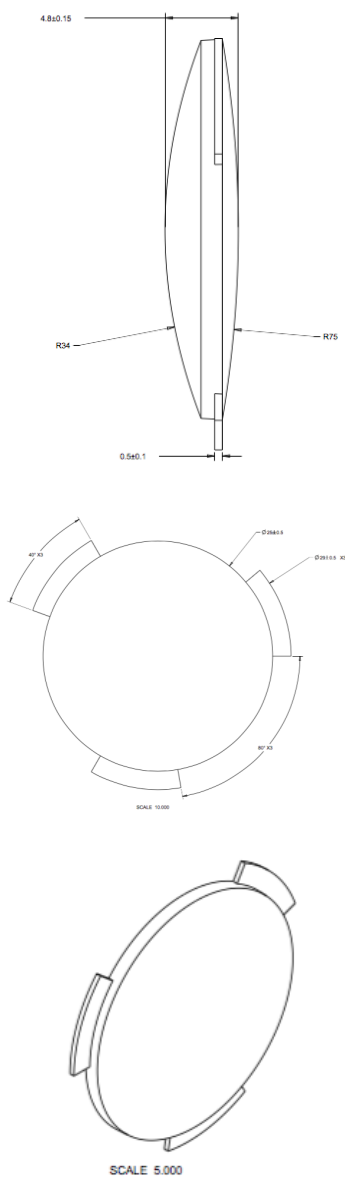
4.1.1 Google CardBoard

The Google Cardboard comprise a piece of cardboard cut into a precise shape, 45 mm focal distance lenses, magnets or capacitive tape, velcro, a rubber band, and an optional near field communication (NFC) tag. A smartphone is inserted in front of the lenses and a Google Cardboardcompatible app splits the smartphone display image into two, one for each eye. The lenses create a distortion effect that sends each half to one eye and creates the impression of a stereoscopic 3D image with a wide field of view.

- **Cardboard**
  Die-cutting template for Google Cardboard can be folded together into a shipping-ready envelope (8.3 x 6.72 x 0.69).
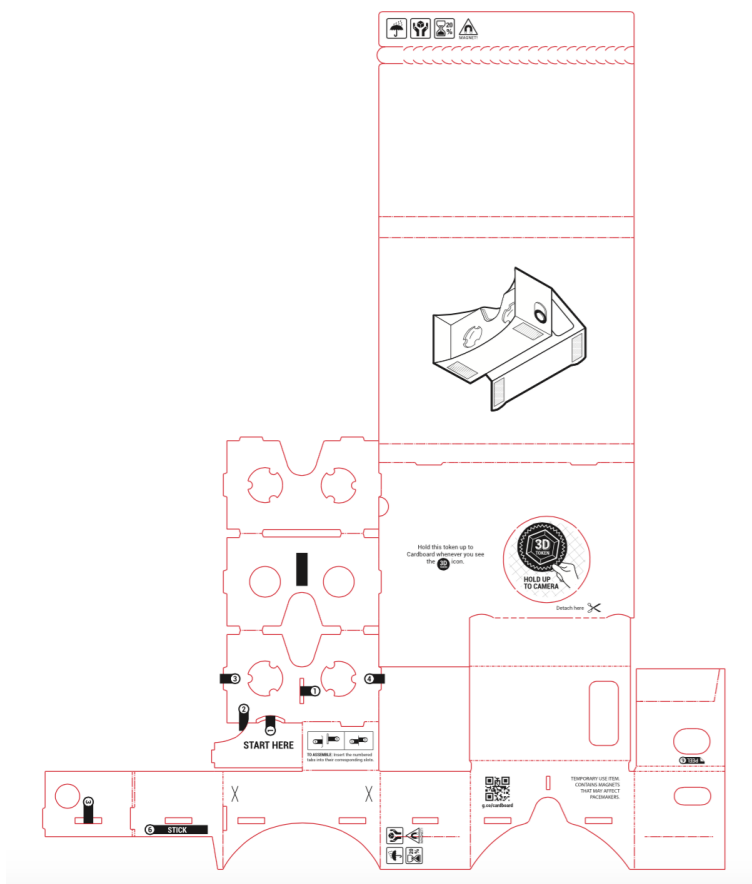
- **Lens**
  Google Cardboard uses 45mm focal distance, asymmetric biconvex lenses. This type of lenses works best to prevent distortion around the edges.

