[POSTER] Rubix: Dynamic Spatial Augmented Reality by Extraction of Plane Regions with a RGB-D Camera

Masayuki Sano* Keio University

Kazuki Matsumoto† Keio University

Bruce H. Thomas‡ University of South Australia

Hideo Saito§ Keio University

Abstract

Dynamic spatial augmented reality requires accurate real-time 3D pose information of the physical objects that are to be projected onto. Previous depth-based methods for tracking objects required strong features to enable recognition; making it difficult to estimate an accurate 6DOF pose for physical objects with a small set of recognizable features (such as a non-textured cube). We propose a more accurate method with fewer limitations for the pose estimation of a tangible object that has known planar faces and using depth data from an RGB-D camera only. In this paper, the physical object's shape is limited to cubes of different sizes. We apply this new tracking method to achieve dynamic projections onto these cubes. In our method, 3D points from an RGB-D camera are divided into a cluster of planar regions, and the point cloud inside each face of the object is fitted to an already-known geometric model of a cube. With the 6DOF pose of the physical object, SAR generated imagery is then projected correctly onto the physical object. The 6DOF tracking is designed to support tangible interactions with the physical object. We implemented example interactive applications with one or multiple cubes to show the capability of our method.

Index Terms: Spatial Augmented Reality, Six Degree of Freedom Tracking, RGB-D Camera

1 Introduction

Spatial Augmented Reality (SAR) allows for the illumination of a physical object with perspectively-correct computer generated imagery. A driving functionality of SAR is to change the visual appearance of physical objects, and this functionality may be applied in a diverse range of applications such as entertainment and product prototyping. Dynamic SAR focuses on the particular problem of moving projected target objects and providing real-time registration of the augmentations. This form of SAR requires real-time pose estimation of the objects so that they can be projected onto.

Different technologies are available to perform 6DOF object tracking, ranging from dedicated infrastructure (for example the OptiTrack and Polhemus systems) to natural feature tracking with a RGB camera. These current tracking technologies require either 1) the attachment of a physical device on the physical object that affects the appearance of tracked objects or 2) the use of an RGB camera that requires a clear view of the texture on target objects. This paper presents a depth-based tracking system that does not depend on attached sensors [1], and also overcomes the problem of obscuring textures from the projected light required for dynamic SAR [2]. Current investigations into the use of RGB-D cameras has overcome some of these problems, but current solutions either only work in a static manner, or require a physical object with a large number of physical features that can be recognized with the depth information.

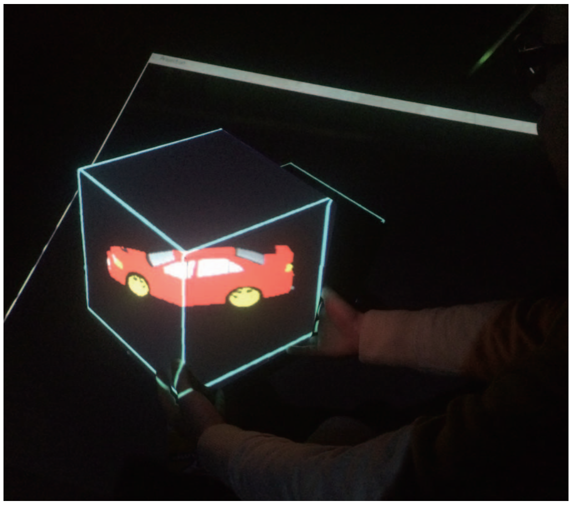

Figure 1: Our new Rubix system

We present Rubix, a 6DOF tracking system that solely employs the depth information from a RGB-D camera, which is coupled with a dynamic SAR rendering system to drape the physical object with computer-generated imagery. Our first implementation of Rubix is applied to cubic-shaped objects of different sizes, see Figure 1. The system operates on a pre-defined object shape, a cube. The system provides 6DOF tracking by the pose estimation based on the construction of planes from the object's shape. The defined 3D object from the plane segmentation phase is then applied to our new 6DOF tracking algorithm. Finally, a set of rendering methods is provided to produce a complete SAR system.

