CompSci 705 Seminar Final Report - Augmented Reality

Wallace Yuen
University of Auckland
wyue013@aucklanduni.ac.nz

ABSTRACT

Augmented Reality is a vigorously researched area due to its vast scale of applicability in different areas, enabled by continually improved computer hardware and technology. Augmented Reality was explored in many researches for different applications such as navigation systems, medical visualization systems, and computer games. Despite recent advances in computer hardware, Augmented Reality still remains a challenge with its performance due to the complexity of its algorithms involved, from disciplines such as Computer Vision, Computer Graphics, and Artificial Intelligence, which is a problem due to the limited resources available. Existing researches focused on improving the quality of virtual content registration geometrically and photometrically, while others focused on improving the performance for real-world applications such as mobile Augmented Reality. Interaction technique is also a highly popular area with many researches trying to investigate ways to interact and manipulate virtual contents to enhance the user experience and practicality of Augmented Reality applications. This paper analyzes the problems in Augmented Reality and discusses the current approaches and methodologies in overcoming these problems to bring Augmented Reality to practical use.

General Terms

HCI; Augmented Reality;

INTRODUCTION

With recent technological advances, Augmented Reality has become a popular research topic in Computer Science due to the capability of low-cost hardware in enabling Augmented Reality applications to run with interactive performance. Augmented Reality is a subcategory of Mixed Reality which merges the real and virtual contents together. As shown in Figure 1, Augmented Reality is based around the real environment, and adds virtual contents on top of the real-world layer [7]. Virtual information based on the real content can be supplied to users of Augmented Reality to enhance their perception of the real world and support them in completing specific tasks.

Augmented Reality mixes real-world contents with virtual contents and aligns them seamlessly so that the virtual contents are placed in the correct location and orientation to provide accurate information to the users. Real-time performance is required due to it largely being used as an interactive application, with some form of tracking such as motion or image detector. These requirements define some main problems in the area of Augmented Reality. For example, correct alignment of virtual and real content requires high precision detection and tracking algorithm from the Computer Vision discipline if a vision-based approach is used. However, these algorithms tend to be computationally and memory intensive, which would make it difficult for Augmented Reality applications to run in interactive rates and satisfy hardware constraints.

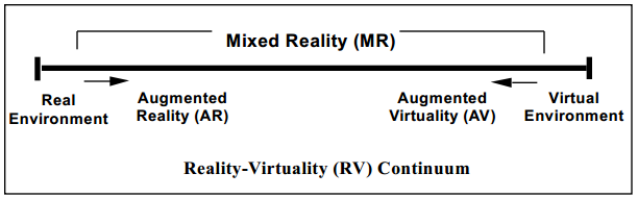

Figure 1. Reality-Virtual (RV) Continuum, extracted from Milgram et al. [7]

Mobile devices have experienced exponential growth in recent years, and Augmented Reality can be integrated into these devices due to the capability of its hardware and the availability of different types of sensors that these devices possess, such as multi-touch sensor, accelerometer, GPS sensor, light sensor, and proximity sensor [10]. However, computational and memory resources availability on these devices are still limited in comparison to traditional PCs, thus more optimization techniques have to be applied to improve the underlying algorithms. However, complexity reduction often results in reduction in accuracy, which would affect the accuracy in tracking and the overall effectiveness of the application. It is critically important to optimize between the performance and the accuracy of the underlying algorithms, which is a core problem for many Augmented Reality researches.

In the rest of the paper, the Augmented Reality section introduces some of the fundamental concepts in the area, then problems in Augmented Reality are identified and are discussed in following subsections. Approaches to overcome these problems in existing reaches are then discussed and compared against others for their effectiveness in handling these problems. Finally, summary and future work are described at the end of the paper.

