PTAMM-Plus: Refactoring and Extending PTAMM

Thanh Nguyen* Christian Sandor† Jun Park‡

University of South Australia
Magic Vision Lab

ABSTRACT

Augmented Reality (AR) is a promising candidate for the next generation of human computer interaction. However, a great number of obstacles have to be overcome in order to make AR become widely-acceptable. Among those obstacles, tracking is one of the major well-known challenges. The state-of-the-art in AR proved that PTAMM is one of the most promising tracking systems. However, it is quite difficult to develop AR applications based on PTAMM system because of its monolithic architecture. Therefore, modularizing it for seamless integration is needed. On the other hand, PTAMM distorts graphic results instead of undistorting the video images, the result of which is a distorted augmented view. In this paper, we present PTAMM-Plus, which refactored and extended PTAMM by developing a well-formed API, re-arranging threads, and improving the undistortion mechanism. Using PTAMM-Plus, AR developers may easily build AR applications independently on top of PTAMM. By rearranging threads and undistortion mechanism, presented PTAMM-Plus also enhanced performance and accuracy. In this paper, a testing oriented approach in software development was applied when implementing the PTAMM-Plus.

Index Terms: D.1.3 [Concurrent Programming]: Parallel programming— [I.3.6]: Computer Graphics—Methodology and Techniques I.4.1 [Digitization and Image Capture]: Camera calibration—Imaging geometry I.4.8 [Scene Analysis]: Tracking—[H.1.2]: Models and Principles—User/Machine Systems

1 INTRODUCTION

Augmented Reality has evolutionarily changed our reality in several aspects during the last decade. More than anything else, vision-based tracking technologies are near commercial-product quality. One of the most advanced and promising tracking systems is PTAMM(Parallel Tracking and Multiple Mapping) [2], which is an extension of PTAM(Parallel Tracking and Mapping). Using PTAM, 3D point maps can be built while tracking on unprepared environments. It applies fast corner detection algorithm for detecting features and Lie group algebra for estimating 6DOF camera pose. PTAMM, in addition, allows for multiple maps, detecting and loading the matching map from the map database. Using PTAMM, a user may scan over an area of interest and observe annotations attached to the matching map.

However, PTAMM is known to be a highly monolithic system, and hence it is difficult to improve performance or integrate with other systems. Consequently, building a new AR application on top of PTAMM is considerably difficult although tracking capability of it has high potential for successful application development. For this reason, demand for PTAMM wrapper is very high. There are also rooms for accuracy improvement. PTAMM applies distortion on rendered graphics instead of undistorting the video images, probably for performance reasons. But the results of which are distorted AR images.

In this paper, we presented PTAMM-Plus, which improved PTAMM in terms of seamless integration with other systems, more natural blending of real and virtual contents, and performance. For black-box approach integration, we refactored PTAMM and developed interface wrapper. By applying undistortion on wide images, we enabled undistorted AR images, naturally overlaying virtual objects on real images. Finally, we re-arranged the threading mechanism to compensate for performance degrade due to applying image undistortion. As a result, computation overhead was even smaller than the original PTAMM. PTAMM-Plus is not just a wrapper, it is a refactorization and extension of PTAMM, which also enhanced accuracy, performance, and adaptability.

2 RELATED WORK

From the dawn of augmented reality, the pioneers of AR have focused on embracing the challenges inherent in vision-based tracking. Two major approaches for Vision-based AR tracking are marker-based and natural feature-based. Marker based tracking is generalized as being off-line because it uses prior knowledge about markers for tracking. In the off-line segment, ARToolKit[7] has been recognized as the best candidate and widely used in AR systems. ARToolkitPlus[11] was developed based on ARToolKit, improving interface (by providing class-based API), performance, and stability.

Natural feature-based tracking technologies have recently improved, demonstrating feasibility. There are three types of natural features that are used for AR tracking: point features, edges, and textures. Each of feature has its own strengths and weaknesses. Point feature based tracking has been known to have advantages in computational cost and real-time performance[8, 9]. Edge features are robust and suitable for building structures[10]. Textures are highly applicable in large scale situations because of benefits from distinguish-ability[6].

