Tech-note: Spatial Interaction using Depth Camera for Miniature AR

Kyungdahm Yun* Woontack Woo†

GIST U-VR Lab.

ABSTRACT

Spatial Interaction (SPINT) is a non-contact passive interaction method that exploits a depth-sensing camera for monitoring the spaces around an augmented virtual object and interpreting their occupancy states as user input. The proposed method provides 3D hand interaction requiring no wearable device. The interaction schemes can be extended by combining virtual space sensors with different types of interpretation units. The depth perception anomaly caused by an incorrect occlusion between real and virtual objects is also alleviated for more precise interaction. The fluid interface will be used for a new exhibit platform, such as Miniature AR System (MINARS), to support a dynamic content manipulation by multiple users without severe tracking constraints.

Keywords: spatial interaction, depth camera, augmented reality

Index Terms: H.5.1 [INFORMATION INTERFACES AND PRESENTATION]: Multimedia Information Systems—Artificial, augmented, and virtual realities; I.3.6 [COMPUTER GRAPHICS]: Methodology and Techniques—Interaction techniques

1 INTRODUCTION

Miniature AR System (MINARS) is an exhibit platform that creates an augmented space from static scale models to support a dynamic content manipulation as shown in Figure 1. Considering the need for the system to display to a large group of people in public, a natural hand interaction is highly desirable without depending on wearable devices. Users also need several widgets for virtual 3D object manipulation, while the correct depth perception from the augmented scene is being retained.

(a) (b)

Figure 1: A Miniature AR System for a campus. (a) Original scale models. (b) An augmented scene with a virtual fountain and characters. Placing a hand in the space triggers the scene change.

In this paper, we propose a novel interaction method that exploits the interaction space around virtual objects for the Miniature AR System. The physical occupancy of each interaction space is monitored by virtual smart sensors using a depth-sensing camera. A user's action in the space can then be interpreted as a 3D input for the system. The proposed method does not require any wearable interface, while retaining flexibility for complex physical input and expandable by allowing additional widget composition. Furthermore, it does not suffer from a false occlusion between real and virtual objects which has been a critical issue for natural interaction in augmented reality.

*e-mail: kyun@gist.ac.kr

†e-mail: wwoo@gist.ac.kr

IEEE Symposium on 3D User Interfaces 2009
14-15 March, Lafayette, Louisiana, USA
978-1-4244-3812-9/09/$25.00 ©2009 IEEE

2 BACKGROUND AND RELATED WORKS

AR interfaces can be classified as active or passive. Active methods directly locate a subject for initiating interaction, while passive methods simply observe possible targets. They are further categorized as contact or non-contact, depending on whether any wearable or attachable equipment is required for tracking.

Active Interaction Contact active methods allow hand tracking in a 3D AR space, but require additional devices such as a paddle or a glove with fiducial markers [1]. Non-contact active methods have also been proposed, but reveal some major constraints on the interaction subjects. One example, HandVu [8] supported hand gesture commands, but its tracking was restricted to a 2D image plane. Handy AR [9] assumed a stationary hand posture for 3D pose estimation, but resulted in a limited number of possible interaction types. Finger or pen-based interactions using dense stereo range data have also been explored, but posed constraints on a planar scene background and a sharp pointing subject [5].

Active methods are also restricted by the direct tracking of subjects, as it is often challenging and inefficient when only simple interactions are required. These issues can be avoided if the focus of tracking is transferred to the target objects. Since one objective of interaction is to act upon objects in the space, it is more important to understand what is being approached than how it is being approached. Non-contact passive methods usually satisfy this property in a more cost-efficient way.

Passive Interaction SpaceSensor observed any physical movement in an eight section space surrounding the user by assessing a depth map acquired from multiple view cameras [12]. While it successfully detected body motion, the traits such as the fixed shape of the space, relatively large volume requirement, and limited interaction range prevented it from discreet interaction. It could only be applied to a body-level interface for controlling a virtual reality application and could not make a direct spatial mapping between the interface and the application space [6]. The Visual Interaction Cues (VICs) paradigm also investigated a passive interaction model [3]. As an aspect of cascade matching, 3D gesture volume was suggested for extracting 3D appearance and motion features from stereo images [14]. Its use of depth information, however, was limited for distinguishing touch action on a planar surface and mostly supported only 2D gestures.

