Interactive Virtual 3D Gallery Using Motion Detection of Mobile Device

C. Sinthanayothin, N. Wongwaen, W. Bholsithi
National Electronics and Computer Technology Center, NSTDA
112 Thailand Science Park, Phahonyothin Rd, Klong 1, Klong Luang, Pathumthani, 12120, Thailand

Abstract—Normally all digital images are displayed in a flat two-dimensional (2D) image space. However, the three-dimensional (3D) techniques exist that allow users to interact and walk through a virtual environment in mobile devices using OpenGL. So in this paper, an interactive virtual three-dimensional (3D) photo gallery on mobile devices is presented. The system will allow users to take pictures with the mobile device and exhibit in the form of virtual 3D gallery and navigate or walk through the gallery by pressing the button or moving the device. The device has the computational unit which is capable of determining motion data from the g-sensor or accelerometer sensor. The motion data describes the movement of the device including a rotation of the device. Therefore, the benefit of the sensor could be applied such as modifying the view displayed on the screen vertically or horizontally automatically. Also the system can detect tilting left—from right device for viewing image and navigating in the virtual 3D Gallery as well. The result shows that the interactive virtual 3D photo gallery contains digital photos taken by a user and the user can navigate through the gallery by pressing buttons or moving the mobile device.

I. INTRODUCTION

A) 3D graphics on mobile platforms

The creation of Three-dimensional (3D) scenes in real time using mobile devices is becoming more and more usual. 3D graphics on these devices is challenging due to at least two factors that are drastically different from PC systems with graphics cards. First of all, they have very small displays, and second, they have very small amounts of resources for rendering [1]. Thus, when designing 3D virtual environment for mobile devices, the developer should be extremely careful when using memory resources. Fortunately, OpenGLES (Open Graphics Library for Embedded Systems) is available for free as a standard 3D graphics API for embedded systems makes it easy and affordable to offer a variety of advanced 3D graphics and games across all major mobile and embedded platforms. OpenGLES is a low-level, lightweight API for advanced embedded graphics using well-defined subset profiles of OpenGL. It provides a low-level applications programming interface (API) between software applications and hardware or software graphics engines [2][3]. Therefore OpenGLES has been applied for 3D visualization rendering on mobile device in this paper.

B) 3D Motion detection of mobile device.

Motion sensors, such as inertial sensors like accelerometers or gyroscopes, can be used in mobile devices. Accelerometers can be applied for measuring linear acceleration and gyroscopes can be applied for measuring angular velocity of a moved electronic mobile device. Android phone includes an orientation sensor that is used to detect the orientation of the phone in space. The orientation is given through three values [4] as can be shown in Fig 1.

Azimuth is the angle in current reference to Magnetic North. The units of measure are within 0-360 degrees. Pitch is the degree to which the device is tilted forwards or backwards. The measure of pitch is zero when the device is laying flat, -90 when standing upright and 180 when laying flat on the device face. Pitch value can be realized that -90 is the value of the device upstanding and 90 being the value when it is upside down. Roll is the rotation of the device in relation to the bottom left hand corner of the screen. The units returned represent the degree of change in degrees from 0 to +/-90. Zero is when the device is upright standing and forward facing. Any movement left or right will then increases or decreases to 90 or -90 and on to 0 when going past these values.

Fig 1: A perspective view of an orientation sensing mobile device.

C) Interactive virtual 3D gallery using motion detection on mobile device.