Hybrid technology in tracking, for indoor and outdoor environments, has increasingly become popular since early 2000s. For indoor solutions, the most well-known one may be combining inertial sensors, camera and land-markers. For instance, Foxlin and Neimark[5] have built a hybrid system which includes an inertial sensor InertiaCube2, a low-resolution camera, and several land-markers.

On the other hand, outdoor motion tracking problems seem harder than indoor ones. The most widely adopted hybrid solution, for outdoor combines orientation sensors and GPS receivers. For example, Azuma et al. [1] have developed an Augmented Reality system for outdoor, which includes 3DOF orientation sensors and a GPS. He found that error rate was quite high because of limitations of GPS and orientation sensors.

From examining outdoor tracking problems, we believe that hybrid solutions, which combine several commercial-off-the-shelf sensors without utilizing computer vision technologies, may not achieve in reliable tracking information. Limitation of traditional hybrid solutions, as discussed, has lead researchers to an alternative approach which combines traditional hybrid solutions and advanced computer vision technologies. For example, Reitmayr et

*e-mail: thanhnguyen.cs@gmail.com
†e-mail: christian.sandor@unisa.edu.au
‡e-mail: joseph.j.park@gmail.com

20th International Conference on Artificial Reality and Telexistence (ICAT2010), 1-3 December 2010, Adelaide, Australia
ISBN: 978-4-904490-03-7 C3450 ©2010 VRSJ

al.[10] have developed an AR system combining a GPS and several orientation sensors with edge recognition technique by applying a Kalman filter.

Texture-based approch is computationally high comparing with other two methods. While edge-based vision techniques are applicable to building structures, point feature-based methods can be more widely applicable. Among point-feature based tracking methods, PTAMM is one of the most promising and stable method. However, PTAMM is known to be difficult to integrate with other systems because of its monolithic nature. In this paper, we introduce PTAMM-Plus, which is a better architecturally-designed and improved version of PTAMM as ARToolKitplus is an improvement of ARToolKit.

3 System Design

When refactoring a highly optimized system like PTAMM, it is very important to choose an appropriate strategy. Because of the high level of optimization in PTAMM, there are many dependencies between components and even some deliberate violations of encapsulation principles. As a result, it is very challenging to modify, for example, to connect external software components with PTAMM. Therefore, we had to carefully approach this problem.

In the following, we explain how we have overcome these problems. First, we present the components of our extension. Second, we present how we improved the threading model of PTAMM. Finally, after having modularized PTAMM, we have integrated it with another AR framework to prove the successful modularization of PTAMM.

3.1 Component Overview

Figure 1 depicts an overview of PTAMM-Plus, by showing which additional components have been introduced and which components of PTAMM have been modified. Before giving a brief description of each additional or modified component, we explain the basic groups of components in PTAMM. PTAMM contains four main groups of components: UI, core, utilities and IO. The core components in PTAMM perform the central control tasks, such as tracking and mapping. The UI components are built for user interface and event handling. The IO components handles camera and image processing. The Utilities components are considered as add-ons to the system, because they contain additional functions which make the system usable.

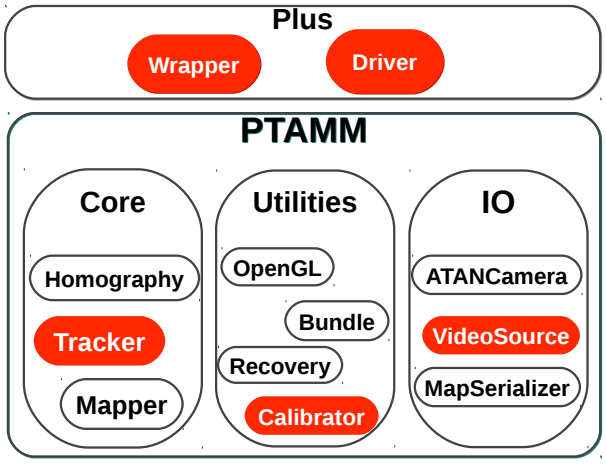


Figure 1: Overview of PTAMM-Plus (modified and new components are highlighted).