Depth Acquisition Non-contact passive interaction methods need to remotely recognize any change applied to the interaction objects. Hence, a range sensor conforming to non-contact property and producing a realtime dense depth map with an accompanying color image are required. ZCam [13], an infrared-based depth-sensing camera from 3DV Systems, was found to meet these criteria. Registration in such a camera is essential for combining acquired depth information with an image scene. While a depth camera has been previously used for some research connecting physical

and virtual objects on a table-top system [10], no known calibration process for ZCam has been specifically designed for existing AR frameworks.

Natural Occlusion The false occlusion between real and virtual objects caused by an incorrect rendering order of the augmented scene is not only an issue of aesthetics, but also of usability [2]. Users are often confronted with incorrect depth perception which significantly degrades the ability to manipulate virtual objects. Available depth information can be used for scrutinizing an exact boundary between virtual and real objects, providing the correct occlusion [4, 7].

3 Proposed Method

Spatial Interaction (SPINT) is a non-contact passive interaction method that recognizes user input by assessing a physical occupancy of an interaction subject $\mathscr{S}$ against an augmented space $\mathbf{S}_i$ created from each interaction object $\mathscr{O}_i$ registered in the system. A brief process is depicted in Figure 2, where each oval represents a corresponding section of the paper.

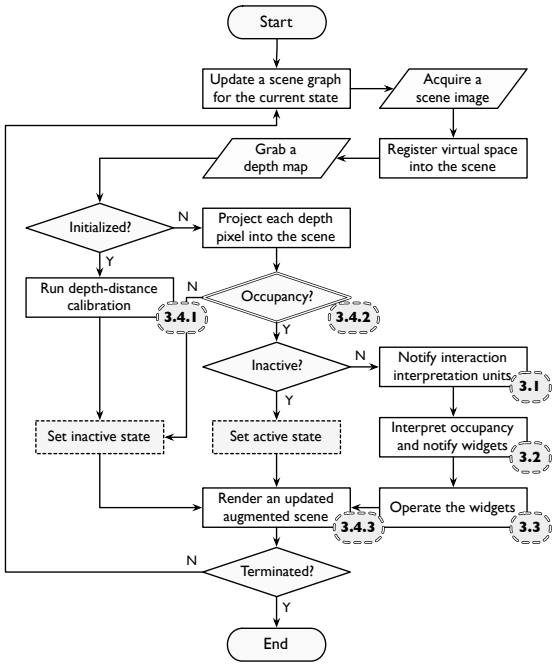


Figure 2: Spatial Interaction (SPINT) flowchart.

3.1 Nested Structure

An augmented space $\mathbf{S}_i$ consists of an interaction layer $\mathbf{I}_i$ and an activation layer $\mathbf{A}_i$. Figure 3 illustrates this structure.

Interaction Layer It contains a group of virtual space sensors which recognize any physical contact from an interaction subject $\mathscr{S}$. A space sensor c is a small cubic box whose occupancy state is determined to be one of $\{\mathsf{F},\mathsf{O}\}$. F is free and O is occupied. Once occupancy occurs, a set of registered interaction interpretation unit $\mathbf{U}_{ij} = \{u | u \in \mathbf{w}_{ij} \in \mathbf{W}_i\}$ receive propagated state information and accordingly interpret an action of interaction subject. These units constitute different types of interaction widget $\mathbf{w}_{ij}$ in $\mathbf{W}_i = \{w | w \in \mathbf{I}_i \in \mathbf{S}_i\}$ as a user interface.

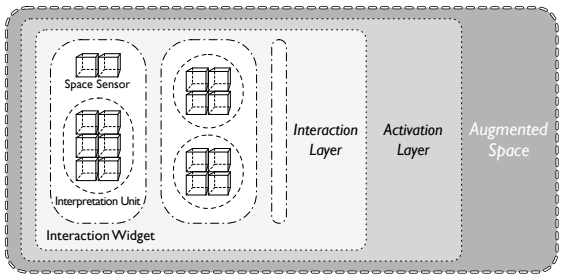


Figure 3: Nested structure of the augmented space.