All mobile devices are normally display camera images in a flat two-dimensional (2D) screen space. These images can be used to create the 3D object [5][6] or applied for texture mapping in mobile game [7]. Many three-dimensional (3D) games exit that allow users to walk through a virtual environment [8]. However, only a few interactive virtual 3D photo galleries that containing pictures taken by a user and displayed on a display of a mobile device, have been...
developed. Most interactive virtual reality requires users to navigate through the virtual environment using buttons with rotating, zooming and panning [9][10]. GPS could be another option for tracking the user during walk through the virtual reality [11] but the user need to move for some significant distance. Therefore, the purpose of this paper is providing the enhanced functionality of a mobile device by using device motion to navigate through the virtual 3D gallery. Interactive virtual 3D gallery by using device motion can allow easier and quicker than using buttons, as well as reduce scratch on the device from use of physical elements such as touch screen, buttons and so on. Also motion sensing devices enable users to interact with virtual environment with small phone screen since relative motion can be an effective navigation technique to access 3D virtual environment as well. This paper proposes a technique for applying orientation and accelerometer sensing technology for navigating the 3D virtual environment. The method starts with the implementing of virtual 3D photo gallery using OpenGL ES and applying the motion detection data for navigating through the 3D gallery.

II. METHODOLOGY

A. Required Tools

For developing a java application with Eclipse on HTC My Desire HD mobile device, the following SDKs (Software Development Kits) and tools were required:
- Java JDK SE 6.0
- Eclipse SDK 3.6.1
- Android SDK Components
- HTC My Desire HD Connectivity tools

The Android software development kit (SDK) works on Windows, Linux, and Mac OS X. The applications can be deployed on any Android devices. Before coding, the installation of Java, an IDE, and the Android SDK are needed.

B. Implementation

The implementation of a 3D visualization gallery rendered on Android platform for mobile device is written as a java language using Eclipse. The flowchart of how to navigate the virtual 3D gallery with motion detection of mobile device is shown in Fig 2. Starting with the virtual 3D gallery in various styles is created and the coordinate’s positions to attach and display images are located as shown in Fig 3. Next the surface of the wall where the photo will be attached has been removed as can be seen in Fig 4. Then the model of 3D gallery with the coordinates to display images has been loaded into the memory. While all pictures available in the SD Card in mobile device are also connected to the program automatically. Set of images are loaded and displayed as a thumbnail on top of the gallery which allows user to click and choose for display in 3D gallery. After preparing 3D gallery and photo sets then user is able to change the gallery style or random a new set of photos for displaying. In addition, if the mobile devices that have g-sensor or accelerometer sensor (Gyroscopes / Orientation), users are allow to enable it. If sensor is disabling, the 3D gallery can be interact with buttons or touch screen instead. In the case of motion sensor is enable, user can tilt the mobile device in different directions to navigate through 3D gallery. For more details on how to navigate the virtual gallery using motion detection will be described later on. The implementation of a 3D visualization gallery and navigating the virtual gallery using motion detection can be divided into 3 steps as follows.

Fig 2: Flowchart shows how to navigate the virtual 3D gallery with motion detection of mobile device.

C. 3D graphics on mobile platforms

The android.opengl.GLSurfaceView class makes it easier for us to use OpenGL ES rendering 3D object as
primitive in our applications. GLSurfaceView is a good base for building an application that uses OpenGL for part or all of its rendering. For the simplest OpenGL application that requires interactive (such as a game) can be created into three main classes [12].

C.1. public class ProgramActivity extends Activity
Almost all activities interact with the user, so the Activity class takes care of creating a window, which the user interface (UI) can be placed with setContentView(View).

C.2. class ProgramGLSurfaceView extends GLSurfaceView
GLSurfaceView is an API class in Android 1.5 that helps developing OpenGL applications and allows user for interactive with the 3D object such as onTouchEvent.

C.3. class ProgramRenderer implements GLSurfaceView.Renderer
The GLSurfaceView.Renderer interface has three methods:

C.3.1 The onSurfaceCreated() method is called at the start of rendering, and whenever the OpenGL drawing context has to be recreated. (The drawing context is typically lost and recreated when the activity is paused and resumed.) OnSurfaceCreated() is a good place to create long-lived OpenGL resources like textures.

C.3.2 The onSurfaceChanged() method is called when the surface changes size. It's a good place to set OpenGL viewport. The camera can also set here if it's a fixed camera that doesn't move around the scene.

