Digilog Miniature: Real-time, Immersive, and Interactive AR on Miniatures

Kiyoung Kim* Youngmin Park† Woontack Woo‡

GIST U-VR Lab.

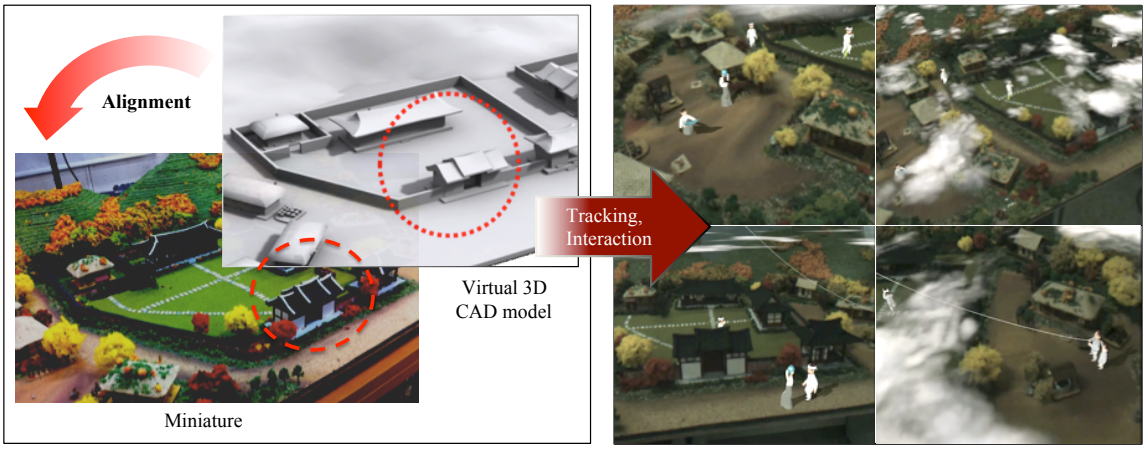

Figure 1: *Making a miniature alive! We present the framework of building a real-time, immersive, and interactive AR system on miniatures.*

Abstract

In this paper, we propose a framework to realize a real-time, immersive, and interactive Augmented Reality running on a miniature. We achieve these challenging goals by harmonizing unknown tracking data generation, structure-supported marker-less tracking, and user-event handling within VR authoring tool. We cluster unknown 3D keypoints by using selective keyframe-based 3D reconstruction for real-time parallel detection by tracking. Afterwards, we align 3D keypoints and CAD models by using user-assisted iterative fitting method to resolve occlusions and shadows between real and virtual objects. At last, we retrieve active/passive commands by interpreting events from users' speech, ray-pointing interface, and camera distance to interact with the augmented contents. We developed the *digilog miniature* system to show validity of the proposed framework. According to experimental results, the system worked about 20 frames per second: 7.82 ms for tracking, 38.35 ms for contents rendering. The proposed framework can be applied to storytelling, manufacturing simulation, urban planning and so on.

CR Categories: H.5.1 [Information Interfaces and Presentation]: Multimedia Information Systems—Artificial, Augmented, and Virtual Realities; I.4.8 [Image Processing and Computer Vision]: Scene Analysis—Shape

Keywords: augmented reality, digilog miniature, 3D tracking

*e-mail: kkim@gist.ac.kr
†e-mail: ypark@gist.ac.kr
‡e-mail: wwoo@gist.ac.kr

VRCAI 2010, Seoul, South Korea, December 12 – 13, 2010.
© 2010 ACM 978-1-4503-0459-7/10/0012 $10.00

1 Introduction

Recently, Augmented Reality (AR) have been applied to various traditional systems. For example, in *digilog books* [Ha et al. 2010; Kim et al. 2010], users can experience the relevant 3D digital contents augmented on book pages while they are reading texts. In addition, the users can manipulate the augmented scene by interacting with virtual characters or real objects. As a result, *digilog book* overcame monotonous storytelling of the traditional systems by applying AR technologies. In this paper, we want to discuss how to make a hundred of miniatures interactive and reusable with AR technologies like *digilog book* because most of miniatures are left alone after installation in museums or art galleries.

Most interactive AR systems have been using distinctive planar markers [Kato and Billinghurst 1999; Zhou et al. 2008] for 3D contents augmentation. The attached markers enable robust tracking over challenging target objects. However, the working space of these systems is limited on the plane, on the markers exactly. Moreover, they distract users' sights so that the immersion to the augmented contents may decrease. To realize more natural and immersive systems, 3D marker-less tracking is an appropriate method. However, 3D marker-less tracking-based systems have rarely been developed because of the lack of practical real-time tracking methods including the instability in unknown 3D structure acquisition, the difficulties of virtual 3D model alignment, and the absence of proper interaction methods. Furthermore, there have been no frameworks to facilitate the required processes.