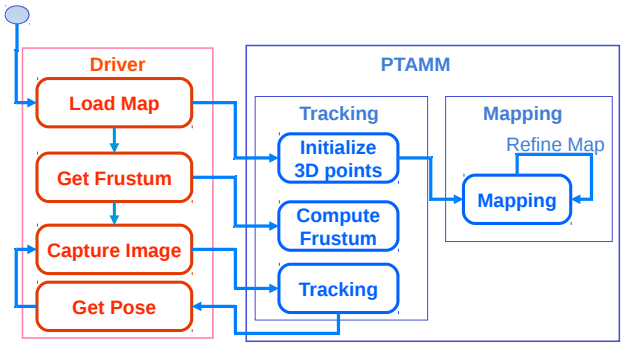


Figure 2: Collaboration diagram of PTAMM-Plus.

Wrapper The Wrapper component is used to interface with other applications or frameworks. Section 3.3 describes how we have used the Wrapper component to interface PTAMM-Plus with another framework.

Driver The Driver component is responsible for coordinating the interactions between the other components; for example, it coordinates the two core activities in PTAMM: tracking and mapping (see Figure 2). During the initialization of the Driver component, the Tracker component must be provided with an existing map. Next, the main loop is entered. Whenever an image is provided (typically by the Video Source component), it will be passed to the Tracker component, which determines the camera pose, which will be passed back to the Driver. The detailled threading behaviour of this process is described in Section 3.2.

Tracker We have modified the original PTAMM tracker by adding three additional methods. The first method allows the Driver component to get the frustum matrix from the Tracker component (for example, to provide it to outside renderers). The next two methods, have been added to make the tracking loop more modular. One method has been added for providing the Tracker component with a camera image. A complementary method has been added to acquire the calculated pose from the Tracker component.

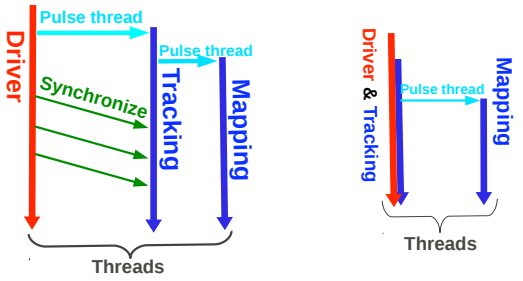
Calibrator In order to support our improved undistortion method (see Section 4), we had to modify PTAMM's Calibrator component. For example, we had to add the abilities to compute an undistortion map and generate a central cut-out. Furthermore, we have added the capability to use the undistortion mechanism of OpenCV[1].

3.2 Threading

PTAMM is a multi-threaded program; UI and tracking are optimized within one thread while mapping runs on another thread. A third thread is used to handle input from the command line; this is for debugging purposes only. Further investigation reveals that the central part in PTAMM is the main loop in the main thread, which is responsible for grabbing images and estimating the camera pose. The main thread in PTAMM includes: UI, tracking, and central control.

In order to improve PTAMM's performance, we have consecutively removed functionality from the main thread. To support this activity, we have created an additional threading model, besides the standard deployment threading. In the following, we describe these two threading models.

[1] http://www.opencv.org



(a) Development (b) Deployment

Figure 3: In development stage, a seperate thread (Driver) is implemented, which synchronously communicate to Tracking thread of PTAMM. However, in deployment stage, Driver thread and Tracking thread are merged together.

Multiple Threads in Front-End for Development In order to improve the speed of the Main thread, we have migrated functionality from the Main thread into the Driver thread. During development, the Driver thread is synchronously connected to PTAMM's threads (see Figure 3a). This made development and testing easier, as changes could be tested outside PTAMM. During development, the Driver thread runs first as the main thread. Then, the tracking thread from PTAMM is initialized and updated continuously. At the moment of being started, the tracking thread in PTAMM initializes and starts the mapping thread. The mapping thread runs asynchronously with the tracking thread, while the Driver thread and tracking thread are synchronized (see Figure 3a). However, because of synchronization, performance of the whole system suffers. Therefore, it is needed to merge the Driver thread and the tracking thread at deployment time, in order to improve performance.

Merging Threads for Deployment In the deployment stage, the Driver thread and PTAMM's main thread are merged (see Figure 3b) in order to improve performance of PTAMM-Plus.

3.3 Integration with TINT

PTAMM-Plus has been successfully integrated into TINT[4], the AR framework used in our lab. The integration was straightforward, as only the Wrapper component of PTAMM-Plus had to be exposed. We believe that an integration into other frameworks would be as straightforward. In the case of the integration with TINT, the main challenge was to communicate efficiently between C++ (PTAMM-Plus) and Python (TINT). We have addressed this problem by employing Python Boost[2] to automatically generate a Python extension module.

