An advanced interaction framework for augmented reality based exposure treatment

Sam Corbett-Davies*[1][2] Andreas Dünser†[1] Richard Green‡[2] Adrian Clark§[1]

[1]Human Interface Technology Lab NZ, [2]Department of Computer Science
University of Canterbury, New Zealand

Abstract

In this paper we present a novel interaction framework for augmented reality, and demonstrate its application in an interactive AR exposure treatment system for the fear of spiders. We use data from the Microsoft Kinect to track and model real world objects in the AR environment, enabling realistic interaction between them and virtual content. Objects are tracked in three dimensions using the Iterative Closest Point algorithm and a point cloud model of the objects is incrementally developed. The approximate motion and shape of each object in the scene serve as inputs to the AR application. Very few restrictions are placed on the types of objects that can be used. In particular, we do not require objects to be marked in a certain way in order to be recognized, facilitating natural interaction. To demonstrate our interaction framework we present an AR exposure treatment system where virtual spiders can walk up, around, or behind real objects and can be carried, prodded and occluded by the user. We also discuss improvements we are making to the interaction framework and its potential for use in other applications.

Keywords: Augmented reality, 3D interaction, Kinect, environment awareness, exposure treatment

Index Terms: H.5.1 [Information Interfaces and Presentation]: Multimedia Information Systems—Artificial, augmented, and virtual realities

1 Introduction

Augmented reality (AR) is slowly moving from being a simple visualization tool to an immersive, interactive medium. Interactive augmented reality is an emerging field [1], with applications in entertainment [5, 13, 22, 17], education [9], user interfaces [22, 20] and rehabilitation [4, 14, 7]. We were motivated to develop a new interaction method during development of an interactive AR exposure treatment (ARET) system for spider phobia. We found that existing interaction methods did not allow the user to interact naturally with the virtual spiders in the system. This was principally because existing methods were limited either in the objects that could be used for interaction or in the scope of possible interactions. In this paper we present an interaction framework that allows arbitrary objects to interact physically in tabletop AR environments, and demonstrate the framework in an ARET system.

Our system uses a Microsoft Kinect mounted above the tabletop to record 3D information about the AR space (see Fig. 1). This information is used both to develop a model of the environment and to track and model real-world objects moving within it. By knowing the shape and movement of real-world objects, we can create virtual content that more realistically reacts to and interacts with the real world. One particular advantage of our system is that we place almost no limitation on what objects can interact in the scene; as long as an object is visible to the Kinect it can be used for interaction.

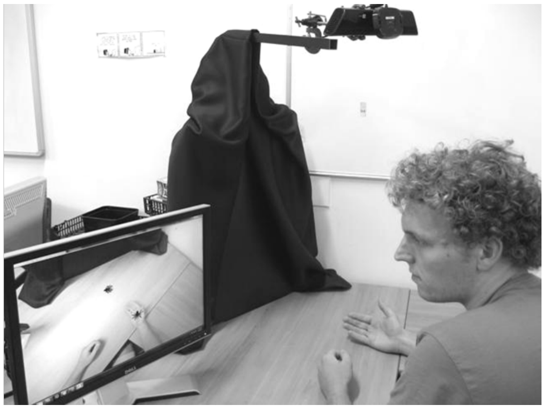

Figure 1: The required setup for our interaction framework, with the Kinect mounted over the interaction space.

Our framework allows users to interact with virtual content intuitively using natural gestures and physical objects. In this paper we present an application where arachnophobic patients can interact with virtual spiders to help overcome their fears, but our interaction framework is general enough to be used in a wide range of applications.

In Sections 2 and 3 we discuss the development of the framework, before presenting an ARET application based on the framework in Section 4.

2 Background

Our framework seeks to enable the better integration of the real world in AR applications, building on previous work in the areas of interactivity and environmental awareness in augmented reality. In this section we review the important research in these areas.