Our contribution, in this paper, is to overcome the difficulties in 3D marker-less tracking-based systems systematically. We propose a framework that facilitates unknown tracking data generation, structure-supported marker-less tracking and on-line user-event handling to accomplish the goals. Firstly, we cluster unknown 3D keypoints by using selective keyframe-based 3D reconstruction for real-time parallel detection by tracking. Afterwards, we align 3D keypoints and CAD models by using user-assisted iterative fitting method to resolve occlusions and shadows between real and virtual objects. Then, we retrieve local and global commands by interpreting events from users' speech or ray-pointing interface to

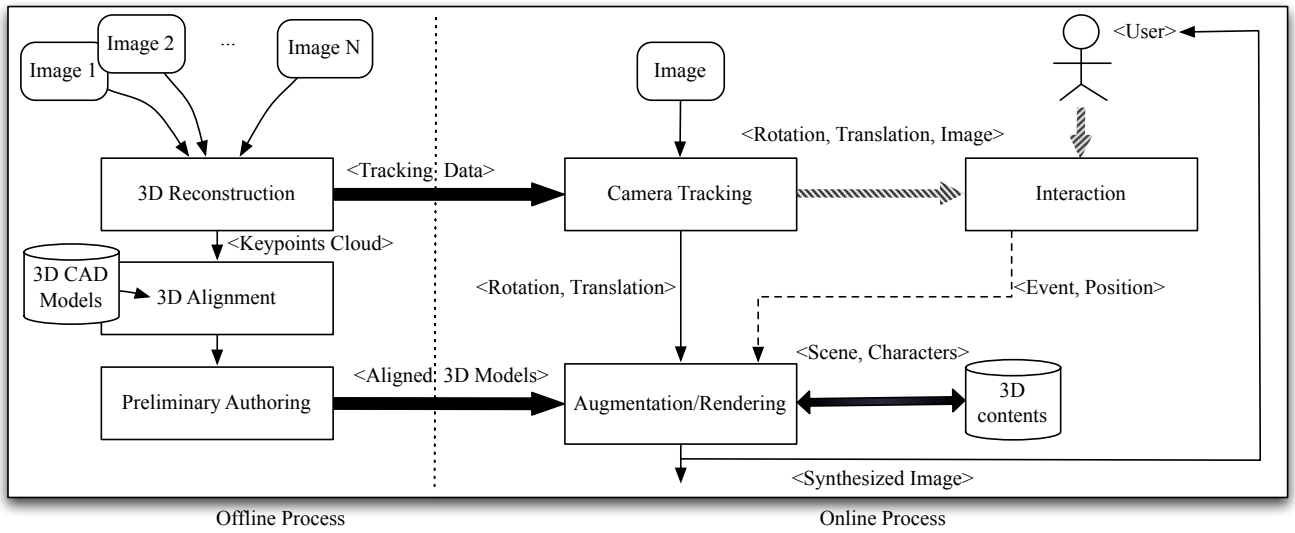

Figure 2: *Overall flowchart of the proposed framework that consists of the off-line and online parts. In the off-line process, we prepare tracking data of the target miniature followed by the alignment of 3D CAD models. In the online (real-time) process, we augment all 3D contents according to the users' input commands to the system. Details are found in Section 3.*

interact with the augmented scene or characters.

We develop the *digilog miniature* system, whose concept is borrowed from *digilog book*, using the proposed framework. Simply, the difference is in metaphors for augmentation; shapes of miniatures (3D targets) versus figures or texts in books (planar targets). We provide the final AR view through a hand-held or a large displaying devices. In our implementation, AR rendering and contents authoring are integrated within the VR authoring tool[1] so that the proposed system inherits the properties of it. The users can make their own application and redistribute it easier. Any miniatures having reasonable textures can be tracked in real-time so that they can be easily used as a part of users' final system.

This paper is organized as follows. We introduce the further related works about interactive AR systems in Section 2. Then, the proposed framework and its implementation notes are explained in Section 3 and Section 4, respectively. Experimental results with our *digilog miniature* system are shown in Section 5. Finally, we conclude the paper in Section 6.

2 Related Work

In this section, we explain the related works of existing interactive AR systems, especially, according to metaphors for augmentation. And also, we consider the fundamental technologies, such as modeling and tracking, that are used to construct interactive AR systems.

2.1 Metaphors and Systems

One of the best planar metaphors except markers is a "book". *digilog book* or *magic book* [Billinghurst et al. 2001] are representative applications. J. Pilet et al. [2006] proposed the all-in-one solution to solve the difficulties to make marker-less AR system using planar objects. The system supports the training of tracking data, the illumination parameters estimation, and the multi-camera tracking.

[1] http://www.virtools.com

In case of non-planar objects, 3D structure information is mandatory. "cube-type tangible objects", where different markers are attached to each side, have been used in many interactive storytelling systems [Zhou et al. 2004; Zhou et al. 2008] because computing coordinates is uncomplicated. To solve this unknown structure problem, as an earlier work, I. Skrypnyk and D.G Lowe [2004] proposed the method to use an image-based 3D reconstruction and SIFT [Lowe 2004] feature-based tracking. A "cup" was used as an example. However, the tracking speed was not that enough to use in real-time applications. Recently, Qi Pan et al. [2009] proposed the system to model the 3D objects for real-time tracking. However, most natural objects have been exploited to show just the results of tracking algorithms.