Figure 4 shows the resulting component model. Compared to Figure 2, it contains two additional layers. First, the Python Boost layer, as explained in the previous paragraph. Second, it shows the Python component in TINT, which is using PTAMM-Plus. Inside TINT, this component is a sensor node in the dataflow network between input sensors (more details on TINT's dataflow network are given in[4]).

4 Undistortion

PTAMM has been proven to gain advantages from using a fish-eye lens. This type of lens allows PTAMM to discover more features in the environment per frame than a normal lens. However, it will also distort video image; as a result, straight lines in the world will appear as distorted lines. Therefore, un-distortion must be applied in order to realistically render and blend graphics and video image.

[2] http://boost.org

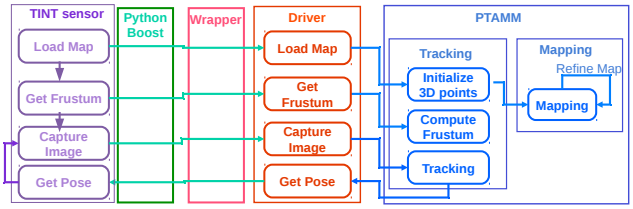
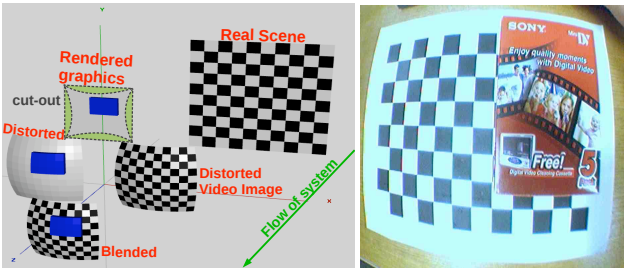


Figure 4: Collaboration diagram of PTAMM-Plus and TINT.



(a) (b)

Figure 5: PTAMM's distortion pipe-line.

4.1 PTAMM's Camera Model

The camera model in PTAMM is the Tang model. It is believed that using this type of model has advantages for developing mathematical framework. This model helps developing and explaining distortion model more effective. Mathematical framework of FOV model:

$$x' = c(x/z) \quad (1)$$

$$y' = c(y/z) \quad (2)$$

$$c = r'/r \quad (3)$$

$$r = \sqrt{(x^2+y^2)/z^2} \quad (4)$$

$$r' = (1/w)arctan(2r*tan(w/2)) \quad (5)$$

$$u = c_x + f_x x' \quad (6)$$

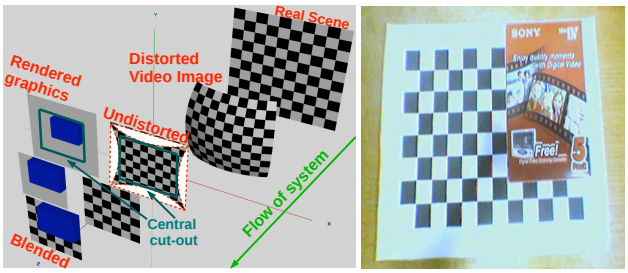
$$v = c_y + f_y y' \quad (7)$$

x, y, z: 3D position in the world coordinate
x', y': image plane at z=1
w: radial distortion coefficient
f_x, f_y: focal lengths of real camera
c_x, c_y: center point of real camera
u, v: 2D position of the point after projection to the camera plane

At (1) and (2), 3D point in the real world is projected to 2D point in image plane at z =1; then this 2D point is distorted. In (6) and (7), distorted point is projected to image plane of the camera. It has been proven[3] that this FOV model produces lower error than well-known model of Brown , with a caveat that each model can have better performance in particular types of cameras.

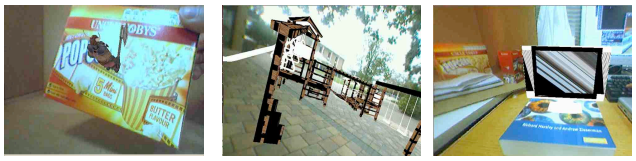
4.2 New Method

In Figure 6, an image of the real world is seen as a distorted video frame. Then, this video image will be undistorted. Undistorted video image and graphics are cut out at center. Finally, processed frame and graphics are blended together. PTAMM's distortion pipe-line (see Figure 5) is quite different from other AR systems. In



(a) (b)

Figure 6: Improved distortion pipe-line.