2.1 Interactive AR

The oldest AR interaction technique used specially-designed fiducial markers which were tracked in the real world and used for specifically-defined interactions with AR objects. A number of “paddle-based” interaction methods have been developed, including [9], which allowed users to manipulate virtual household objects with a marked “paddle” in an interior design application. Such interaction techniques are inherently limited in the scope of interactions possible, because every interaction device must be marked. This requirement also limits the possibility of natural interaction, although more recently [20] used a multi-colored glove to detect hand poses for interaction.

*samcorbettdavies@gmail.com
†andreas.duenser@canterbury.ac.nz
‡richard.green@canterbury.ac.nz
§adrian.clark@canterbury.ac.nz

IEEE Virtual Reality 2013
16 - 20 March, Orlando, FL, USA
978-1-4673-4796-9/13/$31.00 ©2013 IEEE

Methods have been developed to facilitate interaction without the need for markers, by fitting computer models to scene observations. Such methods have become particularly popular since the introduction of the Microsoft Kinect, a consumer-priced depth sensing camera, because depth data allows for much more robust model fitting. [17] and [10] are two notable examples of model-based interaction using the Kinect; the former fits a full-body skeleton and the latter an articulated hand model. Both methods allow complicated natural interaction without markers by detecting gestures and poses.

As with marker-based methods, the main disadvantage of this approach is the lack of flexibility, as all objects considered for interaction must have predefined models. Attempts to generalize the models to fit a class of objects exist [17, 10], but these often encounter problems due to object variations within the class (such as body size and shape for human pose detection). Our proposed framework attempts to track objects without the need for predefined models, with the assumption that objects undergo minimal non-rigid deformation.

2.2 Environmental Awareness in AR

Research has shown that a stronger perceptual connection between real and virtual content in AR can be established if virtual content behaves realistically in the physical environment [18]. The realism of AR can be increased by ensuring correct responses of the virtual object to gravity, collision with real objects [2], realistic shadowing and lighting effects [21], and occlusion between real and virtual content [11].

Early work in this area required manual creation of a model of the environment [11], which is both time consuming and inflexible, requiring modification of the model if the environment changes [12]. Later approaches attempted to model the environment automatically using stereo camera depth information [23] and online SLAM [19]. Although these approaches removed the need to manually create a model of the environment, they are susceptible to poor illumination in the scene, and fail in environments with low or repetitive textures.

In 2007, Wilson used a depth sensing video camera to reconstruct the physical surface of a table, and using an overhead projector displayed a virtual car on the table which would react realistically to obstacles [22]. Due to the projective nature of the display, the game was effectively two dimensional, reducing the realism and making it difficult to estimate the exact position of the car in three-dimensional space.

More recently the KinectFusion system supports real time environment reconstruction and basic interaction using the Microsoft Kinect device [5]. It only requires depth information obtained using structured infrared light, and is invariant to visible light illumination and texture quality. The authors demonstrate a physics simulation in the reconstructed environment, but little dynamic foreground interaction is evident. As KinectFusion uses the Kinect as a viewing camera and has no co-ordinate system origin in the real world, the scene can only be viewed from one viewpoint at a time.

3 Implementation

This section describes the operation of our interaction framework. The only infrastructure required is a Kinect mounted at least 0.8 m above the tabletop, facing down (see Fig. 1). Point cloud data from the Kinect is manipulated using the Point Cloud Library (PCL) [16]. To ensure real time performance, the raw point cloud from the Kinect is downsampled by a factor of 25. We found that this reduction in resolution actually improves the tracking performance by speeding up the processing time for each frame. This is because tracking quality degrades with frame rate; the longer the time between frames the farther the distance an object must be tracked over.

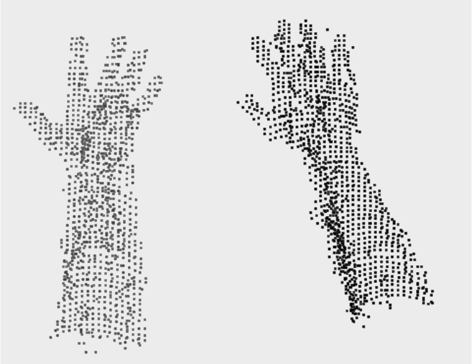

Figure 2: The point cloud representation of foreground objects.