  1. Material needs to be optical PMMA or equivalent
  2. All gates need to be trimmed flush
  3. All contaminants need to be removed from surfaces 6. Lens needs to be RoHS and REACH compliant

- **Magnetic Pull**
  If your viewer is using a magnet based input, use a neodymium ring magnet (at least NH35 grade) as an outside trigger. Google Cardboard ring magnets dimensions are 0.740 x 0.105.
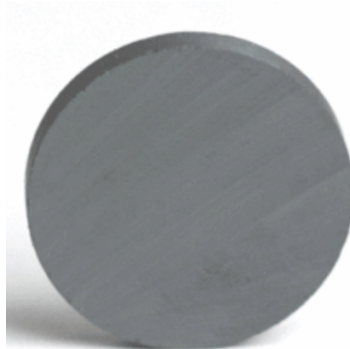
- **Inputs**

  1. Use a ceramic disk magnet with at least C8 grade for Cardboard inside. Ceramic magnets dimensions are 0.701 x 0.197. Make sure to glue the ceramic

  2. disk magnet in place, otherwise the whole magnet pair could come off easily and become a swallowing risk.

  3. You can also use different types of inputs, from a simple smartphone screen touch, to Bluetooth-based buttons, various types of conductive and capacitive inputs, and so on.

  4. Make sure that your viewer contains exactly one input. If your viewer uses the screen touch, ensure that there is at most one dedicated area for touching the screen.

- **Miscellaneous Google Cardboard parts (Velcro, stickers, rubber band, print)**

  1. Google Cardboard uses two strips of regular-strength, adhesive-backed velcro. Approximate size is: 0.75 x 1.25 (20mm x 30mm). Make sure that the adhesive is sufficiently strong to not loosen from the cardboard top over time, as the strips will be getting a lot of use.

  2. Similarly, make sure to use a strong adhesive sticker on the magnet side. This is what holds the whole assembly together, and it tends to loosen over time.

  3. Use the rubber band to prevent the phone from sliding out. It is enough for the rubber band to be wrapped around the bottom of the viewer and touch the bottom of the phone it does not need to be wrapped around the box.

4.2   Requirement for Application Logo

1. We designed our application logo, using Adobe Photoshop CS6 and Adobe Illustrator CS6.

2. Display it on the Smartphone.

4.3   Requirement for loading page

1. Show our application name

2. Show our team name

4.3.1   Requirement for main page

1. It would have five items

2. Jumper

3. 3D Cube

4. Volume

5. Preferences

6. Exit

4.3.2   Requirement for Jumper

1. Click the Jumper Icon among the main page

2. Show the photographs of tourist attraction on the wall

3. To use gyroscope, users will find the photograph which they want to go

4. Users will choose the photograph by pulling down the Magnetic pulling

5. Users are able to trip the attraction zone

6. If the tour is over, users could exit by back button

4.3.3   Requirement for Cine

1. Click the Cine Icon among the main page

2. Show the movie poster on the notice board

3. To use gyroscope and Magnetic button, users will find the movie which they want to see and choose the movie

4. Users could be able to watch the movie to have experience the ambience of a movie theater

5. If the movie is over, users could exit by back button

4.4 Development Environment

Before you begin to format your paper, first write and save the content as a separate text file. Keep your text and graphic files separate until after the text has been formatted and styled. Do not use hard tabs, and limit use of hard returns to only one return at the end of a paragraph. Do not add any kind of pagination anywhere in the paper. Do not number text heads-the template will do that for you. Finally, complete content and organizational editing before formatting. Please take note of the following items when proofreading spelling and grammar:

4.4.1 Choice of software development platform

1. Which platform and why. We will use Windows, Yosemite for android application development.

2. Which program language and why. We will use Java(Cardboard SDK for Android) for internal functions. We also use XML for design layouts. We will use Unity for 3D Modeling.