We choose the miniature as the metaphor of the framework. The miniatures are tractable and customizable 3D objects and it has its own story; users can imagine what the miniature is representing in real environment. From the viewpoint of the procedure, our approach is more similar with the work [Pilet et al. 2006]. However, we consider more challenging 3D cases. We support complex and large-size miniatures via 3D reconstruction like the work [Lowe 2004]. Our tracking time was less than 10 ms without attaching any markers.

2.2 Further Modeling and Tracking

Recently, PTAM-based approaches [Klein and Murray 2007; Castle and Murray 2009], alternative SLAM, were proposed for robust tracking in unknown environments. They used keyframe-based tracking. The relocalization is done by searching the most similar keyframe in the low resolution. In our tracking, it is done by keypoint-based matching running on the background thread. Hybrid approaches, such as merging visual cues [Park et al. 2010] and using multi-threads [Lee and Hollerer 2008], to provide better performance were also proposed. Our approach also exploits multi-threads to guarantee the real-time performance.

About modeling, Van den Hengel et al. [2009] integrated VideoTrace [Van den Hengel et al. 2007] and PTAM. Initial keyframes and camera poses were obtained using PTAM. The modeling was

performed with VideoTrace interface. High-quality models can be obtained. A real-time modeling system was also proposed [Simon 2010]. They showed that the objects with planar surfaces were modeled on the fly by moving a camera.

3 Proposed Framework

3.1 Overview

The overall flowchart of the proposed framework is illustrated in Figure 2. The framework consists of an off-line and an on-line processes. In the off-line stage, we mainly prepare essential data for the target miniature tracking. The corresponding 3D CAD models of the miniature are aligned in 3D so as to handle virtual-real contents occlusions and shadows. In on-line stage, we synthesize the camera image with the pre-authored 3D contents by computed camera pose-based registration in real-time. And also we provide the minimum interaction interface that manipulates the scene on the fly. We explain each component in the following sections.

3.2 Unknown Tracking Data Generation

Our problem in tracking the target miniature is that the essential feature information, such as feature descriptors and 3D coordinates, for camera pose estimation is unknown. We acquire these information from the several images or the video stream taken around the target miniature. We apply SIFT features and multi-view image-based 3D reconstruction to track the unknown target miniature. Firstly, we need to select keyframes from the given images. It is more efficient to use keyframes instead of using all images. We then do feature matching between keyframes to know keypoints correspondences. The first two keyframes are used to build the initial structure. We update the structure step by step with the remaining keyframes. Finally, sparse bundle adjustment [Lourakis and Argyros 2009] is applied to get optimal structures and motions.

In case of large miniatures, it is difficult to gather 3D data at once. Especially, it often fails when we use still images. Thus, in our framework, we support video stream so that users can achieve a good reconstruction result with the minimal effort. A key idea is to extract keyframes based on the following criteria. We firstly examine the baseline between images so that too much close images are ignored. The stability of the image is determined by the number of inliers and reprojection error.

By doing this, we obtain 3D positions of keypoints for each keyframe. As a result, our data structure for 3D tracking includes {kd-trees, 3D positions, keyframe images}. kd-tree is constructed per keyframe. Note that this process needs not to be done in online. We keep all data as a file.

3.3 3D CAD Model Alignment

So far, we recover the structure of the miniature up to 3D point. However, we need higher level structure elements to improve the realism of AR applications. The high-level elements, such as meshes, enable handling occlusions between real and virtual entities and rendering virtual shadows onto real entities. Thus, in our framework, we propose to register 3D CAD model of the target miniature by aligning two local coordinates systems of the virtual model $\mathcal{V}$ and the miniature $\mathcal{M}$. We assume that the 3D CAD model is prepared in advance. This is done by finding 3D transformation $T_{m,v} = \begin{pmatrix} R & t \\ 0 & 1 \end{pmatrix}$ scale factor λ in Eq. 1. R and t are rotation and translation matrices.

$$\mathcal{V} = \lambda T_{m,v} \mathcal{M} \tag{1}$$

Basically, we manually and repeatedly obtain $T_{v,m}$ and scale factor λ because there exist differences in both the miniature and the CAD model though they are made sophisticatedly. We firstly compute $R = \begin{pmatrix} r_1 & r_2 & r_3 \end{pmatrix}$ where r_k is kth 3×1 column vector. For that, we need to select N (more than 3) keypoints $p_{1 \cdots N}$ on the same plane in a certain keyframe. Then, we compute the plane equation to find potential coordinates. RANSAC-based [Fischler and Bolles 1981] approach is applied to find parameters (a, b, c, d) that minimize

$$\sum_{i=1}^{N} d(p_i), d(p_i) = \frac{|ap_{k,x} + bp_{k,y} + cp_{k,z} + d|}{\sqrt{a^2 + b^2 + c^2}} \tag{2}$$

Usually, we normalize the obtained parameters ($\sqrt{a^2 + b^2 + c^2} = 1$) so that we can assume that the normal vector $(a, b, c)^T$ is a potential r_3. Then, two keypoints are required to find potential r_1 and r_2. Any vector v formed from two keypoints of the plane, i.e, $v = p_i - p_j (i \neq j)$ can be chosen as $r_1 = \frac{v}{\|v\|}$ after normalization. Then, the last r_2 is determined by $r_3 \times r_1$. Next, we set the initial $t = \frac{\sum_{i=1}^{N} p_i}{N}$ as a center of the user-selected keypoints.