3.1 Foreground Extraction

Foreground extraction is achieved by assuming the Kinect is in a fixed position above the interaction space. Our system keeps a dynamic record of the farthest depth observed by the Kinect at each pixel, which becomes the background depth map of the scene. Any pixel with a current depth value less that 98.5 % of this background depth is considered to be part of the foreground. This percentage value was determined by experimentation to be the value that minimized the false-positives in the foreground map while still allowing a hand placed flat on the tabletop to be classified as a foreground object. Any pixel with a depth gradient of more than 2 cm per pixel is removed form the foreground map, which serves to disconnect overlapping objects at different depth levels before connected-component labeling is performed

Morphological opening is applied to remove noise from the foreground map, removing most of the false-positive foreground pixels. Finally, the foreground map is segmented into objects using a connected-component labeling algorithm [3].

3.2 6DOF Model Tracking

Information about the motion of objects in the scene is required for realistic physics simulation. Objects are tracked in 6DOF using the Iterative Closest Point (ICP) algorithm. As advised by [15], we use projective correspondences and the point-to-plane error metric to align the objects' point clouds to the Kinect's observation in real time. Correspondences are rejected if the distance between pairs of points is greater than 5 mm. We found that this method had difficulty tracking translations in the x-y plane of the Kinect, so to overcome this we seed the ICP algorithm with an initial estimate of the object's translation. This estimate is calculated as the translation of the centroid of the connected component since the last frame. The transformation of the object between frames (as found by ICP) is stored with the object for use in physical simulation.

Tracking using ICP can fail when fast motions occur, which one might expect in our chosen application of arachnophobia therapy. We decided to use such a tracking regime because we are aiming to develop a general interaction framework, with exposure treatment being just one application. In the future we will implement object tracking on the GPU, which we believe will improve its performance during fast motion.

3.3 Model Updating

For each frame obtained from the Kinect, all models are aligned to the observed data using ICP (see Fig. 2). The models are updated with every new frame from the Kinect, both to improve the quality of the tracking and to allow for minor non-rigid deformation of objects. Every point in each model is assigned a weight between -1 and 1 which reflects the confidence that the model point is indeed

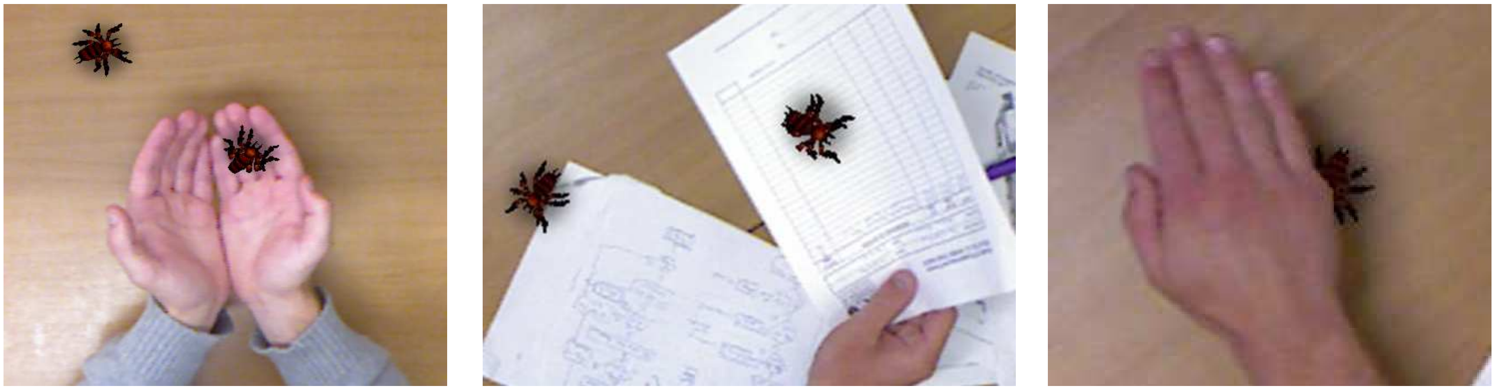

Figure 3: Screenshots of our ARET system showing a user interacting with virtual spiders. The right-most image shows a virtual spider being realistically occluded.

