RoomAlive: Magical Experiences Enabled by Scalable, Adaptive Projector-Camera Units

Brett Jones[1,2], **Rajinder Sodhi**[1,2], **Michael Murdock**[1,3], **Ravish Mehra**[1,4], **Hrvoje Benko**[1], **Andrew D. Wilson**[1], **Eyal Ofek**[1], **Blair MacIntyre**[1,5], **Nikunj Raghuvanshi**[1], **Lior Shapira**[1]

[1]Microsoft Research, Redmond [2]University of Illinois at Urbana-Champaign
[3]University of Southern California [4]UNC Chapel Hill [5]Georgia Tech

Figure 1. ***RoomAlive*** **is a proof-of-concept system that transforms any room into an immersive, augmented gaming experience. E.g.** ***RoomAlive*** **can change the room's appearance to be (a)** ***Tron*** **themed, or (b) a swampy river. In an example** ***Whack-A-Mole*** **game, players physically (c) whack or (d) shoot moles popping out of their floor, walls and couch. (e) Players control a robot running around the room and shoot enemy robots using a video game controller. (f) An adventure game where players avoid deadly blow dart traps with physical body movement. (*** ***All images in the paper were captured live; with the real-time working prototype*** **).**

ABSTRACT

RoomAlive is a proof-of-concept prototype that transforms any room into an immersive, augmented entertainment experience. Our system enables new interactive projection mapping experiences that dynamically adapts content to any room. Users can touch, shoot, stomp, dodge and steer projected content that seamlessly co-exists with their existing physical environment. The basic building blocks of *RoomAlive* are projector-depth camera units, which can be combined through a scalable, distributed framework. The projector-depth camera units are individually auto-calibrating, self-localizing, and create a unified model of the room with no user intervention. We investigate the design space of gaming experiences that are possible with *RoomAlive* and explore methods for dynamically mapping content based on room layout and user position. Finally we showcase four experience prototypes that demonstrate the novel interactive experiences that are possible with *RoomAlive* and discuss the design challenges of adapting any game to any room.

UIST'14, October 8–11, 2014.

ACM 978-1-4503-2268-3/13/10...$15.00.
http://dx.doi.org/10.1145/2501988.2502032

Author Keywords

Projection mapping; spatial augmented reality; projector-camera system

ACM Classification Keywords

H.5.2 User Interfaces: Graphical user interfaces, Input devices and strategies, Interaction styles.

INTRODUCTION

In recent years the way people play video games changed. We can now interact more naturally with the game world by moving our body (i.e., Nintendo Wiimote, Microsoft Kinect and PlayStation Move). Coupled with new display technologies (e.g., Oculus Rift), we can now feel more *"in the game"* than ever before. However, despite these advances, the game world is still distinctly separate from our real world. We may feel more present in the game world, but the game is not present in our world.

This paper makes a system contribution by demonstrating a novel distributed system that enables a unique set of augmented reality experiences in any room. Our proof-of-concept prototype, *RoomAlive,* transforms any room into an immersive, augmented entertainment experience and enables users to naturally interact with augmented content in their everyday physical environment. RoomAlive builds on a wide array of research fields. Although each individual component of RoomAlive could be improved, the unified system demonstrates the immersive, augmented gaming experiences that might be possible when games can adapt to our room.

Figure 2. Each projector + depth camera unit (a procam unit) consists of a commodity wide field-of-view projector, depth camera and computer.

The basic building block of *RoomAlive* is a combination of a projector and a depth camera (Figure 2), or *procam* unit [28]. By tiling and overlapping multiple procam units, *RoomAlive* is able to cover the room's walls and furniture with input/output pixels. *RoomAlive* tracks users' movements and dynamically adapts the gaming content to the room. Users can touch, shoot, dodge and steer virtual content that seamlessly co-exists with the existing physical environment.

RoomAlive captures and analyses a unified 3D model of the appearance and geometry of the room, identifying planar surfaces like walls and the floor. This is used to adapt the augmented content to the particular room, for instance spawning enemy robots only on the floor of the room. The content also reacts to the user's movement. *RoomAlive* uses a distributed framework for tracking body movement and touch detection using optical-flow based particle tracking [4,15], and pointing using an infrared gun [19].

To demonstrate the scalability of our prototype system, six procam units were used to cover the floor, walls and furniture of a large living room. We explore the design space of whole-room, augmented interactive experiences and the authoring process for these experiences. Through our exploration, we showcase four example experiences that demonstrate the rich and immersive experiences that are possible with *RoomAlive*. Finally, we discuss design guidelines and system limitations for future game designers who wish to create adaptive interactive projection mapped games.

