VRoamer: Generating On-The-Fly VR Experiences While Walking inside Large, Unknown Real-World Building Environments

Lung-Pan Cheng* Eyal Ofek† Christian Holz‡ Andrew D. Wilson§

Microsoft Research Redmond

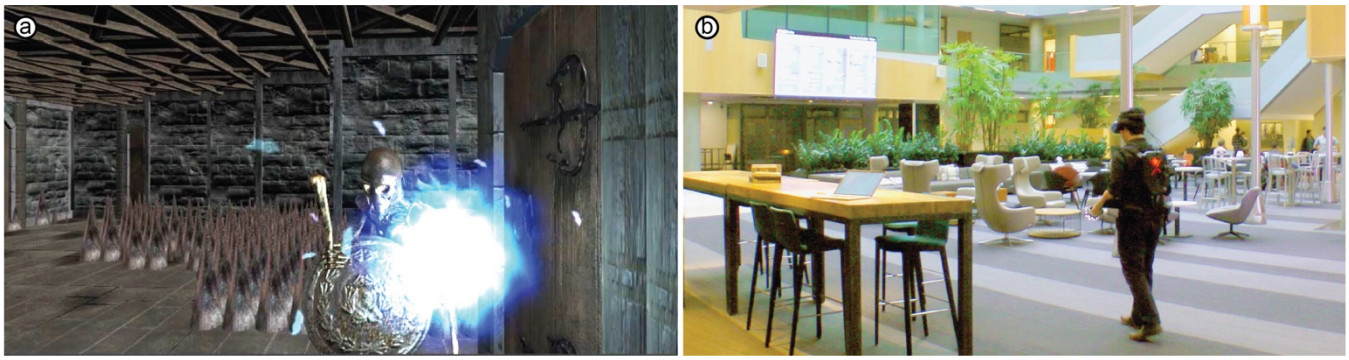

Figure 1: (a) VRoamer dynamically generates a virtual reality (VR) experience such as this dungeon adventure that fits (b) the user's environment, such as this atrium, while the user is walking through building-wide areas. VRoamer requires no manual real-world geometry scanning prior to use and can handle dynamic environments. VRoamer tracks the user's location using inside-out tracking and detects obstacles and walkable areas around the user in real-time using a depth camera to update the VR scene to prevent collisions.

ABSTRACT

Procedural generation in virtual reality (VR) has been used to adapt the virtual world to various indoor environments, fitting different geometries and interiors with virtual environments. However, such applications require that the physical environment be *known* or *pre-scanned* prior to use to then generate the corresponding virtual scene, thus restricting the virtual experience to a controlled space. In this paper, we present VRoamer, which enables users to walk *unseen* physical spaces for which VRoamer procedurally generates a virtual scene on-the-fly. Scaling to the size of office buildings, VRoamer extracts walkable areas and detects physical obstacles in real time, instantiates pre-authored virtual rooms if their sizes fit physically walkable areas or otherwise generates virtual corridors and doors that lead to undiscovered physical areas. The use of these virtual structures allows VRoamer to (1) temporarily block users' passage, thus slowing them down while increasing VRoamer's insight into newly discovered physical areas, (2) prevent users from seeing changes beyond the current virtual scene, and (3) obfuscate the appearance of physical environments. VRoamer animates virtual objects to reflect dynamically discovered changes of the physical environment, such as people walking by or obstacles that become apparent. In our proof-of-concept study, participants were able to walk long distances through a procedurally generated dungeon experience and reported high levels of immersion.

*lungpancheng@ntu.edu.tw
†eyalofek@microsoft.com
‡cholz@microsoft.com
§awilson@microsoft.com

Keywords: Virtual reality, procedural generation, real walking, locomotion techniques, redirected walking.

Index Terms: H.5.1 [Information Interfaces and Presentation]: Multimedia, Information Systems-Virtual Realities.

1 INTRODUCTION

Ever since the classic computer game Rogue was developed in 1980 [2], procedural generation has been used to create experiences on-the-fly. It allows quick generation of a large variety of experiences using a set of pre-defined constraints and has often been applied to generate layouts in space [28]. Recent research has thus used it to adapt real-walking virtual reality (VR) experiences [10] to different indoor environments with different geometries and interiors [3,29].