part of the object, and this weight is increased when its assumed position matches the observation, and decreased when it does not. Points with a weighting less than -1 are removed entirely. A linear weighting affinity was used for simplicity; better results could be achieved with a Kalman filter, which we will introduce in the future. The process for updating the set of model points is as follows:

1. Project all model points into the Kinect coordinate space.

2. For each pixel (**i**) in the depth map, calculate all model points which project to that location. Record the model point (**m**) which is closest to the camera.

3. Record the object (o) that model point m belongs to, and the connected component (c) the pixel i belongs to.

4. For each pixel **i**, calculate the corresponding point (**p**) in the point cloud observed by the Kinect.

5. Update individual model point value (see below).

6. For each connected component c, calculate which object o that most pixels belong to, using data recorded in Step 3. For all points in c not already in a model, add them to the model of object o with a weighting of -1.

The individual model point values were updated using the following approach:

1. If the absolute distance between **p** and **m** is less than a threshold ε_1, i.e. $\|\mathbf{p}-\mathbf{m}\| < \varepsilon_1$, the model point matches the observation. Its weighting is increased by α and its position is set to $\beta\mathbf{p}+(1-\beta)\mathbf{m}$.

2. If the model point lies significantly in front of the observation, i.e. $\|\mathbf{p}\| - \|\mathbf{m}\| > \varepsilon_1$, the model point's weighting is decreased by α, as the model point does not match the observation.

3. If the observation lies just in front of the model point, i.e. $\varepsilon_1 < \|\mathbf{m}\| - \|\mathbf{p}\| < \varepsilon_2$, add the observed point to the model with a weighting of -1, and leave the model point unchanged.

4. If the observation lies significantly in front of the model point, i.e. $\|\mathbf{m}\| - \|\mathbf{p}\| > \varepsilon_2$, leave the model point unchanged.

The optimal threshold values were determined by experimentation to be: ε_1 = 1 cm, ε_2 = 3 cm, $\alpha = 0.5$ and $\beta = 0.8$. A significant amount of tuning of these values was required to achieve good results. Future work will investigate how these parameters can be found automatically, allowing our system to be used in other setups.

3.4 Physical Interaction

Physical proxies of real objects in the scene are created in the Bullet physics engine[1] to facilitate physics simulation. The tabletop is represented with a triangulated mesh, with each vertex set to the corresponding value in the background depth map. Foreground objects are represented with collections of spheres with 5 mm radii. Only a subset of model points are represented; a 3D grid (with cube width 15 mm) is fit over the model point cloud and a single sphere used for each occupied box in the grid. The motion of these spheres is determined by the most recent transformation the object has undergone, as found in the ICP step. Bullet handles collisions between these spheres and other virtual content, allowing the user to dynamically interact with AR objects, pushing them, carrying them and even hitting them with real-world objects.

4 Application - AR Exposure Treatment

In this section we demonstrate the potential of our interaction framework in an interactive AR exposure treatment (ARET) system for spider phobia. ARET systems can display virtual fear stimuli in the real world and so allow clients can use their own body to interact with them. Studies have found that simple ARET systems are capable of inducing high levels of anxiety, which is a necessary prerequisite for such a system to be effective. However, existing systems are limited to only displaying moving virtual objects and do not allow interaction with the content.

ARET systems have been created for exposing clients to spiders [6] and cockroaches [7]. In these systems virtual insects are overlaid on top of ARToolkit markers [8] and animated with predefined basic motions. While the authors argue that one benefit of an AR system is that it allows patients to use real objects or their hands to interact with the stimuli [6], such interactivity has so far not been implemented in ARET systems.

We have developed an interactive ARET application (see Fig. 3) where virtual spiders are introduced into the physics simulation of our interaction framework. They are given a simple set of behaviors that cause them to walk around randomly, avoiding steep changes in depth (as caused by drop offs and insurmountable objects). The friction between the spiders and the physics proxies allows the spider to be realistically carried by the user, responding to the movements of their hand. The interactivity of our exposure treatment system exceeds that of any previous such systems.