The goals of our new Rubix system are as follows:

1. To improve accuracy of depth based tracking for physical objects illuminated with dynamic SAR. The projection of the SAR information on the physical object should not affect performance of the tracking system.
2. To provide correct viewpoint rendering for users interacting with dynamic SAR. In particular, support will be provided for head motion parallax and correct perspective rendering regardless of the position of the projection surface. The effect is to enable a user to believe there is a 3D computer generated graphical object inside a physical cube.
3. To support real-time tangible interaction with the physical objects. The user can pick up the object and change their viewpoint. In addition, the system will support a range of interaction control mechanisms to support command entry.
4. To provide real-time illumination for dynamic SAR. The system should be accurate and responsive to enable the presentation of dynamic SAR information.

*e-mail: sano@hvrl.ics.keio.ac.jp
†e-mail: kazuki@hvrl.ics.keio.ac.jp
‡e-mail: bruce.thomas@unisa.edu.au
§e-mail: saito@hvrl.ics.keio.ac.jp

978-1-4673-7660-0/15 $31.00 © 2015 IEEE
DOI 10.1109/ISMAR.2015.43

IEEE computer society

2 RELATED WORKS

We were inspired by the pCubee [3] device to support the illusion of a 3D virtual object within a handheld container. The pCubee provides a very intuitive interface for viewing and interacting with 3D graphical content in a tangible manner. Six LCD panels are combined into a cubic shape, with the user and device 6DOF tracked to enable the graphics to be rendered in a manner that gives the user the correct perspective view of an object. We wished to extend this concept by removing a number of the limitations in the current pCubee device, such as: the device is cabled; the complex technology makes the device fragile; the heavy weight; and the fixed size and thick bevels of each display.Tang et al. [4] proposed a extending method for dynamic projection onto a fixed cube, achieving projection based cubic display. With projected imagery on each side of the cube, the user feels as if there is a virtual space containing a 3D graphical object inside the cube. The researchers employed a Kinect RGB-D camera to obtain depth maps of the cube, and detected flat surfaces with an identical gradient value and direction to form three faces of the cube. The user's head is tracked so that virtual space in the projection image is rendered correctly and undistorted from a user's viewpoint. However, the projected cube must remain in a fixed position. This approach is therefore not suitable to support hand-held and tangible interactions.

Previous investigations have employed RGB-D cameras for use with SAR applications. Kobayashi et al. [5] employed 3D point clouds from a RGB-D camera for registering SAR information onto a physical object. These perform global matching between Fast Point Feature Histograms (FPFH) of the projected target's model and the FPFH of the 3D points detected, and then local matching by the Iterative Closest Point (ICP) algorithm. Sticky projections [6] uses the ICP for incremental tracking to estimate the pose of the object. This ICP produced precise localization of the 3D point clouds. This transformation is to align source point cloud to target point cloud, and is calculated by using the nearest points as the corresponding points. However, when the ICP is applied to objects with a low number of shape features, such as a plane or cube, the pose estimation is not accurate and stable due to difficulty obtaining the correct corresponding points. These methods are therefore not suitable for a dynamic SAR user experience with physical objects such as planes and cubes. In our paper, we employ the planar surface segmentation and its matching for estimation of the pose of the object, so objects with planar surfaces such as cube can be robustly tracked for realizing SAR.

3 PROPOSED METHOD

Our proposed method is divided into the following four main steps: 1) plane segmentation, 2) extraction of cube regions, 3) pose estimation, and 4) creation of projection image. Figure 2 provides an overview of the entire process.