MOTIVATING SCENARIO

Imagine playing a video game in your living room without a television. Instead, the game happens *in* the room, all around you. When the game starts, the room magically transforms into an ancient castle, the walls turn to stone, and flaming torches emerge from the walls casting flickering shadows onto the furniture. Out of the corner of your eye, you see a glowing idol appear on your couch. You walk towards the idol when suddenly, a trap opens on the wall next to you, exposing blow darts ready to fire. You leap out of the way, only to land on the floor face-to-face with a giant cockroach. You quickly get up and jump on the roach. You reach the idol successfully, and a scoreboard drops down showing that you have just scored the best time for the adventure course. This is just one of many interaction scenarios that are made possible by *RoomAlive*.

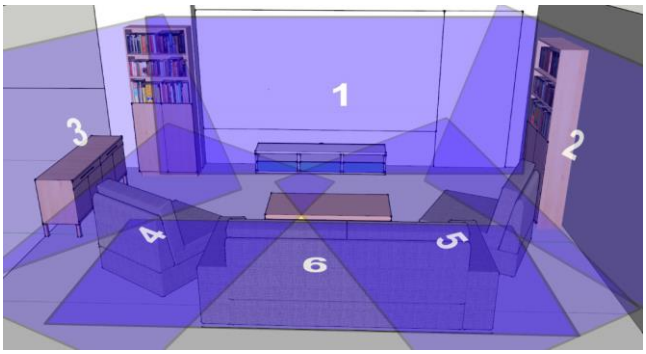

Figure 3. We have installed *RoomAlive* in a large (18 ft x 12 ft) living room using six overlapping *procam* units, with three units pointed towards the three walls of the room, and another three are pointed downwards to cover the floor.

RELATED WORK

Immersive Displays

With gaming experiences, bigger is often better. A larger display has a wider field of view, which makes the user feel more immersed and more present in the experience [16,30]. CAVE systems [10] and tiled displays [7,33] surround the user in visuals, but require special blank projection surfaces. Head mounted displays (HMDs) [26], enable users to be completely immersed in virtual environments, but require each user to wear a device. Optically opaque displays like the Oculus Rift block out the physical environment, and even see-through HMDs often limit the user's field of view (the frame of the glasses). HMDs can also increase the chance of simulator sickness by downgrading the user's perception to the resolution, frame-rate and latency of the display. *RoomAlive* provides unencumbered shared, augmented experience that adapts to the physical environment.

Spatial Augmented Reality

Spatial Augmented Reality (SAR) uses projected light to alter the appearance of physical objects (e.g. [3,5,34,35,36]). Previous work has explored SAR in mobile form factors [14,31,41], steerable projector systems [27,42], and even for novel gaming experiences: e.g. racing a car across a desktop [6], playing chess [44], or playing miniature golf on top of wooden blocks [19]. By using real-time depth sensors, previous work has enabled physical interactions using similar *procam* units [4,43,45].

Most closely related to our work is *IllumiRoom* [18], where a single projector was used to surround a traditional television with projected light to enhance gaming experiences. *IllumiRoom* explored focus plus context visualizations anchored by traditional screen based gaming experiences. *RoomAlive* explores gaming completely untethered from a screen, in a unified, scalable multi-projector system that dynamically adapts gaming content to the room and explores additional methods for physical interaction in the environment.

Commercially, SAR is known as 'projection mapping' and has recently exploded in popularity [22,29]. Projection mapping can create stunning visuals for outdoor advertising,

theater, theme parks, museums and art galleries. However, the projected content is usually passive, and must be laboriously created specifically to the underlying surface. In contrast, with *RoomAlive* users can physically interact with projected content that adapts to any living room environment.

ROOMALIVE SYSTEM

The *RoomAlive* system is comprised of multiple projector-depth camera units or "procam" units. Each unit contains a depth-camera (which includes a color camera, infrared (IR) camera and IR emitter), a commodity wide field-of-view projector, and a computer. A single unit can be used in isolation to create a set-up like IllumiRoom [18], or multiple units can be combined to canvas an entire living room in I/O pixels.

Our current proof-of-concept prototype demonstrates the *RoomAlive* concept in a large living room (18 ft x 12 ft), with six procam units overlapping to cover three walls, the floor and all the furniture in the room (see Figure 3). *RoomAlive* is implemented as a plug-in to the Unity3D commercial game engine. Unity3D enables designers to easily author game content using an intuitive 3D editor and an interactive development environment.

With *RoomAlive* procam units are connected through a distributed client-server model. Each client node is responsible for tracking the players and the room geometry within the view of its own local depth sensor. The clients are also responsible for any un-synchronized local game state (e.g. intensive physics simulations) and for rendering their local view of the global scene, including view dependent rendering. The master server node synchronizes game state and user tracking state across units. The master server node also acts as a client, handling its own game state and rendering.