3. Provide a cost estimation for your built. We need Photoshop CS6 WON 323,158 but we think it is so expensive so we use trial version of Photoshop for free. Next, we need Google Cardboard. We can buy it WON 5,000.
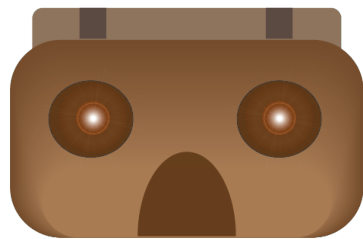
   Android develop
   Windows7 ultimate
   OS X Yosemite 10.10.2
   Android Studio
   Adobe Photoshop CS6 trial version
   Unity 5 Engine

4.4.2 Software in Use

1. There is already a Google Cardboard application that is built by Google.
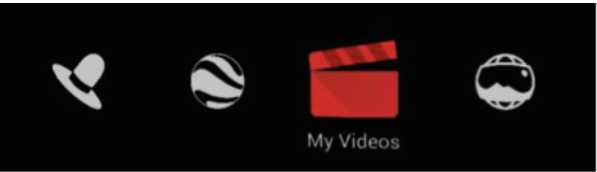
5   Specification

5.1   Specification of App Icon

5.2   Specification of Loading App Page

5.3   Specification of Main App Page

1. The Main App Page would have five applications:

   Jumper
   3D Cube
   Volume
   Preferences
   Exit

2. Due to the gyroscope sensor built in the smartphone, we are able to select different apps inside the Main App Page by turning your head. We built a fixed cursor that will guide the users to what they are selecting. To select the desired app, users will simply pull down the magnetic pull.

5.4   Specification of Jumper
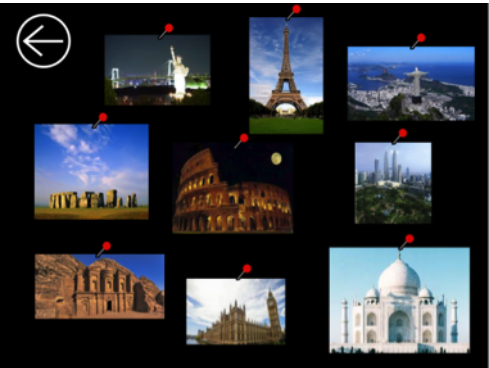
1. The Jumper application will allow users to view an image in a 360 degree angle. This will give them a full view from the side to the back of an image by turning

their head around. Jumper can give the users the experience to actually feel like their in that place.

2. By using gyroscope, users will find the attraction place which they want to go and will choose the photograph.

3. Since the image needs to be 360 degrees there are a limited number of images that we will be implementing. In addition, users are unable to upload their own image. However, our team will continue to update pictures from the famous attraction places such as: Namsan, Eiffel Tower, and even Egypt's Pyramids.