3.1 Plane Segmentation

Our plane segmentation algorithm is a modification of the one presented by Matsumoto et al. [7]. The Matsumoto algorithm employed spatial, depth, color and normal information. We modified the algorithm to remove the color factor. The first reason is when projection-based AR is performed in a darkened room, the RGB-D camera cannot guarantee accurate color information. Secondly, projected light changes the appearance of the targeted objects. Finally, segmentation based on color information is not possible with a plain white cube. Our plane segmentation is performed in the same sequence as the Matsumoto algorithm. The use of color information is employed in two instances in the Matsumoto algorithm. Our two modifications are described in this subsection.

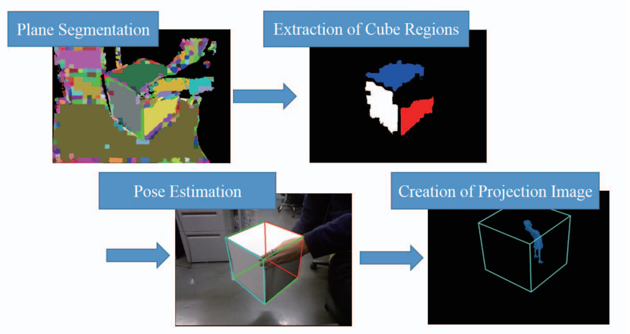

Figure 2: Overview of proposed method

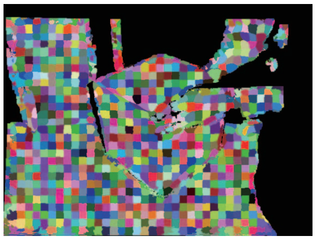

Figure 3: Superpixel segmentation

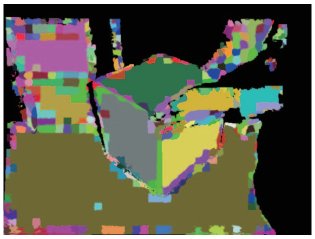

Figure 4: Merged planar regions

Firstly, we modified the Joint Bilateral Filter as follows:

$$D_{f_p} = \frac{\sum_{q\in\Omega} g_s(p-q)g_d(D_p - D_q)D_q}{\sum_{q\in\Omega} g_s(p-q)g_d(D_p - D_q)}. \tag{1}$$

The color information C_p, C_q and g_c has been removed from the equation of Joint Bilateral Filter in Matsumoto algorithm. We finish the pre-processing with this modified Joint Bilateral Filter and the production of a normal map.

Secondly, we apply the modified superpixel segmentation to divide the points into a cluster of small planar pieces as described in the following:

$$D_k(p) = \frac{w_s D_{k_s}(p) + w_d D_{k_d}(p) + w_n D_{k_n}(p)}{w_s + w_d + w_n}. \tag{2}$$

The spatial, depth and normal terms are represented by s, d and n. The weights of spatial, depth and normal (respectively w_s, w_d and w_n) are empirically defined. This modification is equal with the removal of color factors: w_c and $D_{k_c}(p)$. In this process, a number of smaller regions with plane equation are produced as in Figure 3. We therefore merge these areas by employing the graph component labeling algorithm to obtain the global planar structure.

We recognize planar areas on the basis of two conditions. The first condition is that an area and its neighbor's area are geometrically regarded as the same planar region according to their plane equation and the position of their centers. The second condition is that the variance of the normal vectors inside the area must be small, due to the fact that all the normal vectors in a plane must have the same direction. If both conditions are fulfilled, these two areas are merged into the same planar region. Figure 4 depicts that the curved surfaces are not merged and are defined by a cluster of smaller planar regions, and only the planar regions are correctly merged. The figure shows that occlusions, such as users hands, can be discriminated from the cube through this process. This capacity is critical, as the Rubix system is designed to support tangible interactions.