However, existing work requires the physical environment to be *known* or *scanned* prior to use in order to generate a virtual scene that fits the physical obstacles and thus prevents users from collisions. This need thus restricts VR scenes to controlled areas.

In this paper, we take procedurally generated real-walking VR experiences outside controlled and pre-scanned environments. Our system *VRoamer* generates VR experiences *on-the-fly* when users walk in *previously unseen* indoor environments. VRoamer tracks the environment inside-out using a head-mounted RGBD camera and dynamically generates the virtual scene to reflect physical obstacles, thus allowing the user to roam safely and manually avoid objects in the real world.

1.1 VRoamer use: Walk through VR avoiding objects

Figure 1a shows an example of a generated virtual medieval dungeon using VRoamer. The user wears a head-mounted display (HMD) while walking in a busy office environment (Figure 1b). As the user advances in VR, they see a sprawling dungeon with skeletons and spike traps that spring from the floor. As they navigate the corridors, defeat skeletons, and evade traps in VR, they safely walk through the physical environment without

bumping into physical obstacles. To keep safe, the user simply avoids virtual objects in their way, respecting virtual geometries much like they would physical objects in the real world. While the user is walking in VR, VRoamer constantly adapts the virtual world to the physical space, generating virtual rooms and corridors in front of the user according to sensed physical geometries, placing virtual objects to represent detected obstacles.

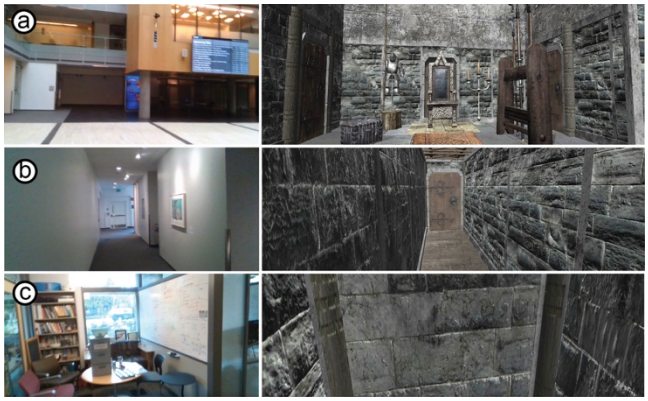

Figure 2: VRoamer (a) places pre-authored rooms when there is a space in the real world, (b) generates corridors to connect to unknown areas, and (c) shows closed virtual rooms when detecting dead-ends.

As shown in Figure 2, if VRoamer finds (a) enough space, it places the pre-authored throne room for the user to find treasures in the virtual experience. If space is insufficient, (b) VRoamer generates corridors that lead the user to undiscovered physical areas, allowing them to keep exploring the dungeon. (c) Finally, if VRoamer detects a physical dead-end, it generates a closed virtual room. The enclosed structures such as rooms and corridors allow VRoamer to (1) slow users down to gain insight into newly discovered physical areas, (2) prevent users from seeing the generation processes beyond the current virtual scene and (3) obfuscate the real-world geometry, enabling rich and thematic virtual experiences.

Finally, VRoamer responds to dynamic objects in the environment such as a pedestrian in front of the user (Figure 3b) by animating spikes to rise from the floor as shown in Figure 3a.

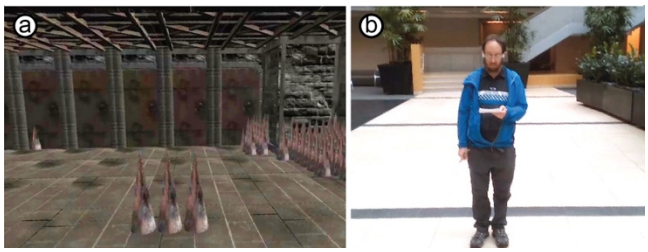

Figure 3: (a) In the middle of the room, spikes dynamically rise from the floor and block the passage, thus (b) preventing the user from colliding with the pedestrian in the real world.

The main contribution of our paper is the wearable VRoamer system that operates in real-time. VRoamer procedurally generates virtual scenes on-the-fly when walking in previously unseen indoor environments. Unlike related work, VRoamer requires no scanning of the environment prior to use. VRoamer also does not replace the user's surroundings with matching geometry, but instead substitutes physical floors, obstacles, and people with virtual mismatching 3D models. We describe the implementation of VRoamer's real-time tracking system and its procedural generation algorithm. We also report our insights and observations from a proof-of-concept study in which participants walked through a virtual dungeon using VRoamer.