At last, we find λ using a known measurement of the structures. We measure the physical distance d_m of the miniature in mm unit. And find the corresponding distance d_v in the keyframe in $pixel$ unit. Then, we obtain $\lambda = \frac{d_v}{d_m}$, a ratio between two different units. The overall procedure is summarized in Algorithm 1.

Input: User-selected keypoints $p_{1 \cdots N}$
Output: scale factor λ, 3D transformation $T_{m,v}$

while *Visually check the registration with* λ_{iter}, T_{iter} **do**

- Select keypoints $p_{1 \cdots N}$ on a plane ;
- $(a, b, c, d)^T$ = plane fitting that minimizes Eq. 2 ;
- $r_3 \leftarrow$ normalization of $(a, b, c)^T$;
- $r_1 \leftarrow$ normalization of $v = p_i - p_j (i \neq j)$;
- $r_2 = r_3 \times r_1$;
- $R_{iter} = \begin{pmatrix} r_1 & r_2 & r_3 \end{pmatrix}, t_{iter} = \frac{\sum_{i=1}^{N} p_i}{N}$
- Measure physical distance of the miniature d_m ;
- Compute the corresponding distance of the CAD model d_v ;
- $\lambda_{iter} = \frac{d_v}{d_m}$;
- iter++ ;

end

Algorithm 1: Pseudo-code of the proposed iterative fitting method to align 3D CAD model and selected keypoints.

3.4 Camera Tracking

In this paper, we extend the algorithm exploited in [Kim et al. 2010] based on camera tracking data prepared in 3.2. In [Kim et al. 2010], the detection and the feature tracking run in parallel on each thread for real-time performance. We adjust this algorithm to our miniature case. In case of small miniatures having less than 50 keyframes, it is enough to use just kd-tree in the detection part. In particular, it is very fast. However, in case of large or multiple miniatures, kd-tree is not a good choice because it requires amount

of memory and yields bad detection results. Thus, two-step approach that uses vocabulary tree [Nister and Stewenius 2006] followed by kd-tree is proposed for scalability. Note that this is slower than forehand approach, but it still provides real-time performance. In our framework, we adaptively use both according to the number of keyframes.

Once the detection is done, we track the features and compute the camera pose. Again, RANSAC-based approach is used right after detection. We select 5 keypoints randomly and use P-n-P algorithm [F.Moreno-Noguer et al. 2007] to recover camera pose represented by rotation and translation matrices $R\ t$, respectively. Relocalization is resolved because the detection process is always running on the background. At the end, the camera pose is refined by applying *Levenberg-Marquardt* non-linear optimization scheme to Eq. 3 with currently tracked points. Note that we set the maximum iteration number to 10 for real-time performance in practice.

$$\underset{R,t}{\operatorname{argmin}} \sum_{x \in tracked\_points} \|x - K \begin{pmatrix} R & t \end{pmatrix} X\|, \qquad (3)$$

where x are tracked 2D keypoints. X are corresponding 3D keypoints and K is a camera intrinsic matrix.

3.5 Miniature Interaction

We support a simple interaction interface to manipulate the augmented contents. Two types of active interactions, speech and ray-based pointing interfaces, and camera distant-based passive interaction are supported. The usage of each interface is explained.

Speech is an effective method to control the global parameters. For example, it is used to change the current scene or the status of the scene. Whenever a user speaks a specific word, the word is transfered and recognized in the speech interpreter. Then, it is converted to the scene parameters to trigger the corresponding event. We use Microsoft's Speech Application Programming Interface (SAPI) to recognize the speech. In practice, we need to test and select the command words carefully to guarantee the reasonable recognition rate.

On the other hand, the ray-based pointing method is an effective method to control the local parameters within the scene. At first, a ray is defined by 2D point resulted from users' screen touch. Thus, once the 2D position is determined, it is converted to the 3D position in the miniature coordinates system. Then, the interpreter retrieves relevant events to the 3D position. For example, the selection of the characters or the location pointing for character movements are possible.

Camera distant-based passive interaction is also considered to play sound effects. The Euclidean distance between moving camera and virtual characters or scene is computed while the tracker is working. When the distance is less or more than defined thresholds, the values are sent to the interpreter and then the commands are generated according to the distance and the type of the closest virtual entities. The procedure is shown in Figure 3.

4 Implementation Notes

In this section, we explain how the proposed runtime modules are effectively integrated. Especially, we used Virtools, VR editor to separate the authoring and rendering from tracking. For that, we developed Virtools Adapter which connects the modules implemented in C/C++ to BuildingBlocks, Plug-in interface of Virtools. Following sections explain the implementation details.