Procam Hardware

To demonstrate the flexibility of the approach, the procam units are built using a variety of commodity wide field of view projectors (InFocus IN1126, BenQ W770ST, BenQ W1080ST). Each unit also contains a Microsoft Kinect for Windows v1 sensor.

In our prototype, each unit is connected to its own personal computer with a high-end consumer grade graphics card (e.g., NVidia GTX 660 Ti). We use high-end personal computers to explore rich, computationally intensive interactions (e.g., GPU based optical flow) that will be possible in smaller form factors in the future. All procam units are currently mounted to the ceiling of the living room. Procams could also be mounted on the ceiling, tripod or placed on a bookshelf or a coffee table, or generally any location that maximizes coverage and minimizes occlusions.

Automatic Calibration

The *RoomAlive* calibration process is completely automatic, requiring no user intervention and is distributed across multiple procam units. Installers simply mount the procam units with some overlap (~10%) between units. Each procam unit displays a Gray code sequence [2], while all other procam units observe and decode the sequences. This establishes dense correspondences between each projector pixel and all Kinect cameras that can observe that point.

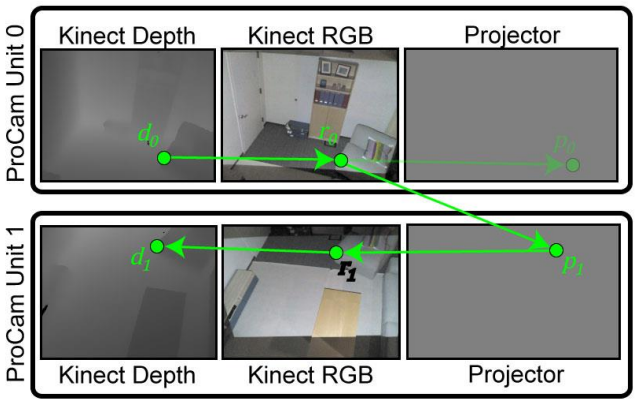

Figure 4. To compute correspondences between two units (*Unit 0* and *Unit 1*), we map a depth pixel in *Unit 0* (d_0), to an RGB pixel in *Unit 0* (r_0). Next, we look up the corresponding projector pixel in *Unit 1* (p_1), via the decoded Gray codes. We then invert the Gray code correspondences to look up the RGB pixel in *Unit 1* (r_1). Finally, we invert the Kinect's transfer map resulting in depth pixel (d_1).

Each correspondence is transformed into the Kinect's depth image via the Kinect SDK, resulting in 2D to 3D point correspondences. Using these correspondences, we solve for the intrinsics and extrinsics of each unit using OpenCV's calibrateCamera function. To increase robustness we embed this in a RANSAC procedure [11]. This process only works for non-planar scenes. A procam unit viewing a planar surface may be temporarily rotated to view a non-planar scene for internal calibration.

To establish global extrinsics (rotation and translation) between units, we chain together correspondences (see Figure 4). A rigid transform is estimated between each pair of procam units using a singular value decomposition, and is refined via pairwise iterative closest point [24]. These transforms are chained together using a maximal spanning tree with weights using the number of inliers for each pairwise transform. The global coordinate system is centered at the first unit to connect to the system, and it is adjusted so gravity points downwards (via the Kinect's accelerometer).

Automatic Scene Analysis

A unified 3D model of the room is formed by combining the depth maps from each unit (on the master node). This model is analyzed, finding continuous planar surfaces (walls, floor) across units and labeling these surfaces. This process must be repeated when the system performs a new scan of the environment, e.g., recalibration or when objects are moved.

The system uses recent techniques in plane and scene model analysis [23]. To find planes, the surface normal is calculated using principal component analysis. The Hough transform [17] is used to select a finite set of planes. Each 3D point and

its surface normal votes for a plane equation parameterized by its azimuth, elevation, and distance from the origin. A greedy strategy is used for associating scene points with planes. Unassigned 3D points that lie in the vicinity of each candidate plane (up to ~10 cm), and has compatible normal direction, are associated with the plane (~0.1° angle). Planes are categorized into 'vertical', 'horizontal' or 'other' based on their orientation with respect to gravity. The 'floor' is identified as the lowest 'horizontal' plane. Within each plane, points that are close together are converted into polygons using the outer hull of the set. Texture coordinates are assigned according to gravity and the principal component axis.

Authoring Augmented Content

RoomAlive makes use of Unity3D's modular plugin framework and scripting interface (Figure 5). Game designers only need to add a single game object to their scene to load the *RoomAlive* plugin. The game designer then has access to the API of *RoomAlive* directly within the scripting interface of Unity3D. *RoomAlive* uses a high resolution 3D model of the room as seen from the current procam unit, and uses Unity3D to render the virtual objects from the projector's point of view. Alternatively, designers could use a stored room model for testing purposes.