AUGMENTED REALITY

Augmented Reality requires the process of detecting and tracking real-world objects in order to embed virtual contents into the real environment. Many types of sensors are avail-

CHI'12, May 5–10, 2012, Austin, Texas, USA.

able for real-world object and motion detection, such as the magnetic, ultrasonic, acoustic, inertial, optical, and mechanical sensors [13]. These sensors have different purposes such as motion sensor (mechanical, inertial sensors), while some others can sense 3D depth information (magnetic, ultrasonic), to acquire different types of real-world information. An example of a common device that uses multiple motion sensors is the iPhone, which uses the low-cost inertial sensors: accelerometer and gyrometer, for detecting rotational and linear motion relative to the device.

Another type of widely used sensor in Augmented Reality is vision-based sensor. Vision-based sensors are available in many devices such as webcams, laptops, and mobile devices, usually in the form of a low-cost camera for capturing real-world images.

After the detection of real-world environment or objects, specific objects' position and orientation are estimated for virtual content alignment, which is known as pose estimation. At the render stage, Augmented Reality renders the virtual content onto the real-world environment seamlessly, by referring to the results estimated in the pose estimation step.

PROBLEMS

There are still many unsolved problems in Augmented Reality, with registration being the most researched and influential area [13]. This is because registration is regarded as an important enabling technology for Augmented Reality, thus improving it would also improve the overall performance of Augmented Reality applications [13]. Registration can be divided into the categories of geometry and photometry, with the prior being concerned mainly with pose estimation, and the latter being concerned with the rendering quality of virtual contents. Performance and hardware limitation, especially in the case of mobile devices, are also popular problems for investigation due to their rising capabilities in supporting Augmented Reality. Lastly, interaction techniques and user interfaces are important problems due to the new types of interaction between physical and virtual world created by Augmented Reality, which could improve the design of Augmented Reality applications [13].

Geometric Registration

Geometric registration is concerned with the correctness of registering virtual content with the real environment. Hence, object detection and tracking have to be highly precise to estimate the position and orientation of real objects. Traditionally, Augmented Reality systems measured the position and orientation of a real-world object using magnetic or ultrasonic sensors [2], but due to their inaccuracy and limitation in tracking volumes, a vision-based marker approach is generally preferred [9].

Figure 2 shows an example of a vision-based marker that can be used to detect an object. An example of the actual matrix code is shown on the left, which can be attached to the surface of any object for its position and orientation to be recognized by the Augmented Reality system. The marker is

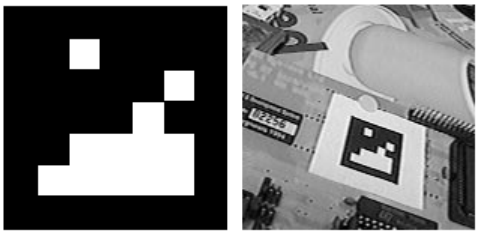

Figure 2. Marker-based pose estimation, actual marker (left) and marker used to identify piece of paper's position (right), extracted from Rekimoto et al. [9]

detected using its feature points by the camera, and the underlying system estimates the position and orientation of the camera relative to the marker [9].

The use of markers is inconvenient and impractical as they are not always available for users in different locations. To avoid the use of markers, many markerless approaches have been proposed to use primitive geometry features such as corners and segments for pose estimation [12, 6]. These features are naturally extracted from the captured images, which are then used to predict the position and orientation of the camera and the objects in the scene. However, both approaches are subjected to the presence of noise in the process of data acquisition, such as occlusion, motion blur, or reflection in a captured image. These problems were solved by using temporal information from previous frames to accurately predict real-world objects location regardless of natural noises [12]. Despite largely relying on the accuracy of previously captured images, such approach provides much improved robustness to the system in terms of continuous target tracking after the initial detection phase.

Performance & Hardware Limitation

The performance of Augmented Reality applications is important and require real-time user interaction. Object detection is a complicated task and is computationally intensive, making it difficult to achieve real-time performance given the complexity involved in object detection. Therefore, existing techniques have to be optimized both in terms of memory and efficiency for real-time applications.