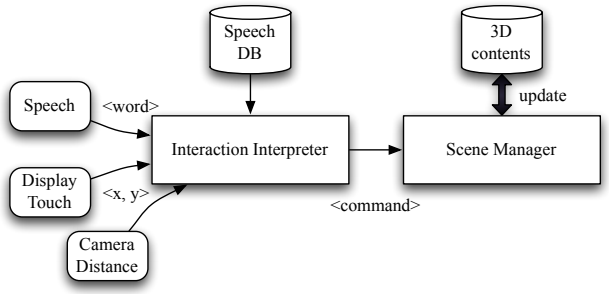

Figure 3: *Event handling procedure*

4.1 Camera and Tracker Design

We implemented 4 BuildingBlocks for tracking and camera interface as follows.

- *Track initializer*: Pre-load information such as 3D CAD models, SIFT features, and camera intrinsic parameters in the memory and perform every pre-processing before running the system. This also calculate the projection matrix for visually correct registration. Note that this module is performed only once in the beginning of the system because we do not consider runtime deformation, addition, or deletion of the objects.

- *Camera grabber*: Retrieve a frame from the camera and set the image as the pixel buffer of a texture unit in Virtools. The image information is shared with another BuildingBlock which performs tracking in a coherent way; they share only the texture ID, and Virtools manages related background operations.

- *Tracker*: Detect and track the pose of the miniature with respect to the camera as described in Section 3.4. This module gets only the camera frame as a texture ID and does not output the estimated pose directly for extensibility.

- *Transformation connector*: Register the virtual entities and the current camera image. This relates the pose estimated in the tracker and the specific virtual entity in Virtools. More specifically, we fix the 3D model corresponding to the miniature in a specific position in the virtual world, and adjust the pose the virtual camera in Virtools to be the inverse of the estimated pose(the pose of the miniature with respect to the camera). In this way, the contents authoring is unrestricted by the tracker.

Among the 4 BuildingBlocks, the latter 3 Building Blocks are performed iteratively and updates the pose of the virtual camera continuously.

4.2 AR Scene Rendering

Basically, AR scene is constructed by overlaying foreground 3D contents on the background camera image. Thus, in actual implementation, the alignment of virtual and real cameras is required to make AR scene. This is done in *Track initializer* mentioned in Section 4.1. The alignment problem is related to the field of view (FOV) and the aspect ratio. The camera in virtual space can be set using $gluPerspective$ OpenGL function. The parameters of this function are obtained from a real camera intrinsic parameters and its image size. The parameters are computed as follows.

$$\mathbf{FOV} = 2.0 \times \arctan\left(\frac{imageheight}{2.0 \times f_x}\right) \times \frac{180.0}{\pi} \quad (4)$$

$$\mathbf{AspectRatio} = \frac{f_y \times imagewidth}{f_x \times imageheight}, \quad (5)$$

where (f_x, f_y) are camera focal length.

5 Experiments

We realized the proposed framework on a usual modern PC. We developed the *digilog miniature* system based on the framework. A FleaMV [Point Grey Research 2010] camera was exploited to provide 640×480 resolution images to the system. A NVidia GTX 285 graphic card was used to accelerate 3D tracking. Overall hardware was configured as shown in Figure 4. The hand-held display has a touch-screen sensor so that we can find out 2D positions mentioned in Section 3.5.

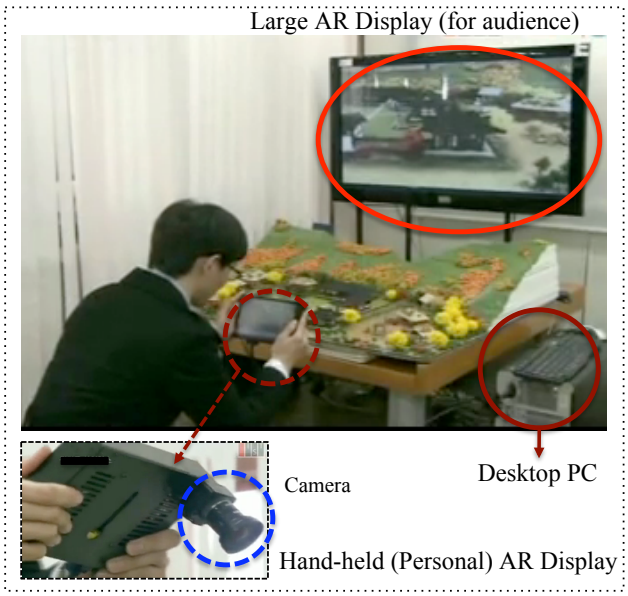

Figure 4: *The implemented system view*

We used OpenCV [Open Computer Vision Library 2010] to implement the algorithm mentioned in Section 3. GPU-based SIFT [Wu 2007] was used to speed up the feature extraction. Table 1 shows the summary of the hardwares and softwares used in implementing our system.