(a) Planar object (popcorn box) is tracked, then virtual model is displayed on top

(b) Outdoor demo with the virtual playground rigorously registered in the real world

(c) virtual popcorn box is displayed in the real world

Figure 7: Demo Applications.

PTAMM, distorted video image is kept and blended with distorted cut-out graphics. While improved un-distortion pipe-line is widely adapted in AR systems. Indeed, approach of improved un-distortion pipe-line is similar to OpenCV's in spirit. However, in the proposed solution, undistorted map – homographic map – is stored with scale information. Scale information in two dimensions is stored in order to support central cut-out of undistorted video image (see Figure 6 for detail). Consequently, cut-out of rendered graphics in this approach will follow central cut-out with scale information. In fact, central cut-out is also applied in ARToolKit[7].

In order to improve performance, we applied homography un-distortion mapping, which is similar approach in OpenCV. Thus, after calibrating, camera parameters and un-distortion map are stored. This makes un-distortion processing at running time more effective; consequently, it leads to better performance.

5 Results

In this section, we first present three demo applications that we have developed using PTAMM-Plus and TINT. Then, we present a benchmark of our overall system.

5.1 Demo Applications

We have implemented three demo applications: object tracking, table top, and outdoor tracking.

Object Tracking In current implementation, PTAMM is limited to object tracking. In particular, it can only track planar object. This is because 3D map points in PTAMM have no visibility testing; thus, all map points in camera view are projected on image frame when searching for detected fast corner features in current frame.

Table Top In this scenario, 6DOF camera pose tracking ability of PTAMM-Plus has been examined. Table top demo tends to show performance of PTAMM-Plus in small area with sustainable light condition. Furthermore, it is also displayed result of proposed solution (as showed in Figure 7c). Compared to PTAMM solution (as seen in Figure 5b, the proposed solution conveys more meaningful ideas for AR applications.

Outdoor Tracking Despite of lack experiment for outdoor scenario, PTAMM can be considered as highly robust tracking system for outdoor environment. This is showed in the demonstration in (Klein & Murray 2007). For outdoor demonstration, a map has been built; and then it was used instantly in PTAMM-Plus.

5.2 Benchmark

In order to compare performance of PTAMM and PTAMM-Plus, we have developed a benchmark. The platform that we have employed for the benchmark is built with AMD Core2 3.0GHz (CPU), 3GB RAM. Our system environment for the benchmark is Ubuntu.

The method we have used to compare PTAMM and PTAMM-Plus is observing and comparing computational time for the whole system at each frame. We run PTAMM and output computational time (include: tracking estimation, rendering, and UI processing) every frame in a squence. With PTAMM-Plus, we also did similar experiment to export another sequence of computational time. Then we compared these two sequences and came to conclusion with a plotted chart (see Figure 8).

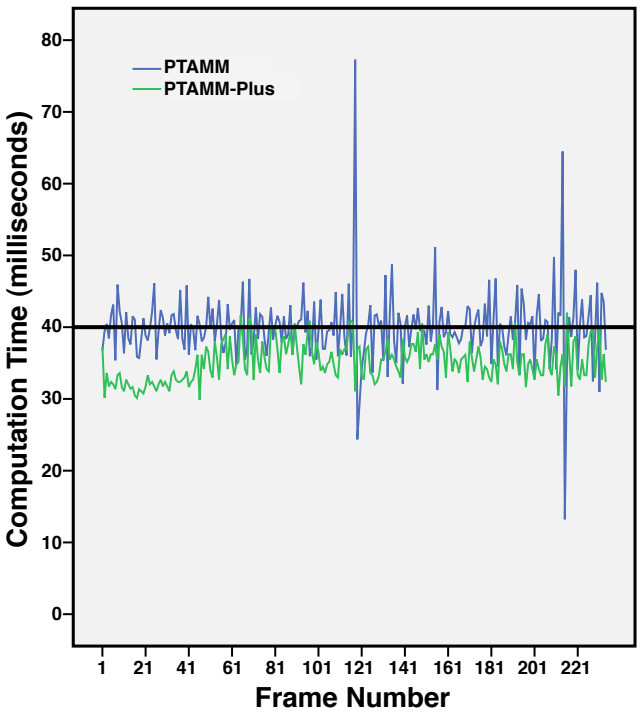


Figure 8: Comparison between PTAMM and PTAMM-Plus (tracking, rendering and UI processing time consume).