2 Related Work

VRoamer relates to locomotion techniques in VR and generating virtual worlds from physical worlds.

2.1 Locomotion Techniques in VR

Real walking is the most natural locomotion technique in VR [10]. A continuous one-to-one mapping of physical to virtual locomotion leads to the highest user satisfaction. The main difficulty of implementing real walking in VR is to prepare a matching physical space. For example, VR arcades, such as The VOID [27], use a predesigned and controlled physical environment for users to physically walk through the virtual experience. Researchers have thus proposed many techniques to approximate real walking even when the given physical space is smaller than the virtual scene.

A range of locomotion devices have been built to simulate real walking while walking in place. The ground surface simulator [36], for example, is a treadmill equipped with individually height-adjustable elements that simulate bumpy terrain and slopes. The torso force feedback system [37] pulls users walking on a treadmill using an active mechanical link, simulating a slope. Gait Master [38] captures the user on every step and uses motion platforms that position themselves where the user is expected to step next. CirculaFloor [39] builds on the same concept but uses four robot units that place themselves under the user's steps.

Redirected walking [11] allows physically walking through virtual scenes larger than the available tracking space by subtly alter the user's walking direction. One of the criteria for ideal redirected walking is imperceptibility [33,40] and it typically requires a large empty space to be effective [20]. Researchers thus have proposed strategies of path planning to reduce rotation gain and achieve a better user experience. Zank et al. [23] proposed planning based on predicting human locomotion and modeling the virtual environment with a skeleton graph. There are also computationally intensive path-planning algorithms [24,25,26] for redirected walking. Hirt et al. [22] used a Google Project Tango tablet to construct a map of the physical environment for existing path planners.

On the other hand, impossible spaces [1] extend the available virtual spaces by employing a self-overlapping architectural layout, allowing users to walk through multiple virtual rooms that share the same physical space. There are mapping algorithms [16,17] that fold large pre-authored virtual scenes to self-overlapping layout. Flexible spaces [18] procedurally generates overlapping virtual rooms [1] to achieve infinite real walking in a confined physical environment. Suma et al. [41] proposed subtly manipulate the virtual architecture by taking advantage of human's inability to detect changes in the invisible environment to achieve redirection. Recent works also explore hiding the existence of other users in the same space by attracting the users away from obstacles [9,15].

2.2 Substitutional Reality

While recent advances in wearable [32] and portable [31] VR enable users to take immersive experiences outside the home into a larger space, the mismatch between physical and virtual environments is a major safety concern: the user may bump into a physical obstacle without seeing it in the virtual world. Therefore, current mobile VR systems typically are used in artificially empty spaces [14]. Researchers have thus examined generating virtual worlds from existing physical scenes.

In Substitutional Reality (SR) [42,43], Simeone et al. conducted a study on how the mismatch between physical and virtual objects can break believability. Reality Skins [47] procedurally generates virtual environments using a set of pre-defined virtual objects to match pre-scanned physical scenes. The substitution process can also be manual [45]. Annexing reality [44] analyzes the environment and opportunistically assigns objects as passive proxies, but a given physical environment may not support all the application needs, and the virtual geometry has to deform to fit existing geometries. Sparse Haptic Proxy [49] extends the ability t adapt virtual environments to physical props by redirecting users' hands. Oasis [3,29] uses a Google Project Tango tablet to build a full model of a physical environment, and then uses the model to procedurally generate a virtual environment the user can walk through. FLARE [8] extracts horizontal and vertical planar surfaces (from SLAM or KinectFusion [7]) and uses them to lay out a set of virtual objects in the real environment according to a set of constraints. Remixed reality [19] brings interactions to AR that are currently only possible in VR, e.g., manipulating time and space. Scenograph [30] procedurally splits a virtual scene into smaller virtual scenes in order to adapt to a smaller physical space.

2.3 Summary

All aforementioned works require scene understanding to be performed beforehand, limiting the application of the system to static, controlled, or empty physical environments. In contrast, VRoamer keeps track of the user's surroundings in real time and generates virtual scenes on the fly that guide the user walking into undiscovered and uncontrolled indoor areas.