5.5   Specification of 3D Cube
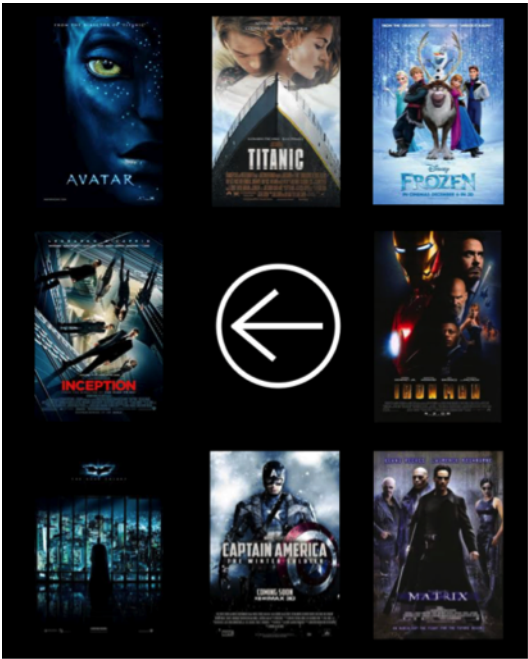
1. Upon clicking the 3D Cube App icon, users will search for the moving 3D Cube object.

2. After the user find the 3D Cube, the users will lower the magnetic button that is installed in the Google Cardboard.

3. This will change the 3D Cube's color.

4. There is a point system within the app that counts the number of times the user find the 3D Cube object.

5.6   Specification of Volume

1. The Volume icon will adjust the amplification of the HeadScope application.

2. Users are able to adjust the volume by turning their head towards the left for a lower sound and to right for a louder sound.

5.7   Specification of Preferences

1. The Preferences App Icon will display the settings based on the HeadScope App.

2. The first setting have a checkbox for smartphones that do not work with the magnetic pull. For those users whom smartphones do not support the Google Cardboards magnetic pull, there is an alternative. Users are able to select apps by holding the cursor shown in the smartphone steadily. Once the users hold the smartphone steadily there will be circle cursor that will fill when not moved. As soon as the circle is filled, the smartphone will select that app.

3. The second setting will contain a checkbox that makes the VR experience more seamless. This can create a smooth transition between apps without exiting the application.

4. The third setting is a function that allows the list of apps to be separated by commas, that the launcher will display.

5. The fourth setting has the same function of the third, but in a blacklist.

6. The fifth setting is the adjustment for the Interpupillary Distance. This setting is the distance between your pupils. 0 is set for pupils that are very close and 100 for pupils that are far apart.

7. The sixth setting is a checkbox for users to disable vibration when selecting an app. When enabled, the phone will initiate a vibration upon selecting a desired application.

8. The seventh setting is a checkbox that will disable the volume buttons in the smartphone. Users will sometime experience situations where they accidentally press their smartphones volume button. For this case, we implemented an override method to disable the volume button when the HeadScope application is running.

9. The eight setting is premium checkbox that when clicked it will give direct users to our teams homepage. This is made just for users to donate to our project.

10. The ninth setting is a checkbox that allows users to modify the Main App Pages background. If not checked the Main App Page will have a default background which is the color of black.

11. The last setting is a checkbox that allows the voice recognition command. To launch the apps in the Main App Page users can also use the smartphones integrated voice command. This is an alternative choice for selecting apps instead of the magnetic pull.

6   Architecture Design and Implementation

Overall Architecture

Directory Organization

Software Module for Launcher

Software modules used in the "Home Launcher" is describe in form of different activities. Multiple activities have been used here to be handled by different functional requirements:

```
1.) private void onCreate(Bundle savedInstanceState)
Function: Called when the activity is starting. Calls method init()

2.) private void print(String request)
Function:prints out String request

3.) private void updateTweens()
Function:calculate accelData by adding tweenStep into original accelData

4.) private void prepareToLaunch()
Function:convert apps status to ready for launch by adjusting boolean
    startingApp to true

5.) private void launchApp()
Function:adjust AudioManager or call method launch()

6.) private void makeImmersive()
Function:compare android version (target SDK_INT >= KITKAT) if true set
    system ui visibility

7.) public boolean isMyAppLauncherDefault()
Function: add activity as a intent. add extra package to array list as an
    item
packageName.equals(activity.getPackageName())

8.) public boolean onKeyDown(int keyCode, KeyEvent event)
Function: compare keyCode to KeyEvents variables (back key, volume
    up/down key). return boolean.

9.) public boolean onCreateOptionsMenu(Menu menu)
Function: Inflate the menu. Add items to the action bar if it is present

10.) public boolean onOptionsItemSelected(MenuItem item)
Function:compare MenuItem items id to boolean which is calculated by
    R.id.action_settings or parent onOptionsItemSelected(Item)

11.) protected void onResume()
Function: for activity to start interacting with the user.
calls private SensorManager mSensorManagers method registerListner that
    return true if the sensor is supported and successfully enabled.
```