Table 1: *Hardwares, softwares and dependencies used in digilog miniature system.*

	Component	Specification
Hardwares	CPU	Quad-core 2.94 GHz
	Graphic Card	NVidia GTX 285
	Hand-held AR Display	7 inch
	FleaMV Camera	4 mm lens, 60 FPS
	Miniature	$120 \times 90 \times 20\,cm$
Softwares	3D Modeler 3D Tracker Virtools Adapter Stand alone AR viewer	OpenCV, GPUSIFT, Virtools SDK

5.1 *Digilog Miniature* Storytelling Scenario

We considered the Korean traditional story, "Hong Gildong", simply Korean Robinhood. The details of the story is found at this site [2]. From the whole story, we selected four important scenes and authored the 3D contents using the proposed system. Around 15 moving-characters were made to organize the scenes. Figure 5 shows the snapshots of the proposed storytelling application. Figure 5 (a) shows the start of the system. Figure 5 (b)-(i) shows snapshots extracted from each scene. The scene is changed by users's speeches, such as 'forward' or 'backward'. And if the camera approaches the characters, then we can hear what the characters are talking.

5.2 Performance

We measured the overall performance of the system. Figure 6 shows the average CPU possession rate of each process for one frame. The most bottleneck of the system was not the tracking part, but 3D contents rendering part. Note that "camera frame rendering" and "tracking" time was not much varied. If we consider only tracking part, the frame rate is over 100 FPS. However, "3D contents rendering" time varied according to the complexity of the scene. In spite of heavy loads in 3D contents rendering (in our case about 103 MB including character animations, narrations, and so on), we achieved real-time performance around 20 FPS, which is usually enough for interactive AR systems.

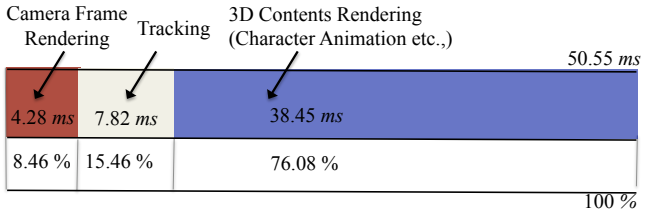

Figure 6: *The average time measurement (CPU possession rate) of the system viewer side to process one frame.*

Figure 7 shows the process of the tracking data generation and the 3D CAD model alignment mentioned in Section 3. Figure 7 (a) shows the 6 images of the selected keyframes. We extracted total 64 keyframes in this case. Figure 7 (b) is the visualization of the reconstructed 3D points and the computed camera poses for each keyframe. Figure 7 (c) shows the snapshots during the 3D CAD models alignment task. We augmented the grid on the ground in order to assist user's manual adjustment.

Figure 8 shows the results of occlusions and shadows. The geometry of building structures is transparently rendered so that the character behind the wall is partially occluded. Shadows of characters are rendered on the ground as well.

6 Conclusions

We presented the framework for realizing interactive marker-less AR systems on miniatures. We introduced three key technologies: the multi-view image-based 3D reconstruction for real-time tracking, the 3D alignment between keypoint coloud and known geometric virtual models for occlusions and shadows, and the event-based active/passive interaction methods for manipulating the augmented scene. In the experiments, we developed *digilog miniature* system

[2] http://en.wikipedia.org/wiki/Hong_Gildong

Figure 8: *Virtual shadows on the ground and the occlusions between real structure and virtual characters.*

that integrates marker-less AR with storytelling based on our framework. And it successfully worked at real-time. As future works, we will consider lighting issues to enhance the realism. And the dense reconstruction approach will be applied to render more detailed shadows. The system can be integrated to various applications, such as storytelling, edutainment, urban planning, manufacturing simulation, and so on.

Acknowledgements

This research is supported by Ministry of Culture, Sports and Tourism (MCST) and Korea Creative Content Agency (KOCCA), under the Culture Technology(CT) Research & Development Program 2010.

References

BILLINGHURST, M., KATO, H., AND POUPYREV, I. 2001. The magicbook - moving seamlessly between reality and virtuality. *Computer Graphics and Applications, IEEE 21*, 3 (May), 6 – 8.

CASTLE, R. O., AND MURRAY, D. W. 2009. Object Recognition and Localization while Tracking and Mapping. In *IEEE/ACM International Symposium on Mixed and Augmented Reality*.

FISCHLER, M. A., AND BOLLES, R. C. 1981. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM 24*, 6, 381–395.

F.MORENO-NOGUER, V.LEPETIT, AND P.FUA. 2007. Accurate non-iterative o(n) solution to the pnp problem. In *IEEE International Conference on Computer Vision*.

HA, T., LEE, Y., AND WOO, W. 2010. Digilog book for temple bell tolling experience based on interactive augmented reality. *Virtual Reality*, 1–15. 10.1007/s10055-010-0164-8.

KATO, H., AND BILLINGHURST, M. 1999. Marker tracking and HMD calibration for a video-based augmented reality conferencing system. In *IEEE/ACM International Workshop on Augmented Reality*, 85 – 94.