The performance issue is particularly important for mobile devices due to their limited computational power and memory availability. Mobile devices have generally lower computational throughput than PCs, along with very limited memory bandwidth and storage [12]. This is often solved by optimizing computationally intensive tasks such as object detection, to reduce the complexity of the algorithm, through the use of simpler corner detection techniques with parallel and GPU programming [4, 6, 12]. As a supplement to this, Wagner et al.[12] delegated the tracking task to a more light-weight tracking algorithm known as the PatchTracker after the initial detection of object features. Alternatively, other approaches such as precomputation can also be adopted to allow devices to simply refer to precomputed values from a look-up table instead of computing the actual values in run-time to reduce the amount of computation required [4].

Photometric Registration

Photorealistic visualization ensures the rendering quality of virtual contents in Augmented Reality is simulated closely to the real environment. Aside from the geometric registration problem, the illumination and reflectance of virtual objects have to be the same as the objects in the real-world to achieve seamless integration. The shading of virtual objects remains to be a difficult task due to a wide variety of material reflectance behaviors and scattering effects, coupled with the difficulty in simulating real-world lighting effects. Global illumination addresses the photorealistic problem in Computer Graphics, but is not suitable for real-time applications due to the simulation of multiple ray bounces, and remains to be an open problem in Computer Graphics.

The location of different light sources has to be acquired to shade virtual objects using shading algorithms. Furthermore, the shadowing of virtual objects also has to be estimated, so that the shadows projected from the virtual objects are similar to the shadows projected by the real objects. These effects remain to be highly difficult to achieve, due to the difficulty in extracting light sources from the captured real-world image. Several attempts were made to solve this problem, by capturing the location of different light sources using external devices such as a perfectly specular orb to reflect incoming light sources [5], or a separate light sensor that faces up to capture the light sources [1], while some systems simply use a separate virtual light source for shadow projection with less realism [8]. However, these approaches are often computationally intensive, which is often infeasible for Augmented Reality applications due to most of the computational resources have already been delegated to the effort of object detection and tracking.

APPROACHES

Markerless Tracking for Mobile Augmented Reality

Wagner et al. [12] proposed to modify the original SIFT and Ferns classification methods by using a corner detector called FAST detector for features detection, to reduce the computation requirements in the feature detection test by using primitive natural features. Because the original approach (Difference of Gaussian) for features detection works in the scale-space to provide scalability robustness in detection, therefore they had to manually compile features in various different scale of the captured image. Aside from that, they also changed the pose estimation technique of the two methods to use the Gauss-Newton iteration to refine the detected pose.

They also proposed to use either of these algorithms in conjunction with an external tracking system called PatchTracker for interleaving the tracking task after the initial detection phase. Using the PatchTracker allowed them to increase the accuracy of tracking and avoid the complex computation in the detection algorithm, so that it can maintain the tracking of objects with more robustness efficiently [12]. Occlusion also happens often in object tracking, as features can be occluded in later frames after the initial detection. However, by using the PatchTracker, it enables them to use temporal information to predict the location of object in next successive frames, so that occluded regions would not affect the tracking of object [12].

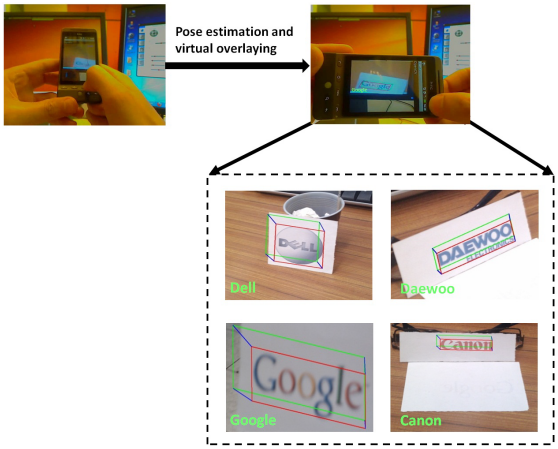

Figure 3. Robust corner recovery for marker recognition, extracted from Maidi et al. [6]

By adopting the FAST detection algorithm, Wagner et al. [12] managed to improve the performance of object detection [12]. The SIFT and Fern Classification algorithms were also modified, so that the parameters are optimized for lower memory consumption and computational complexity, which increases the performance of object matching and classification for detection. Furthermore, the PatchTracker uses previous frames as its only reference data for continuous tracking after detection. Hence, it generally performs faster than using detection algorithms for tracking, by avoiding the calculation of high complexity in pose estimation and object matching on a per-frame basis [12].