While VRoamer is built upon the concepts of SR [42,43] and flexible spaces [18], VRoamer overcomes each of their shortcomings using their strengths. In SR, the acceptable mismatch between physical and virtual objects still reveals the real environment and may break the immersion of the virtual experience. VRoamer does not reveal the real environment but procedurally generates thematic virtual scenes according to virtual narratives: in some places it generates a pre-authored room while in-between it generates transitions geometry that reflects only the walkable surfaces in the real world. In flexible spaces, the given physical space is static and does not take dynamic objects into consideration. In contrast, VRoamer reflects changes of dynamic objects as well as available spaces in the virtual experience on the fly. The blend of these two concepts allows VRoamer to open up new possibilities for real walking VR.

3 Substitution: Rooms & Corridors vs. Generic Objects

VRoamer's key components to generate VR experiences on-the-fly while the user is walking in unknown environments are *rooms* and *corridors*. Our goal is to generate immersive experiences on they fly. While generating virtual generic objects that closely matches real world obstacles utilizes open spaces in the real world well, it comes with the drawback of revealing the geometry of the real world around the user, which may reduce immersion as users are reminded of real-world characteristics in the VR environment. Our generated virtual corridors and rooms follows thematic design constraints and are geometrically different than the real environment around the user.

Another benefit of rooms and winding corridors is controlling the user's visibility. The buildup of virtual geometry is guided by the system knowledge of up-to-date real-world geometry. The tracking system has limited sensing range (distance, field of view, visibility) and the current tracking data may only have partial information about physical obstacles. Areas that have not been scanned by the system yet may be hidden from the user's view to avoid unnatural visual updates that may break the user's immersion. Additional mechanisms that limit the user's visibility include lighting effects (e.g., fog or sparks), occluders (e.g., characters, geometry), and distractions of the user's view.

Finally, rooms and corridors slow down users and delay the system's need to commit to either the next room or corridor until the user opens the door. This allows the tracking system to gain more insight into newly discovered physical areas that are beyond the current virtual scene.

While rooms and corridors are the key components to generate immersive experiences on the fly, VRoamer incorporates the use of generic objects to quickly react to unexpected events within the user's field of view. In our dungeon experience, VRoamer uses animated spikes that can pop quickly from the floor to control the user's access to newly discovered obstacles. This is just one example to a range of scripted events that can be used to dynamically control the user movement. Alternatives include ceiling fragments that are falling down and characters that are in motion, which may adopt dynamic speeds to approach an obstacle's location.

Some virtual objects introduced above merely serve the purpose of supportive elements, such as the ghost skeletons in Figure 1a. Unlike safety elements such as the spikes and walls, they compel the user to advance in the virtual environment.

4 Implementation

Figure 4 shows an overview of our system's hardware equipment. VRoamer uses an HP Omen X backpack computer, equipped with an NVIDA GTX 1080 graphics card, to drive the HMD and process depth images. Two external batteries power the system at full performance for about an hour. The user wears an Acer Mixed Reality headset [12] and carries two hand-held controllers. Windows Mixed Reality delivers the camera pose as well as the position and the orientation of the two controllers. We configure the headset for a "seated setup", which uses no fixed definition for the user's room. Drift may still occur, empirically 50 cm for each 50 m walked (1%).

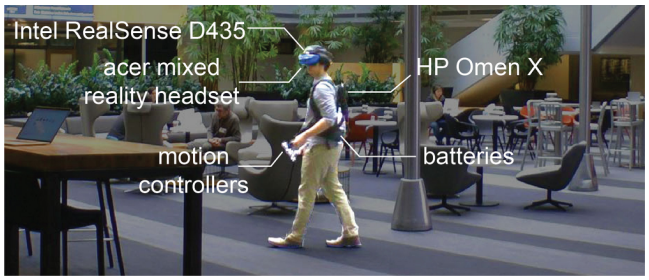

Figure 4: The equipment involved in VRoamer's operation.