Activation Layer It provides an access to the interior interaction layer $\mathbf{I}_i$ while covering an outer area of an augmented space $\mathbf{S}_i$. Every augmented space is initially set to be visually transparent and physically impotent so that an entire scene does not become cluttered with interface handles. When an arbitrary interaction subject $\mathscr{S}$ penetrates into an activation layer $\mathbf{A}_i$, the current space $\mathbf{S}_i$ becomes activated and appears in the scene, initiating observation on the states of internal interaction interpretation units.

3.2 Interpretation Unit

A set of space sensors $\mathbf{C}_{ijk} = \{c | c \in \mathbf{u}_{ijk} \in \mathbf{U}_{ij}\}$ with an arbitrary size and layout can constitute an interpretation unit $\mathbf{u}_{ijk}$. Each unit generates an independent interpretation of the occupancy states received from the subordinate space sensors. Multiple units may share one space sensor. Currently three types have been designed.

Direction Extractor A motion direction of the subject $\mathscr{S}$ penetrating an interpretation unit space $\mathbf{u}_{ijk}$ is extracted. A set of space sensors $\mathbf{C}_{ijk}(t,s)$ with state s at time t is represented by equation 1.

$$\mathbf{C}_{ijk}(t,s) = \{c | c \in \mathbf{C}_{ijk}, c(t) = s\} \tag{1}$$

A newly activating set of space sensors $\mathbf{C}^+_{ijk}(t)$ and a deactivating set $\mathbf{C}^-_{ijk}(t)$ can be obtained from equations 2 and 3, respectively.

$$\mathbf{C}^+_{ijk}(t) = \mathbf{C}_{ijk}(t,\mathsf{O}) \cap \mathbf{C}_{ijk}(t-1,\mathsf{F}) \tag{2}$$

$$\mathbf{C}^-_{ijk}(t) = \mathbf{C}_{ijk}(t,\mathsf{F}) \cap \mathbf{C}_{ijk}(t-1,\mathsf{O}) \tag{3}$$

A global direction vector $v_{ijk}(t)$ of the space $\mathbf{u}_{ijk}$ is then calculated by equation 4. *pos* is the function to calculate the current position of the given sensor.

$$v_{ijk}(t) = \frac{\sum_{e \in \mathbf{C}^+_{ijk}(t)} pos(e)}{|\mathbf{C}^+_{ijk}(t)|} - \frac{\sum_{s \in \mathbf{C}^-_{ijk}(t)} pos(s)}{|\mathbf{C}^-_{ijk}(t)|} \tag{4}$$

Delayed Selector A selection of an interpretation unit space $\mathbf{u}_{ijk}$ is determined by an occupancy ratio due to the subject $\mathscr{S}$. The occupancy ratio $r_{ijk}(t)$ at time t is calculated by equation 5.

$$r_{ijk}(t) = \frac{|\mathbf{C}_{ijk}(t,\mathsf{O})|}{|\mathbf{C}_{ijk}(t,\mathsf{F}) \cup \mathbf{C}_{ijk}(t,\mathsf{O})|} \tag{5}$$

The occupancy duration $t_{ijk}(t)$ is then obtained by equation 6 and 7 with a ratio threshold $\delta_{r_{ijk}}$.

$$t^*_{ijk} = \begin{cases} t & \text{if } r_{ijk}(t) > \delta_{r_{ijk}} \text{ and } t^*_{ijk} = 0 \\ t^*_{ijk} & \text{if } r_{ijk}(t) > \delta_{r_{ijk}} \text{ and } t^*_{ijk} \neq 0 \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

$$t_{ijk}(t) = \begin{cases} t - t^*_{ijk} & \text{if } r_{ijk}(t) > \delta_{r_{ijk}} \\ 0 & \text{otherwise} \end{cases} \quad (7)$$

If $t_{ijk}(t) > \delta_{t_{ijk}}$ is satisfied with the duration threshold $\delta_{t_{ijk}}$, the space $\mathbf{u}_{ijk}$ is then determined to be selected.

Path Detector A gesture path $\mathbf{p}_{ijk}$ in an interpretation space $\mathbf{u}_{ijk}$ is a sequence of arbitrary space sensors defined by equation 8.

$$\mathbf{p}_{ijk} = (c_{ijk1}, c_{ijk2}, \cdots, c_{ijkn}),\ c_{ijkl} \in \mathbf{C}_{ijk} \quad (8)$$

Once the sensors from c_{ijk1} to c_{ijkn} have been sequentially occupied, $\mathbf{p}_{ijk}$ is recognized as a gesture, following Algorithm 1.