Rather than purely tracking and detecting real-world objects, Maidi et al. [6] proposed a new method of pose estimation algorithm, and presented a way of using it to merge virtual objects seamlessly with real-world surfaces. Pose estimation is highly important to Augmented Reality, due to the fact that the virtual content has to be aligned seamlessly to the real environment, but computing it accurately and efficiently remains to be one of the most challenging problems in Computer Vision and Augmented Reality. This paper focused on their new pose estimation algorithm that combines the Extended Kalman Filter algorithm with an analytical algorithm, for mapping 2D points relative to the image coordinates on the image plane to 3D points in the world coordinates of a detected real-world object in world space. They proposed to use an analytical algorithm to first compute the initial parameters to guess the pose, and then iterate using the Extended Kalman Filter to converge it to a more accurate pose [6].

With the increase of accuracy of pose estimation algorithm, virtual objects can be mapped more accurately onto the real object with minimal pixel difference. The papers vision-based algorithm enables them to accurately detect real-world objects position and orientation, so that they can set the position and orientation of virtual objects accordingly. The result of using this algorithm is clearly shown in Figure 3, where the orientation of several rendered cubes are clearly positioned

on top of the detected logo with the same orientation. Using this method, the paper successfully augmented real content with virtual content, where the position and orientation of both parts match seamlessly together.

Maidi et al. [6] proposed several other features to improve the performance of Augmented Reality algorithms on mobile platforms. Rather than using the SIFT and Ferns methods used by Wagner et al. [12], they adopted another feature detection technique - the SURF algorithm, and optimized it for the use on mobile devices. The authors used the FAST corner detector to speed up feature points detection [6], and they also proposed their way of implementing the SURF detector along with some code optimization techniques and parallel programming, which allowed them to improve the performance of their algorithms so that they can be used on mobile devices at interactive rate [6]. They also used k-d tree for searching in image database upon reference image retrieval, which reduces the search time by performing fast approximate nearest neighborhood searches [6].

Scalable Recognition and Tracking

Ha et al. [4] addressed an important issue in Augmented Reality. Most existing researches were done on improving existing vision-based algorithms such as feature detection and object classification for practical use in Augmented Reality. However, the scalability of these vision-based approaches remains to be a largely unexplored area in the field. Scalability recognition is important in Augmented Reality due to the presence of a high number of objects surrounding us, thus there is always a need to search for matching objects in a large sized database.

This paper [4] presented an approach to improve the scalability of existing techniques, so that matching through a large database can be done in Augmented Reality applications on mobile devices. This is done by delegating the feature detection task to a remote server. In essence, the remote server retrieves a frame uploaded by the mobile device, and it detects and constructs natural feature descriptors using the SIFT algorithm on its GPU and matches against the images stored in the database [4]. Furthermore, a vocabulary tree is used by the system to reduce the search space required for each object in the scene, by narrowing down the potential matching objects and eliminating non-matching objects during the recognition phase on the server. Therefore, using a remote server, it enables a reduction of overall time required to perform the recognition task for large sized database [4].

Ha et al. [4] managed to improve the speed of the algorithm to achieve scalable recognition by modifying the algorithm similar to the papers [6, 12]. They implemented the SIFT algorithm on the GPU for the server, parallelized the algorithm to improve the efficiency, and further improved the performance with the use of precomputed look-up tables for the calculation of orientation and magnitude of SIFT descriptors [4].