Game art assets can be easily imported from external content authoring software and positioned relative to the augmented environment and previewed in situ. Behaviors can then be applied to a game object using a C# scripting interface, e.g., how to react to gun fire. These scripts have access to the *RoomAlive* API, enabling queries regarding scene semantics (e.g., "On floor?") or players' touch collision events.

Mapping Content

One of the key technical challenges in making a projection based experience work in any living room is the placement of virtual game elements. Unlike traditional game design where elements like the game terrain are known *a priori*, *RoomAlive* experiences must operate in multitude of rooms.

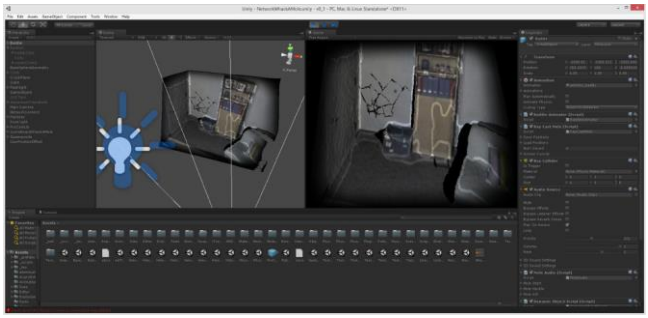

Figure 5. In the Unity3D editor's *Play* mode (debug mode), a high resolution model of the current living room is automatically instantiated, along with virtual cameras (with correct intrinsics/extrinsics) for all the procam units in the scene. On the left is the scene view where artists can drag and drop game assets and view the game from any arbitrary viewpoint. On the right is a live-preview of the image that the projector will display.

The mapping of game elements to a physical environment must be done in real time. Content must be mapped procedurally based on a set of rules that combine the goals of the game designer with the structure of the physical space, which can be significantly complex [39]. While we do not offer a complete solution to this problem, *RoomAlive* employs four techniques for mapping:

- *Random mapping* maps content in a uniformly random way. For example, targets presented to the user can be mapped to random positions around the room.
- *Semantic Mapping* leverages additional semantic information recovered from the scene to map content. For instance, grass and rock game objects could be mapped to only appear in locations on the floor of the room. To accomplish this, the game script queries the *RoomAlive* API on start-up to supply a list of points that belong to the floor of the room, which can then be sampled to instantiate game content.
- *User-constrained mapping* places content based on the current location of the user or user state. For example, a cannon that shoots at a user can be dynamically placed at a location in the room that offers a clear view of the user.
- *User-driven mapping* relies on users to interactively arrange content in the room, spawning content during a gaming experience by touching a physical surface or pointing with a gun at surfaces in the room. This enables users to level-edit or re-decorate their game room.

Tracking User Interaction

RoomAlive supports interacting with augmented content by whole body movement, touching, stomping, pointing/shooting and traditional controller input. For computational efficiency, processing is done locally on each client procam unit, and only resulting changes in game state are synchronized across units.

Proxy Particle Representation

To enable interaction through body movement, touching and stomping, we use the captured depth map along with a real-time physics simulation. Using a similar approach as [4,15], moving objects (e.g., players) in the room are represented by a cloud of spherical '*proxy particles',* that are capable of exerting frictional forces. This enables physically realistic interactions with virtual game objects. For instance, a collision event can be triggered by the proxy particles when the user touches or stomps the physical environment.

The proxy particles are tracked using a depth-aware optical flow algorithm. The 3D flow field pipeline uses a GPU implementation of Brox's algorithm [8] to compute optical flow on the captured 2D depth video, updating the proxy particles to follow moving objects. While flow is typically computed on the RGB image, we do not use the color video, as projected content can lead to incorrect flow results. The computed 2D flow field is re-projected onto the depth data to generate the 3D displacements of the proxy particles.

Gun/Pointing Input

To support pointing at a distance, *RoomAlive* supports a pointing/gun controller. The gun controller uses an integrated IR LED matching the Kinect's infrared band-pass filter. Optics within the gun focus the light, generating a light spot when the player presses the trigger. The light is observed by the IR camera and the target 3D location is recovered.

Traditional Controller Input

In addition to natural user interactions, *RoomAlive* also supports traditional physical game controllers, such as a Microsoft Wireless Xbox controller. This allows users to interact with whole room augmented games using the same input affordances as traditional television gaming experiences. The controller is connected to the server, which distributes user interaction to the clients.

Rendering

Projection mapped content can only physically appear where there is a physical surface. However, virtual content can appear to be at any arbitrary 3D location, by displaying a perspective rendering of the virtual content on the surface, from the view direction of the player.

