Real Time Augmented Reality on Mobile Phone

Fu Yu
yufu@umich.edu

Abstract

As there are many methods designed for augmented reality, it is still hard to really bring augmented reality application to the real life due to the performance constraint of mobile devices. In this project, I tried to implement certain promising algorithms and explore the possibilities of rich augmented reality application. ESM algorithm is chosen for tracking the clue of the scene together with another detection method. In the off line test, the system runs well. However, the mobile application still needs some improvements.

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 [9] 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 dis-

placement 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 [7], Android [4], Symbian [3], BMP [8] 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 angumented 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. Related Work

2.1. Previous Work

Various approaches for monocular augmented reality have been explored in the computer vision community. There are three main approaches, which are marker based, model based and structure from motion based approaches. A taxonomy of the methods are shown in Figure 2

2.1.1 Marker Based Method

The marker based method have been studied for a long time. Some work [15, 18] on this got very good result. Due to the binary nature of the marker, it is relatively easy to detect the marker in the image. Generally, there are several steps. First the image is binarized. Then the binary is processed to connect the potentially connected components. Last, the information of the binary marker is fitted to the processed image to detect the marker and find the homography between the marker and the actual image. Due to the simplicity and robustness of this method, it is used extensively in all kinds of augmented reality application. ARToolkit is an implementation of the augmented reality framework primarily based on marker. Also, Figure 2.1.1 shows the use of marker detection and recognition in the MAGIC robotics competition.

After conquering the marker based tracking, the computer vision research community shift their attention to the more general case augmented reality, which are model based tracking and even the structure from motion based tracking.

Figure 1. Picture of a robot of UM MAGIC team

2.1.2 Model Based Method

The model based approaches require a priori knowledge about the real scene, like the 3D model or the image template that will appear in the scene. The model based technique can be classified in three categories. The first category consists in methods that take only the objects' edges into consideration while doing tracking. The second one relies on the optical flow of the image sequence, while the third one comprises the use of objects' texture information to perform tracking.

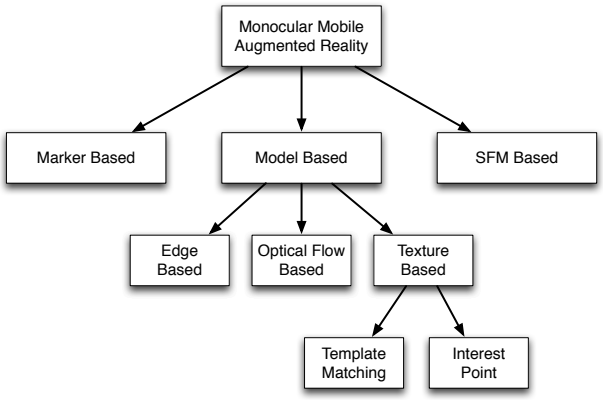

Figure 2. Online monocular MAR taxonomy

Edge Based: In this category, camera pose is estimated by matching a wire frame 3D model of an object with the real world image edge information. This matching is achieved by projecting the model onto the image and minimizing the dispalcement between the projected model and the imaged object. In order to accomplish this task, a good initial hint about the object pose is needed. Sometimes, the initialization process is done manually, like in [14]. Some automatically initialization process using search has been proposed, like in [13]. Once the first pose is estimated, is is used to project the model onto the next frame. Assuming that camera displacement between consecutive frame is relatively small, using the previous pose estimation to predict the next pose will not harm the matching process.

Edge based technique were the first approaches to real-time 3D object tracking. Due to their low complexity, they are easy to implement and have a good performance. Because they only use edge information, edge based approaches are able to track specular objects affected by environment lighting. However, edge based methods usually do not support fast camera movements, since the projected model will be too far from its correct location. Another problem is related to matching errors, which may be caused by elements such as cluttered background or shadows in the image.

Optical Flow Based: Differently from edge based methods, which rely on spatial information obtained by image-model matching, optical flow based tracking exploits temporal information. This is extracted from the relative movement of the object projectiong onto the image. After initialization, which is often manual, the optical flow between the frames captured at time t and t+1 is calculated. Then, the algorithm determines which points from the model projected onto the image at time t are still present in the image at t+1. The displacement of these points over time is calculated using an algorithm such as the Kanade-lucas (KL), described in [10]. This is used to estimate camera movement.