KIM, K., LEPETIT, V., AND WOO, W. 2010. Scalable real-time planar targets tracking for digilog books. *The Visual Computer 26*, 1145–1154. 10.1007/s00371-010-0490-6.

KLEIN, G., AND MURRAY, D. 2007. Parallel Tracking and Mapping for Small AR Workspaces. In *IEEE/ACM International Symposium on Mixed and Augmented Reality*.

LEE, T., AND HOLLERER, T. 2008. Hybrid feature tracking and user interaction for markerless augmented reality. *Virtual Reality Conference, 2008. VR '08. IEEE* (Feb), 145 – 152.

LOURAKIS, M. A., AND ARGYROS, A. 2009. SBA: A Software Package for Generic Sparse Bundle Adjustment. *ACM Trans. Math. Software 36*, 1, 1–30.

LOWE, D. G. 2004. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision 60*, 2 (Nov), 91–110.

NISTER, D., AND STEWENIUS, H. 2006. Scalable recognition with a vocabulary tree. In *IEEE Conference on Computer Vision and Pattern Recognition*.

OPEN COMPUTER VISION LIBRARY. 2010. . *http://sourceforge.net/projects/ opencvlibrary/*.

PAN, Q., REITMAYR, G., AND DRUMMOND, T. 2009. Interactive Model Reconstruction with User Guidance. In *IEEE/ACM International Symposium on Mixed and Augmented Reality*.

PARK, H., OH, J., SEO, B., AND PARK, J. 2010. Automatic confidence adjustment of visual cues in modelbased camera tracking. *Computer Animation and Virtual Worlds* (Jan).

PILET, J., GEIGER, A., LAGGER, P., LEPETIT, V., AND FUA, P. 2006. An all-in-one solution to geometric and photometric calibration. In *IEEE/ACM International Symposium on Mixed and Augmented Reality*, 69 – 78.

POINT GREY RESEARCH. 2010. FleaMV. *http://www.ptgrey.com/*.

SIMON, G. 2010. In-Situ 3D Sketching Using a Video Camera as an Interaction and Tracking Device. In *Eurographics*.

SKRYPNYK, I., AND LOWE, D. 2004. Scene Modelling, Recognition and Tracking with Invariant Image Features. In *IEEE/ACM International Symposium on Mixed and Augmented Reality*.

VAN DEN HENGEL, A., DICK, A., THORMÄHLEN, T., WARD, B., AND TORR, P. H. S. 2007. VideoTrace: Rapid Interactive Scene Modelling from Video. In *SIGGRAPH*.

VAN DEN HENGEL, A., HILL, R., WARD, B., AND DICK, A. 2009. In Situ Image-Based Modeling. In *IEEE/ACM International Symposium on Mixed and Augmented Reality*.

WU, C., 2007. SiftGPU: A GPU Implementation of Scale Invariant Feature Transform (SIFT).

ZHOU, Z., CHEOK, A., PAN, J., AND LI, Y. 2004. Magic story cube: an interactive tangible interface for storytelling. *ACM SIGCHI International Conference on Advances in computer entertainment technology* (Jan).

ZHOU, Z., CHEOK, A., TEDJOKUSUMO, J., AND OMER, G. 2008. wizqubestm–a novel tangible interface for interactive storytelling in mixed reality. *International Journal of Virtual Reality* (Jan).