RoomAlive tracks the player's head position and renders all virtual content with a two-pass *view dependent rendering* [5] (see Figure 6). Content that is aligned with the physical environment can be rendered realistically without a need for view dependent rendering. A unique challenge is the possible presence of multiple users. While multi-user viewpoint rendering remains an open problem [38], RoomAlive uses a simple approach by averaging users' head positions. In practice, 3D scene geometry that strays farther than existing physical surfaces makes *RoomAlive* a single-viewer experience. For scenes where the virtual content is near the physical surfaces, rendering with the average head position offers a good working solution [13].

Another challenge arises from the use of existing room furniture as projection surfaces which may contain a non-white surface albedo. To overcome this problem, a radiometric compensation [32] is applied to compensate the projected image for the color of the surface. This process is limited by the brightness, dynamic range and color space of the projector and some desired surface colors maybe unachievable. For example, a solid red object in our physical environment cannot be made to appear completely green.

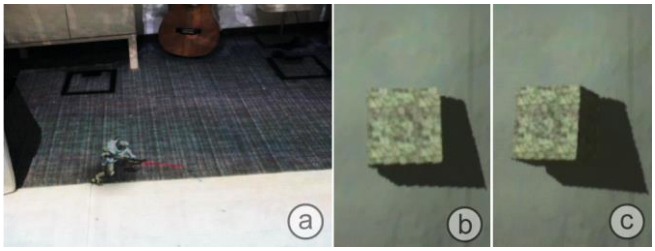

Figure 6. (a) Virtual objects that exist off-surface like this game character are rendered in a view dependent manner with radiometric compensation. (b) A virtual cube placed in front of a wall and viewed straight on. (c) The same cube viewed from a more oblique angle to the right.

Figure 7. Playing room adaptive *Whack-A-Mole* with an IR gun. First, a crack appears to draw the user's attention.

EXAMPLE EXPERIENCES

We envision the *RoomAlive* system supporting a diverse range of applications, limited only by the imagination of game designers. Four example applications were prototyped to demonstrate the potential of whole-room augmented experiences. These experiences were developed in partnership with a game designer, and illustrate the interactive capabilities of the system. These experiences represent a limited survey and are not an exhaustive list. An accompanying video showcases the experiences.

Whack-A-Mole

The *Whack-A-Mole* experience demonstrates a combination of whole body movement, touch and gun/pointing input. Similar to the popular arcade game, in *Whack-A-Mole*, users race to whack, stomp, and shoot moles that randomly appear in the living room. The moles are generated uniformly across the entire room. First, a crack appears on a surface in the room (see Figure 7). Then an audio clip plays, "*I'm over here*", attracting the user's attention followed by a 3D animated mole that emerges from the crack. The mole is rendered from the player's viewpoint and casts appropriate shadows onto the physical room. Figure 1c-d shows a player whacking and shooting a mole on a wall and floor. A virtual scoreboard on the wall counts the player's achievements.

Robot Attack

RoomAlive also supports non-physically realistic gaming mechanics that create entirely new experiences in the living room. *Robot Attack* (based on Unity3D's *Angry Bots*) allows a user to control a virtual character that can run across the walls, floor, chair, etc. (see Figure 8 and Figure 1e). The character is entirely constrained to the surfaces of the living room. The character can move forwards, backwards, left,

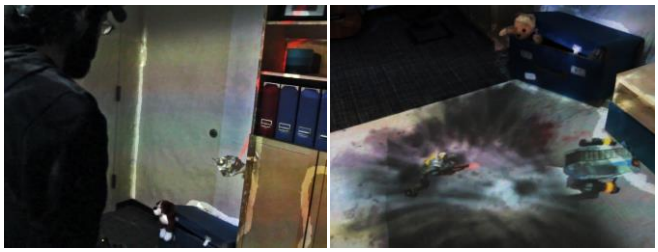

Figure 8. (left) A virtual character runs up the side of a bookshelf. (right) A virtual character fights a robot tank.

right on the surface, but not off of the surface. As the character moves around the living room, surface constrained enemy robots appear, tracking and firing weapons at the virtual character. The character must defend and shoot back against the robots. The surface constrained nature of the experience enables the game characters to walk up walls, adapting their orientation to the normal vector of the surface. Weapons also follow a similar behavior, where 'bullets' remain surface constrained going up and around objects rather than bouncing away. The control of the game is done using a traditional game controller.

Traps

While the previous experiences demonstrate virtual objects near a physical surface, virtual objects can also appear unattached to a physical surface. In *Traps*, a user is surrounded by virtual traps and must navigate the physical environment to avoid being hit by virtual darts. This experience is inspired by many adventure games where players must run, dodge and advance through complex obstacles. If a user navigates through a trigger volume, darts are emitted and collide with physics-enabled proxy particles that represent the user and any dynamically moving object. Because the darts move through open space, they are rendered in view dependent fashion. If the user is hit, blood particles are emitted at the collision location, which follows the user based on proxy particle tracking (see Figure 9).