Due to its integration over time, 3D tracking based on optical flow presents smoother changes between consecutive poses. Another advantage is the moderate processing load needed. However, optical flow techniques tend to accumulate errors produced by sequential pose estimations, lead to a deviation from the correct camera calibration. Optical flow algorithms are also not robust against lighting changes and large camera displacements, originating errors in object tracking and requiring re-initialization.

Texture Based: This category of techniques takes into account texture information presented in images. Naturally, there are two subcategories in this section, which are template matching and interest point matching.

The template matching approach is based on global information, unlike feature based techniques. The strength of this subcategory lies in its ability to treat complex patterns waht would be difficult to model by local features. These techniques are also called sum-of-square-difference, as they consist in minimizing the difference between a region of the image and a reference template. Such technique search for the parameters of a function that warps a template into the target image, so that tracking can be done. According to [10], this is the general goal of the KL algorithm. In [5], the author shows an approach based on the Jacobian of the warping function used in the KL algorithm to do 2D tracking. However, there are some problems with variations in illumination and partial occlusions.

The subcategory of interest point based techniques takes into account localized featurs, instead of a global search used by template matching technique. As a result, this subcategory is less computer-intensive than former ones. Another advantage is the fact that illumination changes are easily achievable. In [12], the author mentions that as no inter-frame assumption is make, it allows a wider baseline than optical flow.

There is also an interest point technique based on

tracking planes, instead of full 3D models. The main idea here is to explore the homography formed by the plane in two consecutive views. This computation is performed using the RANSAC algorithm, and recursively determines which homography is correct.

2.1.3 SFM Based Method

Instead of relying on previously obtained information about the scene to be tracked, some techniques estimate the camera displacement without an a priori knowledge about the environment. SFM based techniques are mainly online, since they do not require any previous offline learning phase. Due to this, it is possible to reconstruct a totally unknown environment on the fly. As a drawback, SFM approaches are often very complex. They also have some constraints related to their real-time nature.

In particular, MonoSLAM was created based on the probabilistic SLAM methodology using a single freely moving wide-angle camera as the only sensor and with a real-time constraint [2]. To initialize the system, a know picture is necessary to be present in the initial frame at an approximated certain distance. Then the features from this frame are used to initialize the environment. When subsequent images come into the system, more features are obtained. The features are inserted in a probabilistic feature based map that is maintained during all the lifetime of the operation and is updated by the Extended Kalman Filter. The map grows as new features are added.

2.2. Purposed Method

Since the goal of my project is to understand the environment on the mobile phone in real time, I have to use some robust and efficient methods in my application. In terms of robustness, marker based method should be used. However, it is not quite interesting to put a marker in the scene and the interaction between the user and the scene is harmed. On the other hand, to run the SFM based method on mobile phone, to get the intended performance, the accuracy is compromised. Therefore, I tried to use the image template based method. However, the image template based tracking requires a good initialization and sometimes it needs manually correction when failed. Therefore, I purpose to use the image template detection to help the tracking. Figure 2.2 shows the idea. First, the detection algorithm finds the image template, such as the Lenna picture and calculate an initial homography. Then, the homography is used by the tracking algorithm. When the tracking algorithm finds that the image template is lost by checking the tracking error, the detection is used again to do the correction.

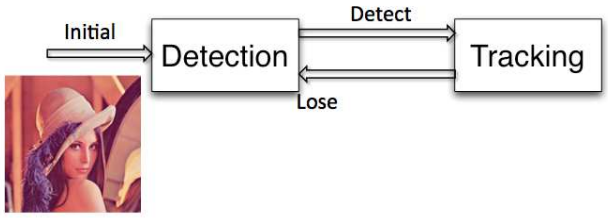

Figure 3. Framework of my application

3. Framework Details

For the detection part, some basic methods are considered since this part is not emphasized in this project. The algorithm first extracts features from both the template image and the current image using methods like SURF [1], FAST [17, 16] . Then the features from both set are tried to be matched based on a threshold. The threshold is set such that the matching is distinct enough. Then based the matching, RANSAC is used to calculate the homography between the template image and the final image.