An Intel RealSense D435 RGBD camera [5] is rigidly mounted on the top of the HMD, aimed at a parallel view direction to the HMD. The D435 is a stereo camera that provides depth images with 1280 x 720 resolution at 30 fps. It uses projected infrared light patterns to add details to texture-poor objects such as indoor walls and floors. The sensing range of the depth camera is 30 cm to 10 m with a FOV of 91.2° x 65.5° (horizontal x vertical). The power of the active projection is set to maximum. We calibrated the D435 with the Mixed Reality headset and aligned the camera's coordinate system to that of the headset (reprojection error: 4.93 pixels). The motion delay between the depth camera and the MR headset was measured to be about 60 ms. The position of geometry derived from the depth images is calculated by interpolating camera position and orientation over time.

The software stack of our system is implemented in Unity [4]. We match the user's motion to an animated avatar inside Unity using built-in inverse kinematic functions. At the start of

operation, the system determines the user's height by estimating the distance to the floor while the user looks down.

4.1 Sensing the Physical Environment

Figure 5 illustrates VRoamer's real-time processing pipeline. As the user walks, the inside-out tracking HMD reports the user position and orientation while the HMD's depth camera transmits all depth frames to the PC in which they are processed to update the VR world seen by the user. The system maintains a map of the user's surroundings that, in contrast to many SLAM systems, does not assume temporal consistency, but rather overwrites prior knowledge with newer captures.

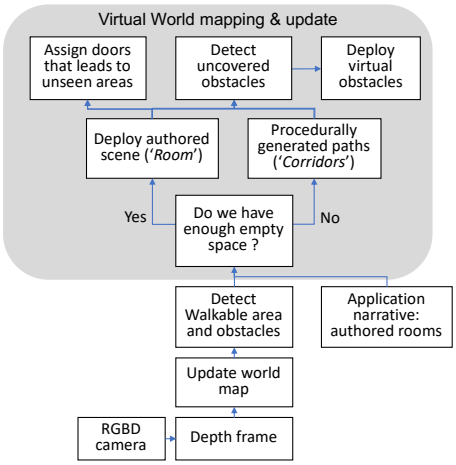

Figure 5: VRoamer's real-time processing pipeline, starting with captured depth frames and inside-out tracking from the HMD and finishing by displaying a view of the virtual world.

Influenced by Oasis [3], VRoamer's pipeline assumes a flat floor and detects the visible part of the floor as a *walkable area*. Any object that lies at a significant height above or below this floor is classified as an *obstacle*.

The key difference is that any location that is not yet classified is considered *unknown*. We store a dynamic 2D map of the world representing both the discovered floor and the obstacles. VRoamer then generates the virtual environment based on the map and then displays the virtual environment to the user from their point of view.

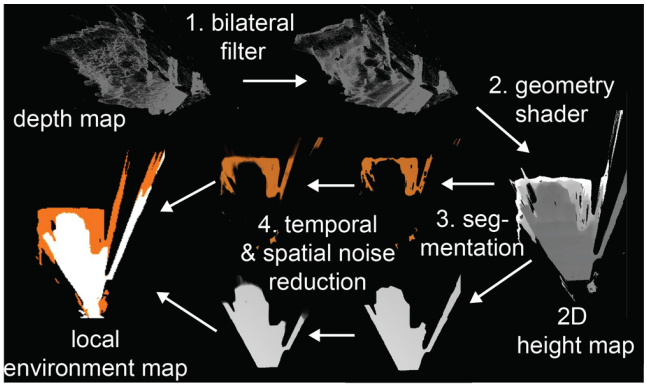

Figure 6: Every depth frame captured, is processed to generate a current local model of the physical environment in front of the user.

VRoamer regenerates the virtual room each time the user revisits the same physical place. This approach overcomes the mismatch between the virtual world and the physical space due to (1) inevitable drift of our inside-out tracking system that can accumulate to noticeable levels after walking a long distance and (2) dynamically changing physical environments. If the physical space is not available, the room is generated at a new available physical location.

To achieve real-time environment representation, our system processes the depth image on the GPU. We implemented custom compute shaders and geometry shaders to implement all steps described below. Our system renders an average of 80 fps (the sensing update rate is limited by the depth camera at 30 fps).

As shown in Figure 6, we process the depth image in 4 steps. Parameter settings for each step were determined empirically.

(1) Unsigned 16-bit depth frames are streamed into Unity from the depth camera, and a GPU Bilateral Grid Filter [7] ($\sigma_s = 32$, $\sigma_r = 0.07$) is applied to smooth out noise.