Algorithm 1 Recognize a gesture path $\mathbf{p}_{ijk}$

```
while p_ijk is not empty do
    if p_ijk.front() ∈ C_ijk(t, O) then
        p_ijk.pop_front()
    else
        break
    end if
end while
if p_ijk is empty then                    ▷ p_ijk is recognized
else                                      ▷ p_ijk is not (yet) recognized
end if
```

3.3 Interaction Widget

An interaction widget $\mathbf{w}_{ij}$ is composed of a set of interpretation units $\mathbf{U}_{ij} = \{u | u \in \mathbf{w}_{ij} \in \mathbf{W}_i\}$. An arbitrary combination of these units can yield an interaction widget that works as a high level interface to the user input.

Vertical Slider A set of space sensors distributed in a vertical layout is registered to a direction extractor. A direction indicator of the widget moves along 1D axis like a scrollbar in GUI, emitting the relative position of the indicator to the widget.

Selection Button A group of space sensors is registered to a delayed selector. If any interaction subject has stayed in the widget longer than a specified period, a selection signal is emitted.

Gesture Recognizer A sequence of space sensors linked in a certain path of gesture is registered to a path detector. Once an interaction subject has walked through the path, the gesture is recognized.

3.4 Depth Camera

Depth-Distance Calibration The ZCam consists of two sensors; one is an ordinary color image sensor and the other is a range sensor measuring the time-of-flight of infrared rays reflected back from any objects in the scene. A stream of depth map can be acquired up to the rate of 320 by 240 pixels at 30 fps. As the grabbed depth value d_{depth} is in an 8 bits relative level, the region of interest must be manually set by using two values: distance z and width w. The near plane is placed at z from the camera and the far plane is placed at $z+w$. Hence, the real distance should be $d_{distance} = \frac{w}{255} d_{depth} + z$. In practice, $d_{distance}$ is not consistent with the camera setting, but contains an error due mainly to the reflectance of the surrounding environment. For an accurate result from the occupancy check and occlusion test, d_{depth} from the ZCam and $d_{distance}$ from the tracker should be in calibration.

An average depth $\tilde{d}_{depth}(t)$ in an area of the marker in the image $\mathbf{M}$ at time t is calculated by equation 9.

$$\tilde{d}_{depth}(t) = \frac{1}{|M|} \sum_{i \in \mathbf{M}} d_{depth}(t,\ i) \quad (9)$$

The distance $\tilde{d}_{distance}(t)$ between the camera and the marker is obtained from a translational component of the model-view transformation matrix. Coefficients α and β for a depth-distance calibration model $d_{distance} = \alpha d_{depth} + \beta$ can be then estimated online by using a linear least squares method with an adequate sample size ($n = 500$). Note that samples of upper and lower extreme depth values are not considered in the fitting procedure, as the saturated values do not follow the model.

Space Occupancy Check The occupancy for each space around an interaction object is determined by investigating any penetration or collision between a point in a depth map and the geometry of the object. The depth map from the camera contains pixels transformed to screen coordinates. As the geometry of the object is represented as a polygonal mesh in a world coordinate, a transformation in either direction between the two entities is necessary.

The occupancy state of the space sensor is identified by Algorithm 2, which checks if a line l projected from a point p_{screen} of a depth map $\mathbf{D}$ collides with any sensor c in the augmented space $\mathbf{S}$. For an efficient collision check, all meshes are sorted in a k-D Tree.

Algorithm 2 Check the occupancy of each space sensor

```
for all p_screen ∈ D do
    p_world ← screen2world(p_screen)
    l ← line(p_world)
    if l intersects c ∈ S then            ▷ c is occupied
    end if
end for
```

*screen*2*world* transforms a point from a screen coordinate to a world coordinate, and *line* generates a line segment for collision detection from $l_s = p_{world}$ to $l_e = l_s + \kappa v_l$. v_l is a direction away from p_{world} to the camera center c_{world}, and κ is a constant for controlling the length of the projection line segment. As the depth map cannot provide any information about the regions behind the frontal objects, an appropriate length of projection may vary along the physical form of interaction subjects.