Setting the Stage

Imagine being able to instantly transform your entire room to match the environment of a video game or film. *RoomAlive* can use projected light to automatically augment the appearance of the entire room, procedurally changing the surface color of objects in the room environment. Figure 1 (a-b) shows two examples of procedurally generated environments. Any Unity3D material, including procedural materials, can be easily assigned to a part of the room geometry with a few clicks. The automatically generated texture coordinates are used for each room polygon. Game designers can specify semantic groups to be given a texture (e.g., *vertical* surfaces). We demonstrate special effects that use 3D scene information, such as rain, or animated insects crawling over floors and tables (Figure 14).

DESIGN CONSIDERATIONS

In order to create experiences for *RoomAlive*, game designers must consider the unique benefits and challenges associated

Figure 9. Blow dart traps pop out of the wall, forcing the user to dodge. If the user is hit, battle damage shows on their body.

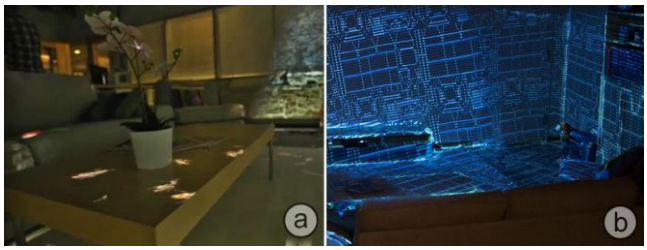

Figure 10. The room is transformed (via the extracted polygons and texture coordinates) into: (a) a cockroach infestation and (b) a futuristic style game room.

with interactive, adaptive, projection-mapped game content. Four critical aspects should be considered when designing room sized augmented experiences.

Input Choice Tradeoffs

RoomAlive supports a variety of input techniques. Some, such as touching or stomping, can enable physical interactions, which move users around the room. However, certain parts of a living room environment may be less suitable for touch-based experiences. For example, an expensive vase or painting may be inappropriate for direct contact. Particularly for children's games, designers may want to limit touch interaction to the floor and walls (large vertical planes). Users could also tag fragile objects removing them from areas of play. Alternatively, the experiences could rely solely on whole body movement, pointing and shooting, or traditional controller input. Pointing and controller input are ideal for controlling objects at a distance, such as a virtual car driving across a wall. We generally found that experiences were most enjoyable when all forms of input were available to users.

Capturing User Attention

When the display surface consists of a large living room environment, directing a user's attention becomes a critical aspect of designing the experience. If a user is on one side of the living room, they may not realize that their attention is needed on the opposite end of the room. Surround sound may be used to direct the user's attention to an approximate location in the room. Also the designer can incorporate a *warm-up* animation into the game design, for instance in *Whack-A-Mole*, a large crack appears seconds before the mole pops out, attracting the user's attention by appearing in their peripheral vision (Figure 7).

Selecting Game Physics

Living room projected experiences enable game designers to create unique interactions that may not be physically realistic. The concept of gravity can be applied globally to all virtual objects in an experience, as is done traditionally in a typical game. Alternatively, gravity can also be applied locally to a virtual object, constraining the object to a surface [19]. For example, a ball rolling on a surface can bounce off the wall or continue rolling up the wall depending on the experience. A local surface adherence model requires a local coordinate system, where each virtual object's local "floor plane" lies tangent to a surface. For instance, a virtual

Figure 11. Calibration errors between units result in ghosting artifacts in projector overlapping regions. (*left*) The character in a single projector, and (*right*) in an overlapping region.

character's "up" direction would be the normal perpendicular to the tangent plane on the surface (see *Robot Attack*).

Controlling Player Movement

In traditional video games, the user is represented as an avatar in a virtual world. The game designer has explicit control over the actions that are possible in the virtual world, including where the user's avatar can walk, what objects they can interact with, etc. In a *RoomAlive* experience, a player's movement and actions are completely controlled by the user, and therefore uncontrollable by the game designer. Imagine a game that involves building physical pillow forts for a virtual snow fight. The game designer cannot stop a user from knocking over another user's pillow fort. A game designer cannot control where a user walks or the surfaces they interact with. Everything is fair game. Therefore, the designer must take care to handle edge cases and ensure that the game mechanics guide the user into a desirable behavior.

SYSTEM LIMITATIONS

While *RoomAlive* enables new and exciting interaction possibilities in the living room, there are several challenges with the current implementation. Foremost are calibration errors caused by non-metric distortions in the raw Kinect data and by deviations from factory calibration settings in the Kinect's own internal calibration. This results in visual errors that causes virtual content to appear *ghosted* in overlapping projector regions (see Figure 11). In practice the overlap regions are small, and we smoothly blend between projector images, so visual artifacts are not as noticeable. In the future, more precise depth estimates of the Microsoft Kinect V2 should reduce the sensing noise and ensure better calibration between procam units. More advanced calibration techniques [1,20,21,37] could also be used to correct for imperfections in calibration.