For the tracking part, Newton method [16] is proved to have a high convergence rate, which is quadratic. However, since we need to minimize the first order derivative of the cost function, Hessian matrix is needed to do the minimization, which is both computationally expensive and unstable. Several first-order approximation were purposed, like Gradient descent, Gauss-Newton and Levenberg-Marquardt method, which tried to get around with the Hessian matrix. But since they approximate in first-order, the feature of high convergent rate is lost.

In this work, another approximation method called Efficient Second-order Minimization (ESM) techniques [11] is explored to solve the tracking problem. In this method, only the Jacobian matrix is needed and therefore it is faster than the Newton method and more stable. Due to the successful implementing stories of this algorithm, it has been commercialized. Therefore,

I think it is worth and promising tying to implement the algorithm and use it in the application.

4. 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 4 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.

	iOS	Android
Programming Language	Object-C/C++	Java
C Support	Native	JNI
IDE	Xcode	Eclipse
Highest CPU	1GHz	1GHz

Table 1. The comparison between iOS and Android

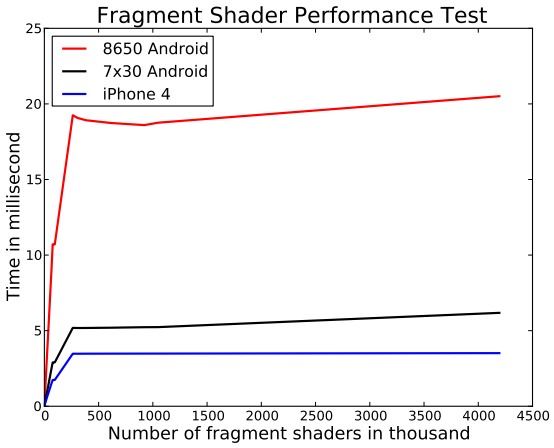

Figure 4. The GPU performance between Android phones and iPhone 4. For Android, we choose the phones with the latest chipset from Qualcomm inc., which was actually released later than iPhone 4

5. Mathematical Framework

I would like to introduce mathematical frameworks for ESM and Matrix Exponentiation here.

5.1. ESM

Let I^* be an image containing the reference template of an object we aim to track , and let I be the current image of the observed scene. Let p^* be the set of coordinates of the projections in the reference image I^* of a set of 3D points lying on the object of interest. Tracking the reference template means finding the projective space automorphism w that minimize:

$$\sum_i (I(w(p^*)) - I^*(p_i^*))^2$$

In this project, we only consider a planar object and w will be based on a homography G parametrized over a vector x. In this project, the parametrization of G is done by matrix exponentiation of x.

During tracking, an approximation $\hat{G}$ of the true automorphism $\bar{G}$ is availabe, and the problem can be redefined as finding an incremental transformation $G(x)$ such that the composition of $\hat{G}$ and $G(x)$ gives the true automorphism $\bar{G}$. Then, the problem consists in finding the optimal parameters $\tilde{x}$ that minimize:

$$\frac{1}{2}||y(x)||^2$$

where $y(x)$ is the vector made of the image difference

$$y_i(x) = I(w(\hat{G}G(x))(p_i^*)) - I^*(p_i^*)$$

. In ESM, we use the second-order approximation:

$$y(x) = y(0) + J(0)x + \frac{1}{2}M(0,x)x + O(||x||^3)$$

where $J(x)$ is the Jacobian matrix of vector $y(x)$ with respect to the motion parameters x, $M(x_1, x_2) = \nabla_{x_1}(J(x_1)x2)$ is based on the Hessian matrices, and $O(||x||^i)$ is a remainder of order i. The cost function is minimized iteratively by estimating $\tilde{x}$:

$$\tilde{x} = J^{\dagger}y(0)$$

5.2. Matrix Exponentiation

As mentioned in the previous subsection, the homography G is parameterized by a vector x using the exponential map of a matrix. The projective transformation matrix $G(x)$ is in the group $SL(3)$ which is Lie group.

Let $A_1, A_2, \ldots, A_8$ be a basis of the Lie algebra. A matrix can be written as

$$A(x) = \sum_{i=1}^{8} x_i A_i$$

A projective transformation $G(x)$ in the neighborhood of I can be parameterized as follows:

$$G(x) = \exp(A(x)) = \sum_{i=0}^{\infty} \frac{1}{i!}(A(x))^i$$

To calculate the exact value of the matrix exponentiation, Pade approximation and the scaling and squaring method [6] are used together to get a relatively accurate value with fast convergence.

