Mobile Augmented Reality of 3D Model in Real-time

Fu Yu
yufu@umich.edu

1. Introduction

Augmented reality has been studied for a long time and has been used in different areas. Traditionally, since most of the augmented reality algorithm have to consider the real-time performance, its study is constrained by the device performance. However, with the emerge of the smartphone platform such as iOS and Android, the cell phone performance developing is on a track similar to that of personal computers and the mobile devices are gaining computing power in the ratio similar to Moore’s Law. Some research such as [7] has proven that the mobile devices have got the computing power to run some sophisticated computer vision algorithm. Considering that the popularity of smart phones among the customers and sets of the integrated sensors such as accelerometer, compass and gyrometer that can help collect more useful data for processing images captured, it is a perfect media to bring the computer vision application to the customers.

1.1. Project Description

In this subsection, I would like to describe what I want to in my project.

There is a basic assumption about scene. There is some hint in the scene that can help infer the pose of the camera relative to the scene. The hint could be a marker like a checkerboard or a image template. In the future, I could substitute the hint with some fast object recognition algorithm. And then I project some interesting or plausible 3D models back to the scene based on the 3D information got from the hint.

1.2. Motivation

To get the 3D information and re-render the images has great applications in people’s life. In general, it can put very useful virtual information into the real scene. For example, in an empty house, the design group could store the decoration design on the server or on the owner’s cellphone. Then the owner could view the design by the augmented house to get a real feeling how the design looks. Also, in a museum and some places of interest, there is too much information can’t be shown physically. However, with the augmented reality technique, people could choose what information to get based on what he is looking at. Also, in the game area, we can use our own yards as arena to play the augmented game.

This work is also helpful for the computer vision research. The goal of computer vision is to help people or machine to see more and understand better. However, most of our algorithms are experimented on the powerful server. An easy-to-use real-time framework can help the researchers get a general feeling that how fast a real time algorithm should be and show their work easily.

1.3. Framework and Project Tasks

There are three main parts in the project, which construct the pipeline of the system. First, images and its related information such as device rotation and displacement are captured by the mobile device. And then in the second part, the data is sent to the kernel algorithm to calculate the pose of
the mobile device. In the last step, based on the pose information, the 3D object is augmented in the scene.

For the first part, the mobile platform is very critical because different mobile platforms provide different features and development environments, which may facilitate the development. Also, different platform has different performance in terms of operating system running and graphics rendering, which are both relevant the performance of the real-time system built on top of it. However, there are several mobile platforms available on the market, such as iOS [5], Android [4], Symbian [3], BMP [6] and so on. Therefore, in the first step of the project, I will explore the development environment of some of the promising platforms and also test the performance of the available devices with different platforms.

For the second part, as a lot of research has been conducted in augmented reality, a lot of methods has been proposed. To find a promising way to do augmented reality, research of the related literature is one the task in this project. I will present my research into the methods in this area in the later and finally I will implement one or two algorithm to build the system.

For the third part, a 3D rendering engine is needed to render the augmented scene. I need to constructed a 3D rendering engine to connect everything together on the selected mobile platform.

2. Technical Tools

As mentioned previously, there are three main parts in the framework.

For the first part, although there are several available mobile platforms. Some of them like Symbian and BMP are very old. The problem with this is that when this kind of operating system first appeared, the mobile devices carrying them has very limited power, which caused a lot of compromizations in the system design. However, due to the compatible issue, most of the compromizations remain up to now, which will make development harder compared to the new mobile operating systems. Therefore, I propose to use iOS or Android for the experiment and development of augmented reality. Nevertheless, it is still not clear which platform is more suitable.

For the second part, the basic algorithm is to calculate the homography between the known image template or marker and the image and then to infer the pose of the camera relative to the plane with the detected clue.

What algorithm to use is a part of the work of this project, since there are many options. However, here are some basic tools to use. To detect the object, there are some fast feature detection algorithm like SURF [1], FAST [10, 9]. To find the optimized homography, we can use Newton method [9], efficient second-order minimization techniques [8]. To decompose homography to get the pose of the camera, we can use the algorithm proposed in [12, 2]

For the rendering part, OpenGL is used to render the augmented scene and help process the images. This also raises a high requirement for the mobile device, since not all of the mobile devices have a dedicated GPU for graphics processing and support OpenGL 2.0, which is very important in computer vision and augmented reality.

3. Milestones Achieved

Test different platforms and build a rendering engine on the selected platform

I first test Android platform by writing a augmented reality prototype in Java. However, it runs very slow. It may be because the runtime of Java is not very efficient, since Java code is run on the virtual machine. Java virtual machine does very bad on the loop of large trunk of data, which is unfortunately needed by the image processing application.

Then I compile the OpenCV on Android. However, although most of the code is in C++, the running time is still not promising and the scheme for
augmented images is the same with my prototype.

Furthermore, I test Qualcomm AR SDK on Android. Although the running time is much better, I cannot get much useful things from this SDK, because it seems build its own framework and the code is not available.

To conclude, Java virtual machine and the Android framework consume a lot of computing resource in the augmented application that make it is hard to build certain real-time system based on the current power of mobile devices. The possible way is to build the augmented reality framework in C++ from scratch, which will take a lot of engineering efforts. Also, it is still hard to debug C/C++ code on the Android devices.

Then I looked at iOS and Table 1 and Figure 1 compare the two platforms. As we can see, iOS can support C/C++ natively, which means it can run code more efficiently. Also, iPhone has a much better graphics processing unit, which is very important in the augmented reality application.

Therefore, I select iOS as my primary platform and build my rendering engine on it.

<table>
<thead>
<tr>
<th>Programming Language</th>
<th>iOS</th>
<th>Android</th>
</tr>
</thead>
<tbody>
<tr>
<td>C Support</td>
<td>Object-C/C++</td>
<td>Java</td>
</tr>
<tr>
<td>IDE</td>
<td>Xcode</td>
<td>Eclipse</td>
</tr>
<tr>
<td>Highest CPU</td>
<td>1GHz</td>
<td>1GHz</td>
</tr>
</tbody>
</table>

Table 1. The comparison between iOS and Android

Research the current methods to recognize image hint and construct the camera pose

Figure 2 [11] shows the taxonomy of currently algorithm for hint detection. For SFM based method, it needs a very good initialization, which is not robust in the practical use. For the model based methods, I choose to use the texture based method, because it may be faster than the optical flow based method and more robust than the edge based method.

To be specific, I may choose to use ESM [8] for homography estimation, because it is much faster than the Newton method, since it doesn’t need to compute the Hessian matrix.

![Fragment Shader Performance Test](image)

Figure 1. The GPU performance between Android phones and iPhone 4. For Android, we choose the phones with the latest chipset from Qualcomm inc., which was actually released later than iPhone 4

![Online monocular MAR taxonomy](image)

Figure 2. Online monocular MAR taxonomy
4. Remaining Milestones

1. Implement the ESM algorithm to get homography (11/14)
2. Decompose the homography matrix to get the 3D pose (11/20)
3. Test template matching and interest point matching (11/27)
4. Integrate the algorithm to the rendering engine (12/7)
5. Finishing the project and prepare for the final report (12/17)

References