3.2 Extraction of Planar Regions

The second step is to extract the cubic sides from the many planar regions. Frame-to-frame matching is executed between planar regions, but the sides are extracted in an initialization phase in the first frame on the assumption that three sides must be captured from the camera. As an initialization, we define three regions as cubic sides if the following are true: 1) the three normal vectors are perpendicular to each other and 2) the cubic center is assumed by the approximation of the intersection point of three lines defined from the center of the each region and the region's normal vector. After the initial process, we search the cubic sides by using the extracted cubic regions and the cubic pose in the previous frame. Firstly, we correspond the extracted cubic regions in the previous frame with the planar regions in the current frame by calculating the distance $D_k(j)$ (see Equation 2) with the size of the region. In this case, j represents the cubic regions extracted in the previous frame and k represents the planar regions in the current frame.

There are anticipated failure cases that are explicitly checked. The first failure case is when three sides of the cube in frame k are captured from two sides in frame j, and the cubic sides cannot be corresponded. This case is classified as a matching failure, and the cubic pose is assumed from the previous frame. We calculate the correspondence and extract cubic sides, considering the distance from the assumed sides j to the planar regions k in the current frame. This is one of the reasons the 3D point cloud for each cubic side is individually acquired. During the process of extraction, the sides extracted in the previous frame are compared with the detected planar regions in the current frame, based on the distance $D_k(j)$. A second failure case is when the comparison for a detected planar region in the previous frame fails for comparisons with every planar region in the current frame. At this point, the side estimated by the pose in the previous frame is compared with the area of every one of the planar regions in the current frame to determine a match.

3.3 Pose Estimation

Pose estimation is performed in two steps, global and local, and is carried out when three and two faces of the cube are visible. The case of when three sides of a cube are visible is described first. A global registration is performed with the translation of the cube being calculated by the intersection between three planes, and the rotation is calculated from the normal vectors of each side. After this step, the local pose estimation is performed. The point cloud is fitted for each side to the corresponding side of the cubic model. The fitting is performed with the Euclidean distance from points inside one cubic region to one side of cubic model is minimized with the steepest decent method. The model-fitting is done to three cubic sides for the local estimation.

In the case when only two sides of cube are captured, model-fitting is only performed on the two visible sides. For the global estimation, an intersection line is calculated between the two visible extracted regions, and points in two visible extracted regions are fitted to the model. This is followed by the determination of the cubic edge that is in the direction of the nonvisible side's normal vector. There are two possible directions for the normal vector, and a comparison of the centers of the point cloud with the cube model from the global estimation provides the direction. This normal vector is calculated by the cross products of the normal vectors of the two visible sides. We define a point that lies on an edge as a location where the depth values drastically change as compared to neighboring points. This detection of the edge in the 3D point cloud associated with the face requires the removal of outliers from a large number of sampled points. In this case, the outliers are caused from the user's hands and some random noise. A contour of the 3D point cloud associated with the face is analyzed to determine an approximate edge. The edge of cube is approximated by the points of contour near the edge which is predicted by the center of the areas and the size of cube. Statistical analysis removes outlier points that do not fit within 10% of the approximate edge. Finally, the cubic model is fitted to inlier points of the approximate edge and the calculated nonvisible normal for the localization.

3.4 Creation of Projection Graphics

The final step is to render the images onto the projection surfaces of the cube. The cube is broken down into a set of six planes, and each of the planes provides a unique view into the graphical object rendered as it was inside the cube. The user's viewpoint is determined by skeleton tracking with the RGB-D camera, and enables the rendering of 3D graphical objects correctly from the user's viewpoint of the physical cube. Six individual textures are created (one for each plane of the cube). Then the textures are mapped onto the visible sides of cube from the viewpoint of the projector.

4 EXPERIMENT

The performance of our method was evaluated in terms of the accuracy of tracking performance. Example interactive applications were implemented with one or multiple cubes to show the capability of our method. All of the experiments were performed with the following system: CPU - Inter Core i7-4770k 3.50GHz, RAM - 16.0GB, GPU - NVIDIA GeForce GTX 760, RGB-D camera - Microsoft Kinect v1.0 (640x480), and Projector - EPSON EB-W8 (1280x1024). Figure 5 depicts our experimental setup.