The code for the algorithm is like:

```
Mat X = A.clone();
double c = 1.0 / 2;
Mat E = I + c * p;
Mat D = I - c * p;
int q = 6;

for (int k = 2; k <= q; k++) {
  c = (double)c * (q - k + 1)
        / (k * (2 * q - k +1));
  X = A * X;
  E = E + c * X;
  if (k & 1)
    D = D - c * X;
  else
    D = D + c * X;
}
```

6. Experiments

Figure 6 examines the number of loops I should use in the matrix exponentiation calculation by showing the residue when adding the loop number. The entries in the matrix is a random number between 0 and 1. As you can see, after 6 loops, the machine error is researched. Therefore, in my implementation, 6 loops are used to calculate the matrix exponentiation.

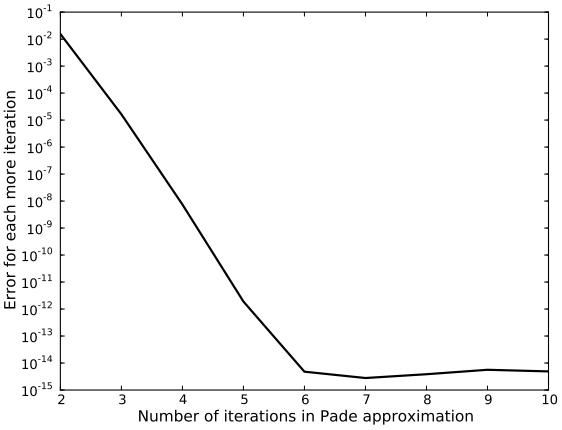

Figure 5. The convergence of the Pade approximation

Currently, SURF features are extracted and matched. The result for the object detection is shown in Figure 6. As you can see, although there are some outliers, the homography found is accurate.

For the tracking part, when there is a good initial homography and the image is not blurred too much, the tracking is normally successfully, as shown in Figure 6. However, sometimes, it doesn't track the image template very well, as shown in Figure 8, in which case the detection will correct the homography.

Finally, Table 6 shows the performance comparison of the detection and tracking algorithm, which is why I want to use the tracking algorithm as much as possible.