12.) `protected void` onPause()
Function:called as part of the activity lifecycle when an activity is
    going into the background, but has not (yet) been killed.
calls `private` SensorManager mSensorManagers method unregisterListner that
    unregisters a listener `for` all sensors

13.) `public void` onSensorChanged(SensorEvent sensorEvent)
Function: Calls when sensor values have changed

14.) `public void` onAccuracyChanged(Sensor sensor, `int` i)
Function: Called when the accuracy of the registered sensor has changed.

`public class MyView extends View`

    `Variables`

`private` Paint paint = `new` Paint();
`private int` width = 0;
`private int` height = 0;
`private` Rect cursorPosition = `new` Rect(width / 2 - 1, height / 2 - 1,
    width / 2 + 1, height / 2 + 1);
`private int` appStartAnimationPosition = 0;
`private` Rect freeAllocate = `new` Rect();
`private int` selectedApp = -1;
`private long` timeSelected = -1;
`private final long` selectionTime = 200;
`long` timeOfLastFrame = 0;
Display dis =
    ((WindowManager)getApplicationContext().getSystemService(Context.WINDOW_SERVICE)).getDe
        Bitmap lEyeBit = Bitmap.createBitmap(dis.getWidth() / 2,
            dis.getHeight(), Bitmap.Config.ARGB_8888);
        Bitmap rEyeBit = Bitmap.createBitmap(dis.getWidth() / 2,
            dis.getHeight(), Bitmap.Config.ARGB_8888);
        Canvas lEye = `new` Canvas(lEyeBit);
        Canvas rEye = `new` Canvas(rEyeBit);

15.) constructor `public` MyView(Context context)
Function: sets variable to `super`(context), paint, width,
    volumePanelWidth, height, timeOfLastFrame. calls method gameloop()

16.) `private void` gameLoop()
Function: calls method move(), updateTweens, invalidate()
`if` width is 0, then reset width, volumePanelWidth, height, cursorPosition
`if` selectedApp is -1 calls `for` loop installedApps.size() times
reset rectangle freeAllocate, selected, timeSelected
`if` starting app
then calls method launchApp() or modify appStartAnimationPosition

17.) `public void` magnetPull()
Function:adjust volume or reset rotation

`float` headMatrix[] = `new float`[16];

```
float headFloats[] = new float[3];

18.) private void move()
Function: adjust volumePanelKnobPosition. set installedApps variable
    through for loop.

19.) float[] getEulerFromMat(float[] rotMatrix)
local variable :
float x, y, z;
float _11, _12, _13, _14;
float _21, _22, _23, _24;
float _31, _32, _33, _34;
float _41, _42, _43, _44;

Function: set array rotMatrix values to float _11~_44. calculate x, y, z
    and return float array[x, y, z]

20.) public float[][] matrixMultiply(float[][] A, float[][] B)
local variable :
int aRows = A.length;
int aColumns = A[0].length;
int bRows = B.length;
int bColumns = B[0].length;

Function: calculate variables. set multiple array C value. return C

21.) protected void onDraw(Canvas canvas)

Function: Draw left, right eyes by using canvas class. Render eyes to
    screen. Draw screen divider
Call method gameLoop()

22.) public boolean onTouchEvent(MotionEvent motionEvent)
Function:Call method magnetPull(). return boolean false
```

6.1   Software Module for Preference

7   Use Cases