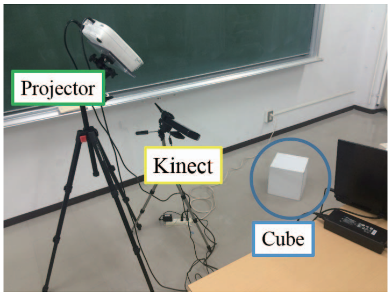

Figure 5: Setting for experiment

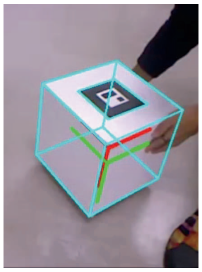

Figure 6: Placement of marker

4.1 Accuracy Evaluation

The accuracy experiment tracked a 250mm cube to determine the accuracy of the pose estimation for our system. We freely moved and rotated the cube with hands. A comparison of our proposed method against the ICP was performed with the ARToolkit method [8] as the ground truth. An ARToolkit marker was placed on the top face of the cube, see Figure 6. The error was determined as the difference between a measured pose (our method and the ICP) and the recorded ARToolkit pose. We calculated the error of translation and the error of rotation of the cube in 176 simultaneous frames. Our method and the ICP were measured on exactly the same 176 frames. The results are shown Table 1.

As shown in Table 1, the mean rotation errors are quite small as estimated by both our method and the ICP. The reason for the small error in our method is that the normal vectors of a similar direction are clustered together in the step for plane segmentation, and we calculate the normal vector of a plane equation in the regions. The ICP also has only a small effect on the error of rotation, as the points on the same plane are corresponded for the transformation.

The mean error for translation in our method is much smaller than in the ICP. The estimated translation for our purposed method is more stable, as shown with a lower standard deviation.

Table 1: Mean errors

	Proposed method	ICP
Translation (mm)	12.86 SD 6.07	31.02 SD 15.35
Angle (degrees)	0.36 SD 0.34	0.57 SD 0.44

4.2 Dynamic Projection on a Tangible Cube

Our Rubix system provides an intuitive method for viewing generic 3D graphical objects. The tangible nature of the physical projection object allows a user to move and rotate objects in any direction and the object appears to be inside the cube. The Rubix provides similar benefits as with pCubee, but without the previously mentioned limitations of cabling, fragile nature, weight, fixed size and display bevels. We found the dynamic projection onto the cube to be quite responsive. During the experiment, our method performed at 12.4 fps, and this proved to be quick enough to track the cube while moving it with our hands.

4.3 Multiple Objects

We experimented with a number of methods to support multiple objects tracked simultaneously with the Rubix system. Figure 7 depicts the system rendering and allowing the manipulation of three different cubes. However the limiting factors for increasing the number of tracked objects is the field of view of the RGB-D camera. Our method of overcoming the field of view problem is to increase the number of cameras. We tried to connect two cameras to a single computer with one projector providing the rendering. This is common method to increase RGB-D camera tracking volumes [9], and we demonstrated Rubix can increase its tracking volume through the use of additional cameras.