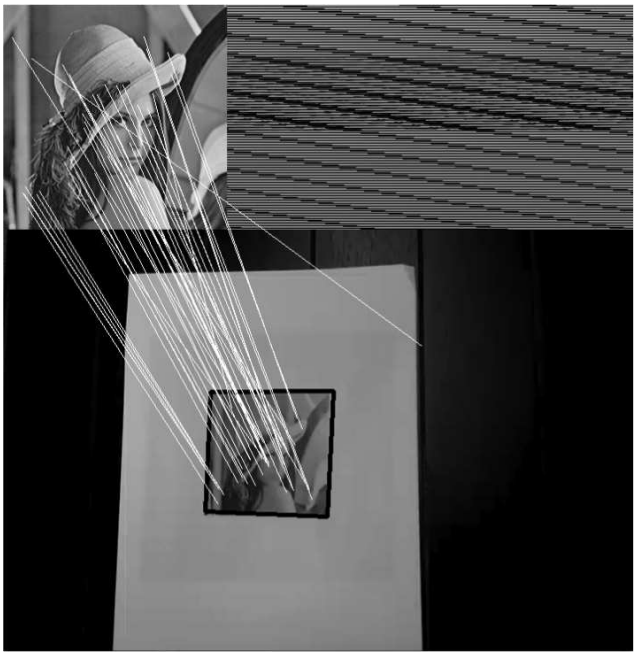

Figure 6. The feature matching using SURF

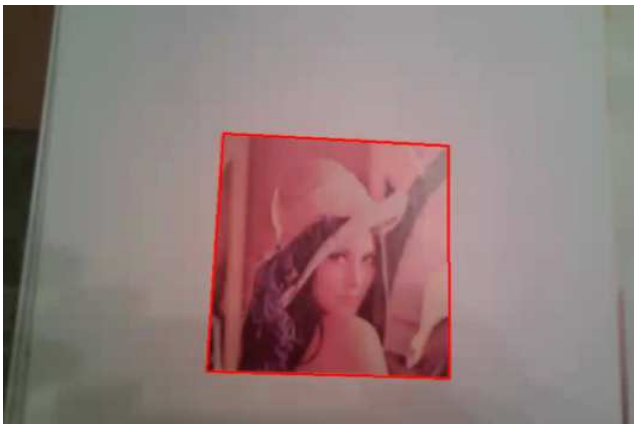

Figure 7. A successful case of ESM tracking algorithm

The experiment is done on the server with 2.67Hz CPU

Template Size	128 by 128	256 by 256	512 by 512
Detection	100ms	210ms	229ms
Tracking	30ms	128ms	550ms

Table 2. The performance comparison between the detection and tracking algorithm

Figure 8. A failing case of ESM tracking algorithm

7. Conclusion and Future Work

In the offline mode, the detection with tracking work flow works quite well. However, on the mobile device, the detection is very slow, which directly affects accuracy of the tracking algorithm.

Therefore, in the further improvement, a faster detection scheme shall be sued. SURF is not suitable for the mobile platform and may be replaced by FAST.

From the homography of the image, we can get more information about the environment and some graph can be rendered in the images in real time to increase the interaction.

References

[1] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. Surf: Speeded up robust features. In *In ECCV*, pages 404–417, 2006.

[2] A.J. Davison, I.D. Reid, N.D. Molton, and O. Stasse. Monoslam: Real-time single camera slam. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 29(6):1052 –1067, 2007.

[3] Symbian Foundation. `http://www.symbian.org/`.

[4] Google. `http://www.android.com/`.

[5] G Hager and P Belhumeur. Efficient region tracking with parametric models of geometry and illumination. *Pattern Analysis and Machine . . .*, Jan 2002.

[6] N Higham. The scaling and squaring method for the matrix exponential revisited. *SIAM Journal on Matrix Analysis and Applications*, Jan 2005.

[7] Apple Inc. http://www.apple.com/iphone/ios4/.

[8] QUALCOMM Incorporated. http://www.brewmp.com/.

[9] G Klein and D Murray. Parallel tracking and mapping on a camera phone. *Mixed and Augmented Reality, 2009. ISMAR 2009. 8th IEEE International Symposium on*, pages 83–86, 2009.

[10] B Lucas and T Kanade. An iterative image registration technique with an application to stereo vision. *International joint conference on artificial ...*, Jan 1981.

[11] E Malis. Improving vision-based control using efficient second-order minimization techniques. *Robotics and Automation, 2004. Proceedings. ICRA '04. 2004 IEEE International Conference on DOI - 10.1109/ROBOT.2004.1308092*, 2:1843– 1848 Vol.2, 2004.

[12] S. Ravela, B. Draper, J. Lim, and R. Weiss. Adaptive tracking and model registration across distinct aspects. In *Intelligent Robots and Systems 95. 'Human Robot Interaction and Cooperative Robots', Proceedings. 1995 IEEE/RSJ International Conference on*, volume 1, pages 174 –180 vol.1, August 1995.

[13] Gerhard Reitmayr and Tom Drummond. Initialisation for visual tracking in urban environments. In *Proc. ISMAR 2007*, pages 161–160, Nara, Japan, Nov. 13–16 2007.

[14] Gerhard Reitmayr and Tom W. Drummond. Going out: Robust tracking for outdoor augmented reality. In *Proc. ISMAR 2006*, pages 109–118, Santa Barbara, CA, USA, October 22–25 2006. IEEE and ACM, IEEE CS.

[15] J Rekimoto. Matrix: A realtime object identification and registration method for augmented reality. ... *Human Interaction*, Jan 2002.

[16] Edward Rosten and Tom Drummond. Fusing points and lines for high performance tracking. In *IEEE International Conference on Computer Vision*, volume 2, pages 1508–1511, October 2005.

[17] Edward Rosten and Tom Drummond. Machine learning for high-speed corner detection. In *European Conference on Computer Vision*, volume 1, pages 430–443, May 2006.

[18] D Schmalstieg and D Wagner. Experiences with handheld augmented reality. ..., Jan 2008.