Object Occlusion Test When writing a fragment to the frame buffer $\mathbf{F}$, the distance between the objects in occlusion should be tested to prevent the virtual objects from always being rendered in front of the real scene. The comparison between a fragment depth d_{object} of virtual object and a depth d_{scene} of the same point acquired from the ZCam is explained in Algorithm 3.

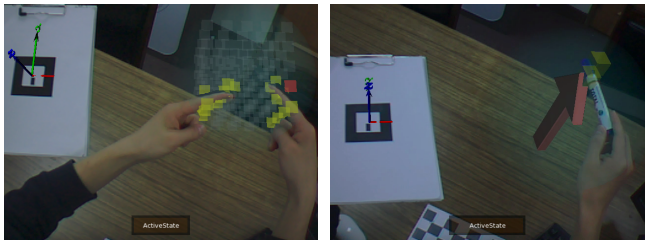
Algorithm 3 Test occlusion in the frame buffer and the depth map

```
for i = 1 to |F| do
    d_object ← F(i).z
    d_scene ← depth2distance(D(i))
    if d_object > d_scene then            ▷ object is occluded
    else                                  ▷ object is not occluded
    end if
end for
```

4 Experiments and Discussion

SPINT was initially tested on a desktop AR environment with a fixed camera position, rather than directly on the MINARS table which has been developed with a 3D model-based tracker [11]. An augmented space was created in a 3D space established by a fiducial marker and three types of widgets were individually triggered. Note that the marker in the figures was only used for setting up a global coordinate, not for tracking fingers or other interaction subjects. For larger images, please refer to the accompanying video material.

Non-contact Passiveness Figure 4 shows that it is possible to use different subjects for the same augmented space, thus eliminating the explicit requirement of a contact input device. The use of arbitrary subjects enables more flexible 3D interaction, as users may choose a tool that best fits the current context.

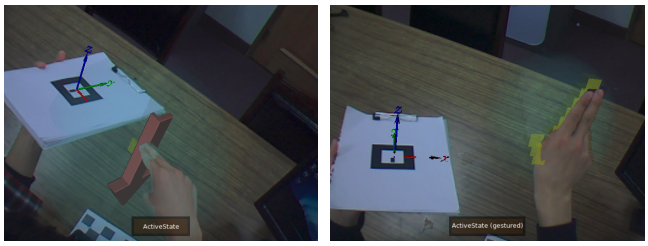


(a) With two hands. (b) With a pen.

Figure 4: Arbitrary subjects accessible for interaction.

We also compared the performance of a spatially extracted direction of the motion with one calculated from marker tracking. The augmented space was composed of a $5 \times 5 \times 5$ cubes of 15 mm side, while the size of the marker was 40 mm for both sides. Correlations between the two results were 0.7467, 0.5074, and 0.6422 for each coordinate axis, showing similar peaks and trends.

Interaction Widgets Figure 5 provides some examples of the widgets explained in the previous section. A vertical slider presents an arrow that follows the user's hand. It would be placed atop a virtual building in MINARS to provide a scale operation. A delayed selector becomes selected when a full hand is inserted, but ignores input from only one or two fingers. It could be used for triggering augmented annotations of the miniatures. A gesture recognizer accepts a 'V'-shaped checking gesture moving from left to right. More critical operations such as object removal and weather change would be confirmed with this gesture.



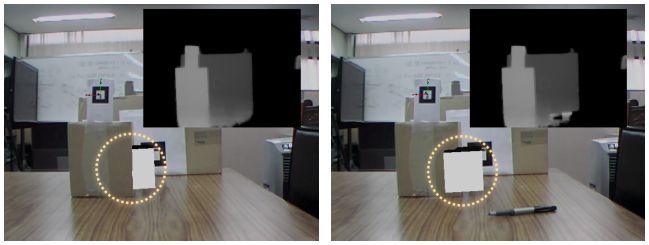
(a) Vertical slider. (b) Gesture recognizer.

Figure 5: Sample interaction widgets.