(2) Given depth camera intrinsic parameters (focal length, etc.) and extrinsic pose given by the inside-out tracking, depth maps are transformed into a 3D polygonal surface. Extra-long polygons (inner angle of less than 0.8 degrees) generated by outlier depth points are discarded, as well as depth points above the user's height (2 m). The polygonal surface is rendered as a 2D horizontal height map, colored by gray level proportional to the surface elevation. The size and resolution of the map is adjustable. We use 1024 x 1024 with 0.01 m resolution for a total area of 10 m by 10 m.

(3) The height map is segmented using estimated floor height to *Floor* pixels (white) and *Obstacle* pixels (orange). A threshold of 20 cm was used to detect obstacles and avoid noise of floor pixels. In contrast to Oasis [4] which models the environment as a pre-process, VRoamer has *Unknown* pixels (black) that have yet to be observed.

(4) The floor and areas with obstacles are further filtered temporally using an exponential filter (ratio = 0.2) and spatially using morphological opening filters (3x3 square structure element) to remove isolated holes. The filtered results are joined to form the map of the physical environment ahead of the user.

4.2 Physical Environment Model Update

To deal with the user's constant-moving reference frame and manage the history of the partially sensed environment, VRoamer maintains *local* maps and a unified *global* map.

The frame-based local maps represent partial views of the physical environment in front of the user (Figure 7). These local maps are sensed each frame and then recorded to the global map.

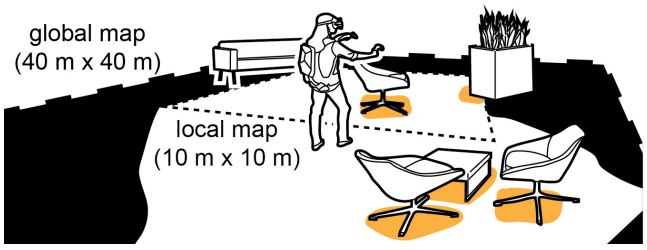

Figure 7: VRoamer updates the local map each frame while maintaining the global map.

Unlike the local map, the global map is an axis-aligned grid in world coordinates. The global map is a wide area centered around the user's location. For our experiments, we used a 1024 x 1024-pixel map with 0.04 m resolution for a total size of 40 m x 40 m. Since we do not want to limit the size of the physical environment explored by the user, the global map also advances with the user.

To maintain the history of sensed area, the global map does not move with the user every frame. Instead, as shown in Figure 8, the global map advances only when the user exceeds a certain boundary (2 m x 2 m). The size of the boundary is adjustable. Our system swaps the global map to a new position while copying the overlapped history. The smaller the boundary the more frequent the swapping operation and more computational power for copying. There is a deadband (50 cm) in between boundaries to prevent flickering between the new and the previous map positions.

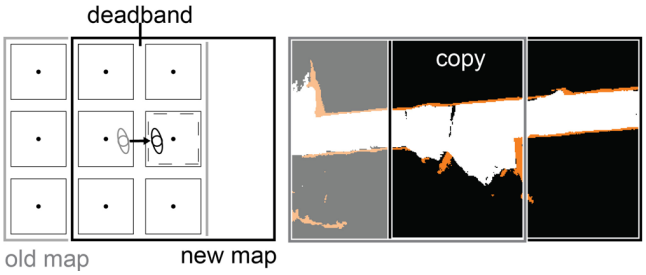

Figure 8: The global map advances only when the user crosses a boundary, reducing computational overhead. The system copies the pixel history from the previous position.

4.3 Generating Virtual Environments

VRoamer constructs the virtual world in three steps:

Step 1. Mask Map, Label Portals and Find Paths: Figure 9 shows the process. VRoamer takes a snapshot of the global map in front of the user (Figure 9a). The walkable area is eroded by 50 cm, which enlarges all sensed obstacles and creates a safety margin. Edges between the walkable area (white) and the large unknown area (black) are marked as *portal* (blue), leading out of the scanned area (Figure 9b). For generating virtual environments, we downsample the map to 64 x 64 and calculate the distance map from the current door locations to all the other door locations. Shortest paths from the current portal locations to all the other (Figure 9c).

Figure 9: VRoamer pre-processes the global map by (a) taking a snapshot, (b) labeling boundaries between walkable (white) and unknown (black) areas as portal (blue) and (c) finding shortest paths (green, selected for clarity) to those portals.