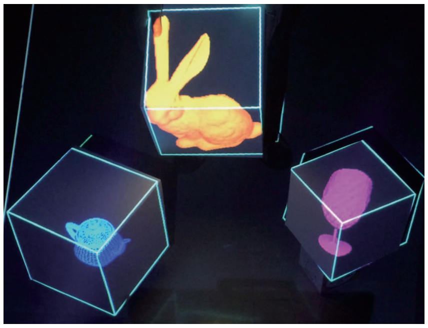

Figure 7: Using multiple physical objects

4.4 Command Entry

The major form of interaction with the Rubix system is through direct manipulation to change the user's viewpoint and move and rotate the cube. We wish to supply additional support to enable an application to receive command entry information from the user, such as changing the graphical object. We developed a set of example button and menu applications to demonstrate how Rubix may support user interactions. These applications are used as exemplars on the possible interaction techniques the Rubix system could support. While there are numerous user interface controls, we have chosen to focus on buttons and menus, two of the most common forms, as an initial exploration into possible control techniques. The buttons represent the ability to send simple binary commands to the application, and the menus represent hierarchical command structures. Our goal for supporting user interactions is to leverage the RGB-D camera as the only sensor for detecting user interactions. The user interface controls are all implemented with a region-based touch method. The Rubix system is able to determine a particular region on a specific face of the cube with the touch of the user's hand.

5 Conclusion

We proposed and demonstrated the Rubix system, as a dynamic SAR environment. The Rubix system is able to provide an accurate and stable method for 6DOF tracking on a textureless cube in SAR. In our proposal, a 3D point cloud from a RGB-D camera is divided into planar regions by superpixel segmentation and labeling. These regions of the targeted cubic sides are then extracted into a set of planar regions. The points on the face sides of these planar regions are fitted to a cubic model to estimate cubic pose. Real-time user viewpoint dependent 3D graphical projection images are created from the cubic pose and the position of the projector. We validated the Rubix system against the ICP algorithm. Our method was shown to be superior in both orientation and translation pose estimations. The system performs at a speed of 12.4 fps, which is effective for tangible interactions. The system supports multiple objects being tracked simultaneously. Example applications were developed to demonstrate the capabilities of the Rubix system.

Acknowledgements

This work was supported in part by a grant from Australian Re- search Council, JSPS Grant-in-Aid for Scientific Research(S) 24220004, and JST CREST ”Intelligent Information Processing Systems Creating Co-Experience Knowledge and Wisdom with Human-Machine Harmonious Collaboration”.

References

[1] M. R. Marner, B. H. Thomas, and C. Sandor. Physical-virtual tools for spatial augmented reality user interfaces. In *8th IEEE International Symposium on Mixed and Augmented Reality*, pages 205-206. IEEE, 2009.

[2] S. Audet, M. Okutomi, and M. Tanaka. Augmenting moving planar surfaces robustly with video projection and direct image alignment. *Virtual Reality*, 17(2):157 - 168, 2013.

[3] I. Stavness, B. Lam, and S. Fels. pcubee: A perspective-corrected handheld cubic display. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI'10, pages 1381-1390, New York, NY, USA, 2010. ACM.

[4] Y. Tang, B. Lam, I. Stavness, and S. Fels. Kinect-based augmented reality projection with perspective correction. In *ACM SIGGRAPH 2011 Posters*, SIGGRAPH'11, pages 79:1-79:1, New York, NY, USA, 2011. ACM.

[5] D. Kobayashi and N. Hashimoto. Spatial augmented reality by using depth-based object tracking. In *ACM SIGGRAPH 2014 Posters*, SIGGRAPH'14, pages 33:1-33:1, New York, NY, USA, 2014. ACM.

[6] C. Resch, P. Keitler, G. Klinker, Sticky projections - A new approach to interactive shader lamp tracking, In *13th IEEE International Symposium on Mixed and Augmented Reality*, pages 151-156. IEEE, 2009.

[7] K. Matsumoto, F. de Sorbier, and H. Saito. Real-time enhancement of rgb-d point clouds using piecewise plane fitting. In *5th European Workshop on Visual Information Processing*, 2014.

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

[9] B. Kainz, S. Hauswiesner, G. Reitmayr, M. Steinberger, R. Grasset, L. Gruber, E. Veas, D. Kalkofen, H. Seichter, and D. Schmalstieg. Omnikinect: real-time dense volumetric data acquisition and applications. In *Proceedings of the 18th ACM symposium on Virtual reality software and technology*, pages 25-32. ACM, 2012.