C.3.3 The onDrawFrame() method is called every frame, and is responsible for drawing the scene. The code should start by calling glClear to clear the frame buffer, followed by other OpenGL calls to draw the current scene. After the OpenGL structure has been created. Next the virtual 3D gallery in various styles is created using CAD/CAM software and the coordinates’ positions for attaching and displaying images are located as shown in Fig 3.

Next the surface of the wall where the photo will be attached has been removed as can be seen in Fig 4. Then the model of 3D gallery with the coordinates to display images has been loaded into the memory and display as a 3D view. Fig 5 shows the different between the 3D galleries with and without getting rid of the wall surface. The gallery that has the surface of the wall shows the spreading/diffusing of the color wall to the picture so called Color Bleeding. Once the wall has been taken out, this problem has been solved and all the photo images are clear displayed.

D. Dealing with camera’s images in SD-card
Typically the photos from phone’s camera will be recorded in the SD Card, so the program will connect to the photo folder and retrieve images. Since each photo image that has been taken by default camera setting, has almost 1 MB with the size 3264 x 1952 pixels. So when it is loaded and displayed in the application, the program always crash with OutOfMemory. To fix OutOfMemory problem, image needs to decode and scale for reducing the memory consumption by resample the image down to the smaller size that appropriate for displaying on mobile phone using BitmapFactory.Options.inSampleSize. This technique has been applied twice, first for display as thumbnail and second for display into the 3D gallery as can be seen in Fig 6.

However, the image that can be applied as a texture mapping
that use for display in 3D view needed to have size with $2^N \times 2^N$ pixels. Therefore, the image that will be display as a texture mapping need to be resize in 2 steps. First using `BitmapFactory.Options.inSampleSize` as discussed previously and second using `createScaleBitmap` for setting the large bitmap to the specific size as required.

![Image]

**Fig 6:** 3D graphics of virtual 3D on mobile platforms

**E. Navigating through virtual 3D gallery using motion detection of mobile device.**

`SensorManager` is a class that lets the application accesses the device's sensors. `SensorEventListener` is used for receiving notifications from the `SensorManager` when sensor values have changed. The method `onSensorChanged(SensorEvent event)` of `SensorEventListener` will be called when sensor values have changed. The rotation in degree can be retrieved from `event.values[0]`, `event.values[1]` and `event.values[2]` which represents azimuth, pitch and roll respectively. Azimuth, pitch and roll can be explained as the rotation angles around Z, X and Y axis as can be seen in Fig 1. To rotate the virtual 3D gallery, azimuth value has been applied for gallery’s scene orientation. The azimuth values are within 0-360 degrees when the device is rotated around Z axis (Fig 7A). Since azimuth is the angle in current reference to Magnetic North, so the value is usually not continuously especially when starting azimuth value is close to 359 or 0 as shown in Fig 7B and 7D. However, this can be solved by normalized azimuth value by setting the starting azimuth value to 180 degree (Fig 7C). So when the device is rotated clockwise or anti-clockwise, azimuth value is continued. How the azimuth can be normalized can be explained as the flowchart in Fig 8. Starting with the g-sensor has activated; then the first azimuth value will be defined as variable `SAV` (Starting Azimuth Value). Next azimuth value (`AV`) that has got later, will compare to the `SAV`. The different between `AV` and `SAV` will be used for calculating the number of steps, which controls the rotation of the 3D gallery scene for clockwise or counterclockwise orientation.

![Flowchart]

**Fig 7:** Azimuth value in comparison with XYZ axis and azimuth value before and after normalized.

![Flowchart]

**Fig 8:** Flowchart of how to normalized azimuth value and applied for rotating the 3D gallery scene.

To navigate the virtual 3D gallery, pitch and roll values have been applied for walking forward or backward through the gallery’s scene. The procedure of how-to walk through the 3D
The gallery can be described as the flowchart in Fig 9. The application starts with checking the device’s orientation, if it is vertical then pitch will be applied otherwise roll value will be applied instead. When g-sensor has been activated; then the first pitch and roll value will be defined as variable $SP$ and $SR$ (Starting Pitch/Roll) respectively. Next pitch/roll value ($P/R$) that has got later, will compare to the $SP/ SR$. The different between $P/R$ and $SP/ SR$ will be used for calculating the number of steps, which controls the navigating of the 3D gallery scene for moving forward or backward.