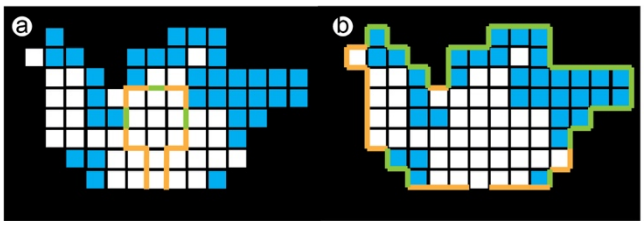

Figure 10: VRoamer generates (a) a 3px-wide pre-authored room when the system finds enough empty space in the front or (b) a procedurally generated corridor that leads to all portals found in the map. The system places the wall (orange lines) and the doors (green lines) on the boundary of pixels.

Step 2. Find Space for Pre-Authored Rooms: Figure 10a shows an example when the system finds that the empty space in front of the user is large enough to hold a virtual room that is pre-authored, i.e., created and modeled during design time for opportunistic use. VRoamer finds the maximal empty rectangle in the map based on the algorithm from Orlowski [48]. VRoamer then lays out that room and creates a corridor that leads from the current user position to the entrance of the pre-authored room. Additional doors are added if there are paths that pass by side walls.

Step 3. Automatic Corridor Generation: Figure 10b shows an example when the system finds that the empty space ahead of the user cannot fit a pre-authored room. VRoamer generates a corridor: a room that starts at the user's position and contains walkable paths to all portals marked in stage 1. That is, there could be more than one door in the generated corridor, and every door is accessible. The size of the generated corridor depends on the size of the walkable area in front of the user.

Step 4. Animating Objects: VRoamer reads the pixel value of the up-to-date local map and animate simple geometry such as bricks or spikes using the geometry shader. For example, when the pixel changes from *Obstacle* to *Floor*, the spike on that position is animated downward and disappear eventually.

For simplicity, our current implementation only generates 1.28m (2px) width virtual elements (e.g., walls, floors) and aligned to a 1.28 m (2px) grid.

All geometry creation is processed on the CPU and is triggered occasionally (e.g. when the user opens the virtual door). The computation is done in a separate thread over multiple frames, preventing slowing down the main thread of the system.

5 Proof-of-Concept Study

We conducted a preliminary study to validate the functionality of VRoamer and to gain insights from the participants. We tested in our uncontrolled building environment and had an experimenter accompanying the participants while walking for safety reason.

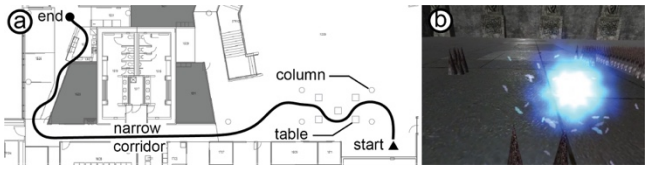

Figure 11: Participants walked (a) the indicated path inside a previously unseen office environment that VRoamer detected, tracked, and generated a virtual dungeon experience for on-the-fly. (b) Participants' task was to follow the blue fairy target in VR while evading VR obstacles that appeared.

5.1 Task

Participants' task was to follow the flying virtual fairy (Figure 11b) and, while walking, evade virtual obstacles in the generated dungeon. All parameters for generating the virtual environment were the same as described in the implementation section. The parameters were chosen empirically following a pilot study. Participants used VRoamer, wearing the VR headset and backpack. The experimenter controlled the motion of the flying target by moving a position-tracked controller in front of the participant along a pre-planned path (as shown in Figure 11). To test our tracking system, the target flew through a busy atrium and obstacles such as columns, tables, and then into a narrow corridor in order. While the path was fixed, the generated dungeons varied

between the participants as the physical environment involved pedestrians.

Even though VRoamer would have supported arbitrary paths through the building, we chose a fixed path during the study because it equalized total walking distances across participants and produced objective and comparable measurements.

5.2 Procedure

The experimenter briefly explained the task to each participant and then assisted them in putting on the backpack and headset to blind them. The experimenter then guided the participant to the starting point, detouring about 5m (involving three turns in place) for obfuscation. Participants' head movements, sensed depth images, and the generated virtual scene were recorded for post-hoc review. After completing the task, participants filled out two questionnaires: the Presence Questionnaire [21] and another questionnaire designed to assess where they thought they were located in the building when they finished.