To increase realism we also added occlusion to the system, passing the Kinect's depth information to a custom fragment shader that discards fragments of virtual objects which are behind real world ones. This allows near pixel-perfect occlusion from the viewpoint of the Kinect; however from alternate viewpoints (such as a HMD) the occlusion may not be as accurate, as we do not know the thick-

[1] http://bulletphysics.org

ness of the occluding object. The following section discusses improvements to be made in the area of occlusion.

5 Limitations and Future Work

Our interaction framework allows a new level of AR interaction, but is not without its limitations. We continue to work on the framework to address these. Firstly, although the tracking method we use is usually accurate enough for the purposes of interaction, it is occasionally prone to drift, particularly when objects rotate quickly. Secondly, while the framework usually runs at 25 Hz, the frame rate can fall below 20 Hz if the number of objects being handled by Bullet is too great. The framework is currently single-threaded, but many of the algorithms we use are parallelizable. Future work will seek to optimize the framework to run on multi-threaded processors and the GPU.

As our current focus is the interaction framework, at this stage the ARET scene can only be viewed from the Kinect's point of view, which results in a less immersive user experience. The reason for this is that we have not yet developed a means of realistically occluding virtual objects from arbitrary perspectives. The difficulty is that the Kinect cannot directly measure the thickness of objects, so it is difficult to determine if virtual content under them should be occluded. By developing 3D models of real objects as more views of them become visible to the Kinect, we anticipate our framework will allow more accurate handling of occlusion from alternate perspectives. We plan to achieve this by projecting the model points into the camera's image plane, and using a custom shader similar to the one described in Section 4 to correctly occlude virtual content from this new perspective.

Finally, while our framework facilitates physical interaction, the complexity of possible interactions does not match a number of the systems discussed in Section 2.1. This is because, to keep the framework as general as possible, no special gestures or interactions have been defined. Enabling more complicated natural interaction will require detection of higher-level gestures, such as that done in [20]. To this end, we are working on a comprehensive "gesture library" which will combine our interaction framework with hand tracking and gesture detection. We anticipate that this will once again improve the level of interactivity possible in AR applications.

6 Discussion and Conclusion

This paper has presented a discussion of the development of our interactive augmented reality framework. We have developed a method that allows a user to interact in a physically realistic way in an augmented reality scene using arbitrary objects. Our proposed method facilitates natural interaction, as it does not require markers or artificial constraints of any kind on objects used for interaction. We have also presented an interactive AR exposure treatment system as a demonstration application for our framework, in which our framework is used to allow a user to naturally and intuitively interact with virtual spiders. The work we have presented could be applied to many areas of augmented reality, from education to entertainment.

References

[1] M. Billinghurst. The Future of Augmented Reality in Our Everyday Life. In *Proceedings of the 19th International Display Workshops*, Nagoya, Japan, December 2011.

[2] D. E. Breen, E. Rose, and R. T. Whitaker. Interactive occlusion and collision of real and virtual objects in augmented reality. Technical report, Technical Report ECRC-95-02, ECRC, Munich, Germany, 1995.

[3] F. Chang, C.-J. Chen, and C.-J. Lu. A linear-time component-labeling algorithm using contour tracing technique. *Comput. Vis. Image Underst.*, 93(2):206–220, Feb. 2004.

[4] A. Dünser, R. Grasset, and H. Farrant. Towards Immersive and Adaptive Augmented Reality Exposure Treatment. In B. S. &. R. G. Wiederhold, B.K., editor, *Annual Review of Cybertherapy and Telemedicine 2011*, pages 37–41. IOS Press, Amsterdam, 2011.

[5] S. Izadi, R. A. Newcombe, D. Kim, O. Hilliges, D. Molyneaux, S. Hodges, P. Kohli, J. Shotton, A. J. Davison, and A. Fitzgibbon. KinectFusion: real-time dynamic 3D surface reconstruction and interaction. In *ACM SIGGRAPH 2011 Talks*, SIGGRAPH '11, pages 23:1–23:1, New York, NY, USA, 2011. ACM.