Natural Occlusion Figure 5(b) displays a proper depth cue because the real hand in front occludes the virtual cubes behind. Hence, the user can comfortably interact without losing 3D spatial context. Figure 6 shows that our depth-distance calibration improved the accuracy of occlusion. When a physical object was inserted for a virtual object located 300 mm away from the origin, an occlusion occurred at 299 mm in the calibrated scene. It was more accurate than the 388 mm in an uncalibrated case.

5 Conclusion

In the future, we plan to design new interpretation units and widgets for more versatile interaction. MINARS is currently being developed for demonstrating scenarios such as a campus navigation scenario, and will incorporate SPINT as the primary interface. Once fully integrated, performance and usability of the whole system will be evaluated.



(a) Occlusion in the calibrated scene. (b) Error in the uncalibrated scene.

Figure 6: A point of occlusion between the real and virtual objects.

Acknowledgements

This research was supported by the CTI development project, MCT, and KOCCA in South Korea.

References

[1] M. Billinghurst, H. Kato, and I. Poupyrev. The MagicBook - moving seamlessly between reality and virtuality. *IEEE Computer Graphics and Applications*, 21(3):6–8, May 2001.

[2] V. Buchmann, T. Nilsen, and M. Billinghurst. Interaction with partially transparent hands and objects. *AUIC '05: Proc. of the 6th Australasian User Interface Conference*, 40:17–20, Jan 2005.

[3] J. Corso, G. Ye, D. Burscbka, and G. Hager. A practical paradigm and platform for video-based human-computer interaction. *IEEE Computer*, 41(5):48–55, May 2008.

[4] J. Fischer, B. Huhle, and A. Schilling. Using time-of-flight range data for occlusion handling in augmented reality. *EGVE '07: Proc. of the 13th Eurographics Symposium on Virtual Environments*, pages 109–116, Jul 2007.

[5] G. Gordon, M. Billinghurst, M. Bell, J. Woodfill, B. Kowalik, A. Erendi, and J. Tilander. The use of dense stereo range data in augmented reality. *ISMAR '02: Proc. of the 1st IEEE/ACM International Symposium on Mixed and Augmented Reality*, pages 14–23, Sep 2002.

[6] D. Hong and W. Woo. A 3D vision-based ambient user interface. *International Journal of Human-Computer Interaction*, 20(3):271–284, Jul 2006.

[7] M. Kanbara, T. Okuma, H. Takemura, and N. Yokoya. A stereoscopic video see-through augmented reality system based on real-time vision-based registration. *VR '00: Proc. of the 2nd IEEE Virtual Reality Conference*, pages 255–262, Mar 2000.

[8] M. Kolsch, R. Bane, T. Hollerer, and M. Turk. Multimodal interaction with a wearable augmented reality system. *IEEE Computer Graphics and Applications*, 26(3):62–71, May 2006.

[9] T. Lee and T. Hollerer. Handy AR: Markerless inspection of augmented reality objects using fingertip tracking. *ISWC '07: Proc. of the 11th IEEE International Symposium on Wearable Computers*, pages 83–90, Oct 2007.

[10] J. Leitner, M. Haller, K. Yun, W. Woo, M. Sugimoto, and M. Inami. IncreTable, a mixed reality tabletop game experience. *ACE '08: Proc. of the 5th International Conference on Advances in Computer Entertainment Technology*, pages 9–16, Dec 2008.

[11] Y. Park, V. Lepetit, and W. Woo. Multiple 3D object tracking for augmented reality. *ISMAR '08: Proc. of the 7th IEEE/ACM International Symposium on Mixed and Augmented Reality*, pages 117–120, Sep 2008.

[12] W. Woo, N. Kim, K. Wong, and M. Tadenuma. Sketch on dynamic gesture tracking and analysis exploiting vision-based 3D interface. *Proc. of SPIE*, 4310(1):656–666, Dec 2000.

[13] G. Yahav, G. Iddan, and D. Mandelboum. 3D imaging camera for gaming application. *ICCE '07: Proc. of the 25th International Conference on Consumer Electronics*, pages 1–2, Jan 2007.

[14] G. Ye, J. Corso, and G. Hager. Visual modeling of dynamic gestures using 3D appearance and motion features. In *Real-Time Vision for Human-Computer Interaction*, chapter 7, pages 103–120. Springer-Verlag, Aug 2005.