Interaction with real-time projector camera systems is always impacted by system latency. Off-the-shelf projectors have significant latency, and when combined with latency due to the depth-camera, graphics card and processing, can lead to image ghosting effects where fast moving objects trail behind. Recent research has demonstrated hardware solutions to virtually eliminate latency [25]. In the meantime, designers must balance creating a compelling game without allowing latency to affect the illusion.

Finally, a key challenge in supporting a wide variety of interactions is handling tracking issues that arise from overlapping Kinect sensors. We found the tracking results from the skeleton tracker to be relatively unreliable in overlapping regions, and therefore used the optical flow proxy particle tracking approach. Existing structured light depth cameras can be mechanically augmented to improve noise [9]. New time-of-flight depth sensors (e.g., Microsoft Kinect v2) may have less noise in overlapping regions. Advanced approaches for pose and dynamic object tracking can also be used to improve the *RoomAlive* tracking system [12,40].

CONCLUSION & FUTURE WORK

We have presented *RoomAlive*, a novel system for transforming a room environment into an interactive display. By calibrating and unifying multiple uniquely situated projector camera units, we demonstrate how a distributed rendering approach can be used to seamlessly merge our physical and virtual worlds. Through the development of a range of example experiences, we have shown wide and varied interaction possibilities where *RoomAlive* can be applied in the real world to create truly new experiences.

In the future we will explore approaches and experiences that involve multiple users and multi-user viewpoint rendering. We will also explore new ways to increase immersion with spatialized sound. *RoomAlive* provides a glimpse of how a scalable multi-projector system can transform any room into an immersive augmented gaming experience. Many new and exciting possibilities remain to be explored.

REFERENCES

1. Aliaga, D., Yeung, Y., and Law, A. Fast high-resolution appearance editing using superimposed projections. *ACM TOG*, (2012).
2. Batlle, J., Mouaddib, E., and Salvi, J. Recent progress in coded structured light as a technique to solve correspondence problem: A survey. *Pattern Recognition 31*, 7 (1998), 963–982.
3. Behringer, R. Placing Artificial Objects in Real Scenes. *Proc. of IWAR*, (1998).
4. Benko, H., Jota, R., and Wilson, A. MirageTable: freehand interaction on a projected augmented reality tabletop. *ACM CHI*, (2012).
5. Bimber, O. and Raskar, R. *Spatial augmented reality: Merging real and virtual worlds*. AK Peters Ltd, 2005.
6. Bimber, O. PlayReal: SAR Games. http://140.78.90.140/medien/ar/research.php. 2008.
7. Brown, M., Majumder, A., and Yang, R. Camera-based calibration techniques for seamless multiprojector displays. *IEEE Visualization and Computer Graphics*, (2005).
8. Brox, T., Papenberg, N., and Weickert, J. High Accuracy Optical Flow Estimation Based on a Theory for Warping. *IEEE ECCV*, (2004).
9. Butler, A., Izadi, S., Hilliges, O., Molyneaux, D., Hodges, S., and Kim, D. Shake ' n ' Sense : Reducing Interference for Overlapping Structured Light Depth Cameras. *ACM CHI*, (2012), 1933–1936.