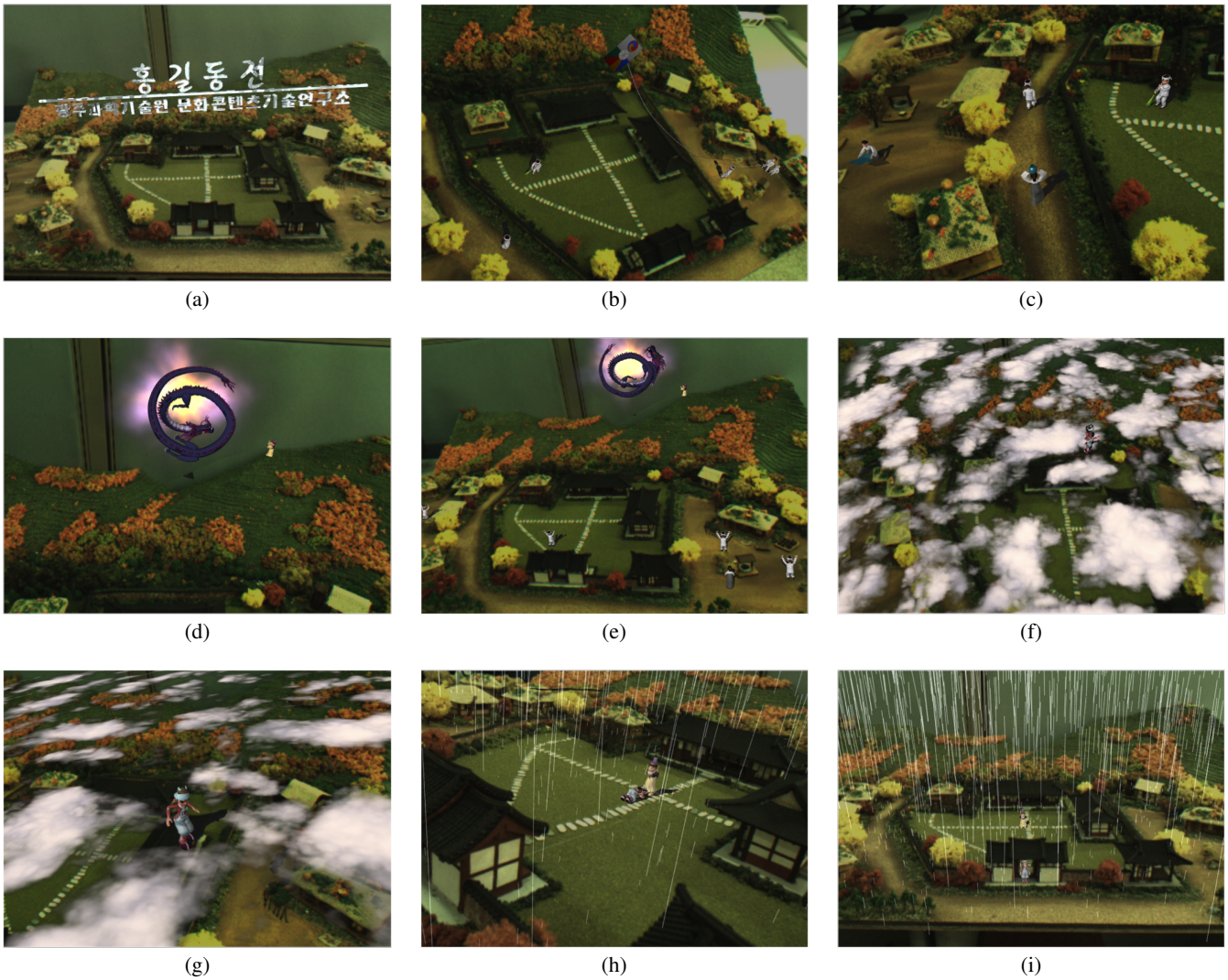

Figure 5: *Snapshots of digilog miniature: We implemented the Korean traditional story with four special scenes. Each scene has its own story. (a) Introduction, (b)-(c) Scene I, (d)-(e) Scene II, (f)-(g) Scene III, (h)-(i) Scene IV.*

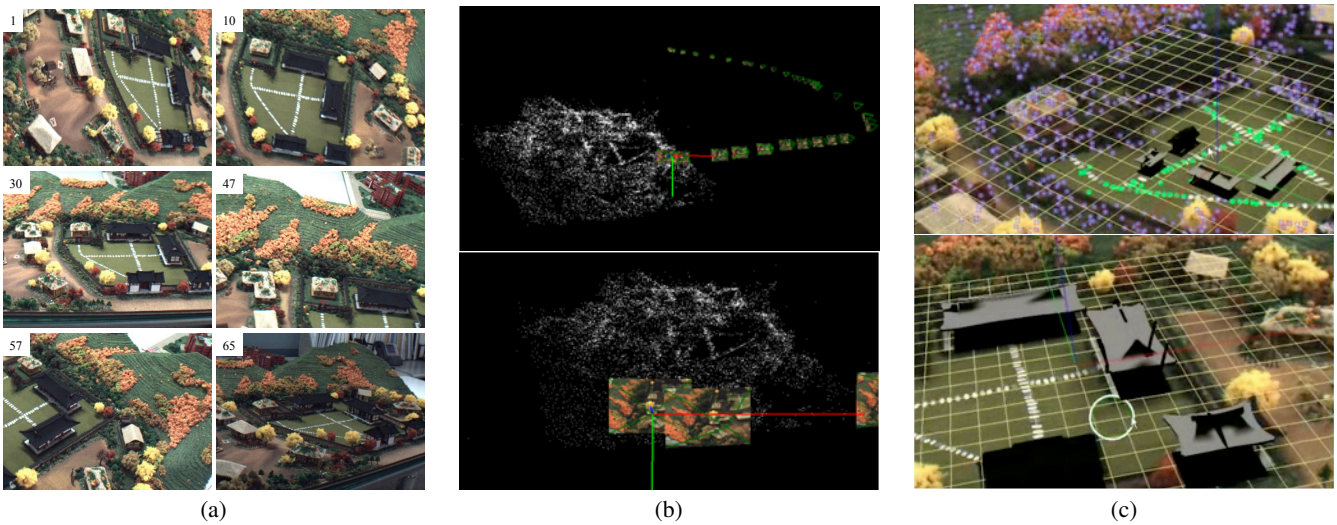

Figure 7: *Results of tracking data generation and 3D model alignment process, (a) Six selected keyframes for multi-view image-based 3D reconstruction, (b) Keypoint cloud and the recovered camera poses at each keyframe, (c) Snapshot of 3D model alignment and its UI.*