**Fig 9:** Flowchart of how to use pitch and roll values for navigating the 3D gallery scene by walking forward-backward.

### III. RESULTS

The result shows that on HTC My Desire HD phone, rendering speed exceeds 30 fps with a displaying of 22 textured mapping images and 4 different gallery model styles. The size of the APK executable file is less than 200 Kbytes. Users can see the images in device storages via virtual 3D gallery on their mobile phones and interact with them. We provide two options for interaction the virtual 3D gallery on mobile phones. One is interacting 3D gallery in a conventional way using buttons or touch screen. The 3D gallery scene can be rotated, zoomed and navigated based on users’ touch interaction. Another is an interacting 3D gallery using motion detection from g-sensor.

The results of an interactive virtual 3D gallery using motion detection of mobile device can be seen as in Fig 10, 11 and 12. Fig 10 shows the virtual 3D gallery which rotated clockwise or counterclockwise based on device orientation. Fig 11 and Fig 12 show how to walk through the gallery forward or backward based on device moving for the device lying vertical and horizontal respectively.

**Fig 10:** The result shows the virtual 3D gallery which rotated clockwise or counterclockwise based on device orientation.

**Fig 11:** The result of navigating through the gallery forward or backward based on device moving and laying vertically.

**Fig 12:** The result of navigating through the gallery forward or backward based on device moving and laying horizontally.

### IV. DISCUSSION & CONCLUSION

From the methodology, testing and results, it can be discussed as follows. According to developing and testing the 3D gallery, the application is well designed with high performance in term of render speed, number of texture mapping and graphics rendering. However the conversion process from 2D photo images into 3D texture mapping needs some steps for reducing the memory consumption by resample the image down to the smaller size until it can be displayed in mobile application. User interaction is an important component of most graphics applications. So in this paper, the accelerometer sensor has been applied in mobile application for richer user interaction. However, such sensor produces discontinuous data over time. The solution to solve this is by normalized the sensor data before used. This approach has provided good results and could lead to reliable data for applying to navigate the virtual 3D gallery.
After implementation, it can be concluded that the challenges in mobile 3D visualization application and user interfaces relate to small display and the limited amount of interaction hardware compared to the PC. So the 3D visualization applications are difficult to build and require mathematical and programming skills to implement. Moreover, the development of 3D graphics applications is a hard work that consumes much more time than conventional 2D graphics applications, requiring specific knowledge about 3D geometry operations as well. Although 3D graphics libraries such as OpenGLES provide a complete application programming interface (API), they demand significant effort to create even simple visualizations. In particular, most of these APIs require that the application developer cultivate a thorough understanding of a complex programming model, which includes matrix operations, 3D geometry, and lighting models. User interfaces is an area where much innovation will happen at every level. Multimodal interfaces that integrate accelerometer sensor, touch screen, buttons and keyboard input with interactive graphics and sound rendering, and take human perceptual and cognitive capabilities into account, will create interaction that’s easier and more fun. Most mobile devices have a camera; exploring how we can integrate images into 3D graphics application is worth exploring. The future of mobile graphics is exciting, and our community will continue to invent new algorithms, techniques, and applications that exploit the context of mobility [3].

ACKNOWLEDGMENT

This research is under the project “MC-Avatar: Multi-Camera Reconstruction and Augmentation of Human in Virtual Reality”, which is financially supported by Discovery and Development Grant (DD grant) from National Science and Technology Development Agency (NSTDA), Thailand.

REFERENCES