Physically-based Augmented Reality

Piumsomboon et al. [8] tackled the problem of physical interaction with virtual object. They suggested that the behavior

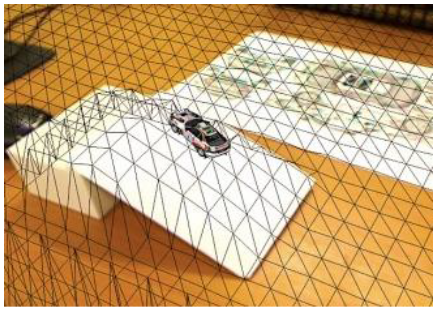

Figure 4. Physically simulated virtual object, extracted from Piumsomboon et al. [8]

of virtual objects without physical simulation may not meet users' expectation. They designed an Augmented Reality framework to provide physically-based virtual object interaction coupled with real environment awareness to enhance user interaction experience [8], and to provide a new way of interaction using Augmented Reality technology.

Contrary to many existing approaches that simply renders virtual objects in the same position and alignment as real objects [4, 6, 12], the physical simulation of virtual objects requires 3D pose estimation instead of 2D, so that the action and behavior of virtual objects can be simulated like a real object. They achieved 3D pose estimation by using the depth information acquired using Microsoft Kinect, which captures the spatial information using its embedded depth sensor [8].

The framework that they proposed uses markers for detection. The Kinect first detects the marker and extracts the captured information of color and depth to construct the homography matrix that transforms between the coordinate systems of the camera and the marker. The homography matrix is then used with the OpenSceneGraph library to reconstruct the camera and marker in a 3D space, which would then be used to reconstruct a 3D surface mesh for the physical simulation of virtual objects. In essence, an actual scene of the captured image is actually setup using the OpenSceneGraph library, but renders only the virtual object in each simulation step to merge with the real environment.

Outdoor Augmented Reality on Mobile Phone

Takacs et al. [11] developed an augmented reality system for mobile devices in outdoor environment, to enable object detection through image matching against a large database of location-tagged images. The system relies on the mobile device's GPS system and matches its captured image against images that are in the same location in the database, and provides the user with information and links to the recognized objects in the captured image.

The system uses a robust feature matching algorithm - an optimized version of the SURF algorithm for mobile devices, with improved efficiency and memory consumption in exchange for reduced matching accuracy. Other optimization methods are also employed, such as the distribution of computation amongst different users in the same location, and the caching of location-tagged images from the database to avoid

Figure 5. Occlusion in a captured image (left) and the top matching image from the database (right), extracted from Takacs et al. [11]

incessant retrieval of images from the database for features matching in the same location.

Their system is comprised of two parts: the mobile device, and the remote server. The remote server is responsible for storing the uploaded location-tagged images, and grouping them by location, known as a loxel. The server extracts features from the images in a loxel and finds the meaning feature descriptors to be used for matching. These feature descriptors are compressed and stored in the database for mobile devices to retrieve and match the features in their captured images against. The actual matching is done on the mobile device, which matches the features from the captured image against the database of image features in the same location. Kd-tree is used to allow fast approximate nearest neighbor (ANN) matching, and the matching result is determined by the ranking based on the number of matched features. Figure 5 shows that in a natural environment, the frequency of noise in captured images is high due to reasons such as occlusion by trees, cars, or different camera poses. Furthermore, different times of the day can produce very different images due to the difference of object reflectance in the scene.

This approach is generally more robust to noise such as occlusion due to the nature of ranking and the use of loxels to filter out impossible locations to avoid matching errors. This is illustrated in the top row of Figure 5, with the building correctly recognized despite its distinctive coloring compared to the matched image, while also being occluded by different colored cars. The bottom row also displays successful recognition with the correct building recognized despite being occluded by a tree.

TranslatAR

Fragoso et al. [3] developed a mobile augmented reality that translates texts in captured images to a specific language. The user has to trigger the translation process by tapping on the a word, and then the system detects and extracts the word for translation. The entire process is executed on the mobile device apart from the actual translation of texts between languages.