[6] M. C. Juan, M. Alcaniz, C. Monserrat, C. Botella, R. M. Banos, and B. Guerrero. Using augmented reality to treat phobias. *IEEE Computer Graphics and Applications*, 25(6):31–37, 2005.

[7] M. C. Juan, C. M. Botella, M. Alcaniz, R. M. Banos, C. Carrion, M. Melero, and J. A. Lozano. An augmented reality system for treating psychological disorders: Application to phobia to cockroaches. In *ISMAR*, pages 256–257.

[8] H. Kato and M. Billinghurst. Marker tracking and hmd calibration for a video-based augmented reality conferencing system. In *IWAR99*, pages 85–94.

[9] H. Kato, M. Billinghurst, I. Poupyrev, K. Imamoto, and K. Tachibana. Virtual object manipulation on a table-top AR environment. *Proceedings IEEE and ACM International Symposium on Augmented Reality ISAR 2000*, pages 111–119, 2000.

[10] C. Keskin, F. Kirac, Y. Kara, and L. Akarun. Real time hand pose estimation using depth sensors. In *Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on*, pages 1228 – 1234, Nov. 2011.

[11] V. Lepetit and M. O. Berger. Handling occlusion in augmented reality systems: a semi-automatic method. In *ISAR 2000*, pages 137–46. IEEE.

[12] B. Macintyre, M. Gandy, S. Dow, and J. D. Bolter. Dart: A toolkit for rapid design exploration of augmented reality experiences. In *ACM Symp. on User Interface Software and Technology (UIST04)*, pages 197–206. ACM.

[13] T. Piumsomboon, A. Clark, and M. Billinghurst. Physically-based interaction for tabletop augmented reality using a depth-sensing camera for environment mapping. In *Image and Vision Computing New Zealand (IVCNZ-2011)*, pages 161–166.

[14] A. Rizzo and G. J. Kim. A SWOT analysis of the field of virtual reality rehabilitation and therapy. *Presence: Teleoper. Virtual Environ.*, 14(2):119–146, Apr. 2005.

[15] S. Rusinkiewicz and M. Levoy. Efficient variants of the ICP algorithm. In *3-D Digital Imaging and Modeling, 2001. Proceedings. Third International Conference on*, pages 145–152. IEEE, 2001.

[16] R. B. Rusu and S. Cousins. 3d is here: Point cloud library (pcl). In *IEEE International Conference on Robotics and Automation (ICRA)*.

[17] J. Shotton, A. W. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake. Real-time human pose recognition in parts from single depth images. In *CVPR*, pages 1297–1304. IEEE, 2011.

[18] N. Sugano, H. Kato, and K. Tachibana. The effects of shadow representation of virtual objects in augmented reality. In *Proceedings of the International Symposium on Mixed and Augmented Reality (ISMAR)*. IEEE Computer Society, 2003.

[19] J. Ventura and T. Hollerer. Online environment model estimation for augmented reality. In *Proceedings of the International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 103–106. IEEE Computer Society, 2009.

[20] R. Y. Wang and J. Popović. Real-time hand-tracking with a color glove. In *ACM SIGGRAPH 2009 papers*, SIGGRAPH '09, pages 63:1–63:8, New York, NY, USA, 2009. ACM.

[21] Y. Wang and D. Samaras. Estimation of multiple directional light sources for synthesis of augmented reality images. *Graphical Models*, 65(4):185–205, 2003.

[22] A. D. Wilson. Depth-Sensing Video Cameras for 3D Tangible Tabletop Interaction. *Second Annual IEEE International Workshop on Horizontal Interactive HumanComputer Systems TABLETOP07*, 106(5):201–204, 2007.

[23] J. Zhu, Z. Pan, C. Sun, and W. Chen. Handling occlusions in video-based augmented reality using depth information. *Comput. Animat. Virtual Worlds*, 21(5):509–521, 2010.