10. Cruz-Neira, C., Sandin, D., and DeFanti, T. Surround-screen projection-based virtual reality: the design and implementation of the CAVE. *ACM SIGGRAPH*, (1993).
11. Fischler, M.A. and Bolles, R.C. Random sample consensus. *Communications of the ACM 24*, 6 (1981), 381–395.
12. Fuchs, H. and Frahm, J.-M. Scanning and tracking dynamic objects with commodity depth cameras. *IEEE ISMAR*, (2013).
13. Hancock, M., Nacenta, M., Gutwin, C., and Carpendale, S. The effects of changing projection geometry on the interpretation of 3D orientation on tabletops. *Proc. of ITS*, (2009).
14. Harrison, C., Benko, H., Wilson, A.D., and Way, O.M. OmniTouch : Wearable multitouch interaction everywhere. *ACM UIST*, (2011).
15. Hilliges, O., Kim, D., Izadi, S., Weiss, M., and Wilson, A. HoloDesk: direct 3d interactions with a situated see-through display. *ACM CHI*, (2012).
16. Hou, J., Nam, Y., Peng, W., and Lee, K.M. Effects of screen size, viewing angle, and players' immersion tendencies on game experience. *Computers in Human Behavior 28*, 2 (2012), 617–623.
17. Illingworth, J. and Kittler, J. A survey of the hough transform. *Computer Vision, Graphics, and Image Processing 44*, 1 (1988), 87–116.
18. Jones, B.R., Benko, H., Ofek, E., and Wilson, A.D. IllumiRoom: peripheral projected illusions for interactive experiences. *ACM CHI*, (2013).
19. Jones, B.R., Sodhi, R., Campbell, R.H., Garnett, G., and Bailey, B.P. Build your world and play in it: Interacting with surface particles on complex objects. *IEEE ISMAR*, (2010).
20. Kainz, B., Hauswiesner, S., Reitmayr, G., et al. OmniKinect : Real-Time Dense Volumetric Data Acquisition and Applications. *IEEE VR*, (2012).
21. Lee, J.C., Dietz, P.H., Maynes-Aminzade, D., Raskar, R., and Hudson, S.E. Automatic projector calibration with embedded light sensors. *ACM UIST*, (2004).
22. Mine, M., Rose, D., Yang, B., Jeroen van Baar, and Grundhöfer, A. Projection-Based Augmented Reality in Disney Theme Parks. *IEEE Computer 45*, 7 (2012), 32–40.
23. Nathan Silberman, Lior Shapira, Ran Gal, P.K. A Contour Completion Model for Augmenting Surface Reconstructions. *ECCV*, (2014).
24. Newcombe, R.A., Davison, A.J., Izadi, S., et al. KinectFusion: Real-time dense surface mapping and tracking. *IEEE ISMAR*, (2011).
25. Ng, A., Lepinski, J., Wigdor, D., Sanders, S., and Dietz, P. Designing for Low-Latency Direct-Touch Input. *ACM UIST*, (2012).
26. OculusVR. http://www.oculusvr.com. 2014.
27. Pinhanez, C. The Everywhere Displays Projector: A Device to create ubiquitous graphical interfaces. *Proc. UbiComp*, (2001), 315–331.
28. ProCams. http://www.procams.org. 2007.
29. Projection Mapping Central. Projection Mapping Central. *http://projection-mapping.org*, 2014.
30. R. Sekular, R.B. Perception 2nd Edition. (1990).
31. Raskar, R., van Baar, J., Beardsley, P., Willwacher, T., Rao, S., and Forlines, C. iLamps: geometrically aware and self-configuring projectors. *ACM TOG 223*, (2003).
32. Raskar, R. and Beardsley, P. A self-correcting projector. *IEEE CVPR*, (2001).
33. Raskar, R., Brown, M.S., Welch, G., Towles, H., Scales, B., and Fuchs, H. Multi-projector displays using camera-based registration. *IEEE Visualization*, (1999).
34. Raskar, R., Welch, G., Cutts, M., Lake, A., Stesin, L., and Fuchs, H. The office of the future: A unified approach to image-based modeling and spatially immersive displays. *ACM SIGGRAPH*, (1998).
35. Raskar, R., Welch, G., and Fuchs, H. Spatially augmented reality. *IWAR*, (1998).
36. Raskar, R., Welch, G., Low, K.L., and Bandyopadhyay, D. Shader lamps: Animating real objects with image-based illumination. *Proc. Eurographics*, (2001).
37. Sajadi, B., Member, S., and Majumder, A. Auto-Calibration of Multi-Projector CAVE-like Immersive Environments. *ACM TOG 17*, 1 (2011), 1–13.
38. Schulze, J.P., Acevedo, D., Mangan, J., Prudhomme, A., Nguyen, P., and Weber, P. Democratizing rendering for multiple viewers in surround VR systems. *IEEE 3DUI*, (2012).
39. Smith, G., Othenin-girard, A., Whitehead, J., and Wardrip-fruin, N. PCG-Based Game Design : Creating Endless Web. *In Proc. of Foundations of Digital Games*, (2012).
40. Straka, M., Hauswiesner, S., Ruther, M., and Bischof, H. Rapid Skin: Estimating the 3D Human Pose and Shape in Real-Time. *IEEE 3DIMPVT*, (2012).
41. Willis, K.D.D., Poupyrev, I., Hudson, S.E., and Mahler, M. SideBySide: Ad-hoc Multi-user Interaction with Handheld Projectors. *ACM CHI*, (2011).
42. Wilson, A., Benko, H., Izadi, S., and Hilliges, O. Steerable augmented reality with the beamatron. *ACM UIST*, (2012).
43. Wilson, A.D. and Benko, H. Combining multiple depth cameras and projectors for interactions on, above and between surfaces. *ACM UIST*, (2010), 273–282.
44. Wilson, A.D. PlayAnywhere: a compact interactive tabletop projection-vision system. *ACM UIST*, (2005).
45. Xiao, R. and Hudson, S.E. WorldKit : Rapid and Easy Creation of Ad-hoc Interactive Applications on Everyday Surfaces. *ACM CHI*, (2013).