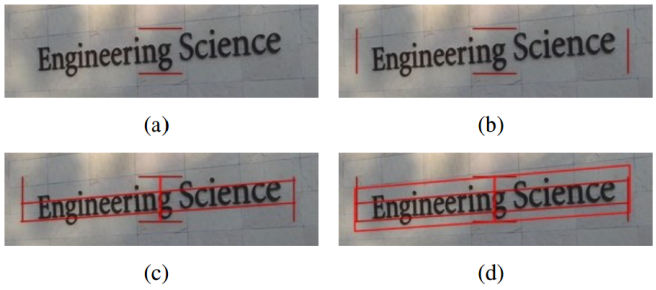

Figure 6. Stages of bounding box construction around the user tapped word, extracted from Fragoso et al. [3]

In order to detect the word, the system constructs a bounding box around the word that is tapped by the user and applies Hough Transformation to correctly position and align the bounding box around the word [3], as shown in Figure 6. Furthermore, the detected bounding box is then transformed into a rectangular shape, to rectify any distortion and transformation applied to the text in the captured image. This system aims to deliver properly rendered text that fits the real-world environment, thus it has to extract the font and background color to simulate the real environment. This is done by using K-Means sampling, by simply sampling pixel colors. They estimate the background and the font color with the assumption that the background color is uniform, as in the case with the font color, therefore the number of pixels with the background color would outnumber the number of pixels with the font color. By doing it, it enables them to achieve more realistic photometric registration in integrating the virtual texts into the environment.

The system achieves extraction and recognition of letters through the use of an existing library called the Optical Character Recognition (OCR) system, and simply uploads the extracted string to Google Translate for the translation task. Lastly, they also integrated a tracking system for tracking temporally to keep the virtual contents rendered after detection.

Methodology

Evaluations of the framework were committed by Wagner et al. [12] to compare the accuracy of detection of different algorithms proposed in their framework. The modified SIFT and Ferns were compared against the original SIFT And Ferns for testing the matching rates with five images in different transformations such as scale, rotation, tilt, and brightness. Fragoso et al. [3] also evaluated the accuracy of their Augmented Reality translator, and monitored the failure for different components, i.e. detector, color estimator, OCR, and Google translator.

Wagner et al. [12] also performed a target tracking test to compare the tracking robustness of the modified algorithms with and without the adoption of the PatchTracker. The robustness of the four methods was tested with different scenarios such as occlusion, camera tilting, fast camera movement,

and target removal [12]. Similarly, Maidi et al. [6] performed a robustness evaluation on its modified SURF algorithm by transforming the target's position and rotation, and Ha et al. [4] also performed a test with regards to rotation and occlusion of the scene, they also took scalability into consideration with a large database made up of 10,000 items.

A performance evaluation was also committed by multiple papers [4, 12] to demonstrate the framework's suitability for mobile devices. Other papers [3, 6] simply recorded the milliseconds per frame to demonstrate their feasibility. On the other hand, Takacs et al. [11] performed tests for memory usage and bandwidth consumption to test the performance of loxel (images in a specific location) retrieval from a remote database.

Findings

The matching rates test committed by Wagner et al. [12] showed an increasing trend of accuracy with the use of outlier removal for both modified SIFT and Ferns algorithms. The accuracy of these two approaches is relatively close to the matching accuracy of the original SIFT and Ferns algorithms. The test for target tracking showed that all algorithms (with or without PatchTracker) worked fairly well in the simple sequence without any occlusion or noise. The use of PatchTracker proved to be more useful for cases such as occlusion, camera tilting, and fast camera movement, achieving 100% tracking for all three cases, while algorithms without the use of PatchTracker suffered from large pixel errors due to their inability to continually detect feature points from the scene [12]. The use of PatchTracker also coped better when the target is partially removed from the scene, as the stand-alone approaches suffer majorly from a loss of features in the scene to maintain the tracking of target object [12]. The tests performed by Ha et al. [4] and Maidi et al. [6] showed good robustness for scaling and rotation of objects, but the approach of Ha et al. [4] suffered with the problem of occlusion, particularly in cases where the tracking of object was lost and the application had to reinitialize the entire detection phase.