As observed results from the comparision, PTAMM-Plus seems to outperform original PTAMM. This is because PTAMM-Plus supports ability to control UI interaction and Tracking in separate threads; while PTAMM highly tights Tracking and UI processing into a single thread.

6 Conclusions

A robust and accurate tracking system is required for successful AR applications. PTAMM allows for building and re-loading dense 3D feature maps, and provides accurate and stable tracking information. Although PTAMM has good potentials, it is not being widely used because of its monolithic architectural nature. Our systematic approach to tackle difficulties of monolithic architecture through exploring parallelization and developing the wrapper enabled mod-

ularization of PTAMM feasible and effective. Presented PTAMM-Plus is not just a wrapper, it is rather a refactorization and extension of PTAMM, which improved accuracy, performance, and adaptability. We integrated PTAMM-Plus with TINT system seamlessly. According to our experiments, PTAMM-Plus provided accurate tracking information with enhanced performance over PTAMM.

We plan to make PTAMM-Plus available to developers and improve usability and adaptability. As a future work, we also plan to develop an automatic, fast, and autonomous initialization.

ACKNOWLEDGEMENTS

The authors wish to thank Rhys Moyne for the implementation of Python Boost wrapper and Arindam Dey for very helpful discussions.

REFERENCES

[1] R. Azuma, B. Hoff, H. Neely, and R. Sarfaty. A motion-stabilized outdoor augmented reality system. In *VR'99: Proceedings of IEEE Virtual Reality*, pages 252–259, 1999.

[2] R. O. Castle and D. W. Murray. Object recognition and localization while tracking and mapping. In *ISMAR '09: Proceedings of the 8th IEEE International Symposium on Mixed and Augmented Reality*, pages 179–180, 2009.

[3] F. Devernay and O. Faugeras. Straight lines have to be straight: automatic calibration and removal of distortion from scenes of structured enviroments. *Mach. Vision Appl.*, 13(1):14–24, 2001.

[4] U. Eck and C. Sandor. Tint: Towards a pure python augmented reality framework. In *Proceedings of the Third Workshop on Software Engineering and Architectures for Realtime Interactive Systems*, pages 41–46, Waltham, MA, USA, March 2010.

[5] E. Foxlin and L. Naimark. Miniaturization, calibration & accuracy evaluation of a hybrid self-tracker. In *ISMAR '03: Proceedings of the 2nd IEEE and ACM International Symposium on Mixed and Augmented Reality*, pages 151–160, 2003.

[6] A. Irschara, C. Zach, J.-M. Frahm, and H. Bischof. From structure-from-motion point clouds to fast location recognition. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 2599–2606, 2009.

[7] H. Kato and M. Billinghurst. Marker tracking and hmd calibration for a video-based augmented reality conferencing system. In *IWAR'99: 2nd IEEE and ACM International Workshop on In Augmented Reality*, pages 85–94, 1999.

[8] G. Klein and D. W. Murray. Parallel tracking and mapping for small AR workspaces. In *ISMAR '07: Proceedings of the 6th IEEE and ACM International Symposium on Mixed and Augmented Reality*, pages 225–234, 2007.

[9] G. Klein and D. W. Murray. Parallel tracking and mapping on a camera phone. In *ISMAR '09: Proceedings of the 8th IEEE International Symposium on Mixed and Augmented Reality*, pages 83–86, 2009.

[10] G. Reitmayr and T. Drummond. Going out: robust model-based tracking for outdoor augmented reality. In *ISMAR '06: Proceedings of the 5th IEEE and ACM International Symposium on Mixed and Augmented Reality*, pages 109–118, 2006.

[11] D. Wagner and D. Schmalstieg. Artoolkitplus for pose tracking on mobile devices. In *CVWW'07: Proceedings of 12th Computer Vision Winter Workshop*, 2007.