Fragoso et al. [3] tested the accuracy of text translation using a set of 30 video clips of signs in outdoor scenes and recorded the percentage of failure of individual components that made up the application. Their results showed that the external library - OCR, had the highest percentage of failure for character recognition, with 32.9% failure rate out of the total failure rate of 54.4%, thus the application requires improvement in order for it to be put to practical use.

The performance test showed that the use of PatchTracker improved the performance of the algorithms significantly on both mobile device and PC. Without the use of PatchTracker, the modified algorithms only managed to achieve an average of 15fps, which is only just acceptable for real-time performance with obvious delays in on-screen rendering. Furthermore, the mobile device that the algorithms were tested on was a high-end device, thus they would be much slower on devices used by the general population [12]. Contrary to Wagner et al's findings [12], Ha et al. [4] showed that the scalability problem has a huge impact on the performance of recognition in remote server, and causes huge latency every time an initialization step occurs between the mobile device and the remote server for re-recognition.

A more detailed analysis was provided by Wagner et al. [12] to show that the feature description and matching process takes the majority of the time in executing the modified SIFT algorithm, similar to the results showed by Ha et al. [4], where the server generally takes 50% of the total time per initialization frame for SIFT detection and recognition. Ha et al's approach generally takes longer in recognition for detection in each initialization step due to the size of the database, thus the overall initialization time is also generally longer. However, re-initialization only occurs if the mobile device loses its tracking, thus it is generally not a concern for the overall performance of the application [4, 12]. Similar results were obtained by Fragoso et al. [3], with the initial steps of text location rectification and character recognition taking the majority of the processing time, per frame tracking was shown to mitigate the amount of processing in each frame. With the use of a tracking approach like PatchTracker, the computational requirements involved are lowered by using a more light-weight tracking algorithm [4, 12], while [4, 11] both showed network latency in executing detection on a remote server slows down the entire detection process.

Piumsomboon et al. [8] tested the run-time performance of the applications developed using their proposed framework on a desktop computer, achieving a frame rate of over 25fps. They also implemented two applications and observed that many participants were able to learn how to use the applications quickly, due to the expected behavior of virtual objects. However, these applications generally have very little rendering workload, which means the majority computations are performed for virtual objects simulation. Physically-based simulation is often a computation and memory intensive task, and in this case the performance of the actual applications appeared to be quite low, thus such approach is still not appropriate for many hardware-limited devices such as mobile phones.

SUMMARY

This paper has introduced the topic of Augmented Reality, and has identified few basic problems in Augmented Reality that has been actively researched in the past. The main problems that were discussed consist of geometric registration, photometric registration, and performance and hardware limitation related to Augmented Reality.

Various approaches for geometric registration were discussed to overcome the problem of occlusion, which caused problem to properly register the virtual object with the real environment. On the other hand, some researches focused on developing new algorithms for correct geometric registration, while maintaining good performance, which is particularly important for mobile Augmented Reality. Performance and hardware limitation are important due to the requirement of real-time performance for Augmented Reality, and is critical for devices with limited hardware specifications such as mobile devices, therefore many researches aimed to tackle this area by optimizing existing algorithms and delegating parts

of Computer Vision-based algorithms to a remote server to reduce the weight of Augmented Reality.

On the other hand photometric registration is also important to properly align the shading and shadowing of virtual objects with the scene, so that they are merged seamlessly. Existing researches focused on methods of gathering location information of light sources in a scene, and some of these approaches produced very convincing shading and shadowing effects. However, most Augmented Reality systems require most computational resources to be used for object detection and tracking, thus realistic virtual content has not been put to practical use in real applications. However, applications such as Augmented Reality translator [3] requires the virtual content to blend into the background and font color of the actual scene, thus simple approaches such as estimating their corresponding colors were done to register the virtual content seamlessly.

FUTURE WORK

Some of the problems like geometric registration and performance and hardware limitation are fundamental problems related to the enabling technologies of Augmented Reality, thus they have been the focal point in research. There are still a lot of work to be done in these fields to improve the efficiency of the basic processes in Augmented Reality to allow more allocation of resources to the actual Augmented Reality application logic. Other areas such as photometric registration and interaction techniques in Augmented Reality are still relatively new, and will be continually researched in the future to bring more realism and better interaction to Augmented Reality applications.

Particularly with photometric registration, there are lack of techniques available that can achieve real-time performance in Augmented Reality applications. Many existing techniques proposed are non-robust approaches and require a lot of preparation for capturing the actual scene before using it for the application, and techniques used in actual applications often simply use the colors captured from real world scenes rather than a physically-based lighting approach in shading virtual objects. This is also the case for interaction techniques, where there are lack of researches to investigate the actual interaction with virtual objects, and interaction techniques such as the incorporation physical-based modeling affects the performance of Augmented Reality severely, and requires improvement to be used on hardware-limited devices.

REFERENCES

1. Agusanto, K., Li, L., Chuangui, Z., and Sing, N. Photorealistic rendering for augmented reality using environment illumination. In *Mixed and Augmented Reality, 2003. Proceedings. The Second IEEE and ACM International Symposium on*, IEEE (2003), 208–216.
2. Bajura, M., and Neumann, U. Dynamic registration correction in augmented-reality systems. In *Virtual Reality Annual International Symposium, 1995. Proceedings.*, IEEE (1995), 189–196.
3. Fragoso, V., Gauglitz, S., Zamora, S., Kleban, J., and Turk, M. Translatar: A mobile augmented reality translator. In *Applications of Computer Vision (WACV), 2011 IEEE Workshop on*, IEEE (2011), 497–502.
4. Ha, J., Cho, K., Rojas, F., and Yang, H. Real-time scalable recognition and tracking based on the server-client model for mobile augmented reality. In *VR Innovation (ISVRI), 2011 IEEE International Symposium on*, IEEE (2011), 267–272.
5. Kanbara, M., and Yokoya, N. Real-time estimation of light source environment for photorealistic augmented reality. In *Proceedings of the 17th International Conference on Pattern Recognition*, vol. 2 (2004), 911–914.
6. Maidi, M., Preda, M., and Le, V. Markerless tracking for mobile augmented reality. In *Signal and Image Processing Applications (ICSIPA), 2011 IEEE International Conference on*, IEEE (2011), 301–306.
7. Milgram, P., Takemura, H., Utsumi, A., and Kishino, F. Augmented reality: A class of displays on the reality-virtuality continuum. In *Proceedings of Telemanipulator and Telepresence Technologies*, vol. 2351, Citeseer (1994), 282–292.
8. Piumsomboon, T., Clark, A., and Billinghurst, M. Physically-based interaction for tabletop augmented reality using a depth-sensing camera for environment mapping. In *Proceedings of the 26th International Conference on Image and Vision Computing New Zealand* (2011).
9. Rekimoto, J. Matrix: A realtime object identification and registration method for augmented reality. In *Computer Human Interaction, 1998. Proceedings. 3rd Asia Pacific*, IEEE (1998), 63–68.
10. Rösler, A. *Augmented Reality Games on the iPhone*. PhD thesis, Thesis,(BA). Blekinge Institute of Technology, 2009.
11. Takacs, G., Chandrasekhar, V., Gelfand, N., Xiong, Y., Chen, W., Bismpigiannis, T., Grzeszczuk, R., Pulli, K., and Girod, B. Outdoors augmented reality on mobile phone using loxel-based visual feature organization. In *Proceeding of the 1st ACM international conference on Multimedia information retrieval*, ACM (2008), 427–434.
12. Wagner, D., Reitmayr, G., Mulloni, A., Drummond, T., and Schmalstieg, D. Real-time detection and tracking for augmented reality on mobile phones. *Visualization and Computer Graphics, IEEE Transactions on 16*, 3 (2010), 355–368.
13. Zhou, F., Duh, H., and Billinghurst, M. Trends in augmented reality tracking, interaction and display: A review of ten years of ismar. In *Mixed and Augmented Reality, 2008. ISMAR 2008. 7th IEEE/ACM International Symposium on*, Ieee (2008), 193–202.