5.3 Participants

We recruited 14 participants (4 female, ages 25–56, M = 38.1, SD = 10.5) from our institution, including administrative staff and students, 2 out of the 14 had no VR experience before, the other participants reported limited experience. None of the authors or people familiar with the project participated. Participants were familiar with the general layout of the building and received a small gratuity afterwards.

5.4 Results

Participants' mean walking speed was 0.74 m/s (SD = 0.09), traveling a mean total distance of 66.23 meters (SD = 3.35). This includes potential errors in tracking and any walking that occurred during the instructions. 10 participants stopped before entering a narrow VR corridor as the physical path became narrow. Only one participant collided with an obstacle when side-stepping.

Participants' mean presence score was 105.21 (SD = 0.53). None of the participants correctly guessed their final location inside the building. Five participants used clues such as ambient sounds, smells, or temperature to make a rough guess.

5.5 Discussion

The quantitative and qualitative measures reported above indicate that VRoamer immersed participants in VR while walking, with a high mean presence score of 5.5/7. Participants followed the target smoothly and did not stumble, all while evading the dynamic spikes in VR (and thus real-world obstacles). Participants' comments showed that they did not expect dynamically changing environments. P1 said "I was very confident until I saw new spikes suddenly appearing." The dynamic changes that prevented participants from collisions might have simultaneously impacted their confidence in their ability to navigate the world quickly. Other participants' comments included "I did not realize that I went this far" (P3), "This was already a fun experience" (P10).

The evaluation also showed some limitations of our current implementation. For example, P8 collided with a table while side stepping, explaining that the field of view of the HMD prevented him from seeing obstacles to his side. In addition to the limited display, currently VRoamer capture 3D structures only within the user viewing frustum. We expect that upcoming HMDs with wider FOV will allow VRoamer to display the space more comprehensively.

6 Limitations

VRoamer demonstrates the generation of immersive virtual worlds with no prior knowledge of the geometry of the real physical world. This process is of course limited by the capability of the sensors: The longer the range of the sensors, the less the need to slow down the user. In addition, objects such as transparent and reflective surfaces may not be sensed accurately by depth cameras. Employing non-optical sensing techniques such as sonar may allow the system to better model a physical environment and would complement our current approach.

Regardless of the sensors, we can expect that some areas may not be sensed at all. In real life, people tend to take risks such as ignoring unseen areas just behind corners or narrow paths, turning a corner and not expecting to encounter an obstacle. To ensure the safety of the user, VRoamer does not take such risks. It does not show likely paths that have not been sensed yet. As a result, some physical areas may be difficult to reach. We hope to look for more ways to establish a real-world geometry in the future, for example by dynamically integrating building floor plans.

As mentioned in the introduction, VRoamer, like other real-walking systems, assumes that users respect virtual geometries, i.e., avoiding the spikes on the floor and not walking through walls. Users are instructed accordingly before use. Experience designers can refer to Simeone et al.'s design guidelines [46] to manipulate movement when designing virtual scenes.

Backtracking the virtual experience is not guaranteed in VRoamer as virtual rooms are regenerated each time the user revisits the same physical place. If the physical space is available, the same room is generated. Otherwise, the room is generated at a new available physical location. However, backtracking can be avoided by preventing the user from repeating their steps exactly. For example, a door may not open in the back direction, forcing the user to go through a different corridor. For some experiences that backtracking is necessary, the regeneration can be turned off, and VRoamer then falls back to use animating objects to represent any physical change in the previous room.

7 Concepts for generalizing VRoamer and future work

In this paper, we have focused on generating indoor open-ended virtual environments as we explained our design considerations that are based on a practical point of view of our current prototype in the Section 3. In this section we speculate concepts for generalizing VRoamer to other contexts.

One of the usual contexts is outdoor virtual experiences, such as playing golf or jogging. Since outdoor scenes usually contain no obstacles that block the user's vision, the major problem would be naturally bringing in new virtual objects. There are many more possible mechanisms in the spectrum between hidden objects such as spikes that react quickly to a sensed obstacle and movable virtual scene objects that incur some delays. Figure 12 shows such an example: Ghost skeleton knights materialize when VRoamer has discovered a physical obstacle and moves to block the user's passage and maintain safe navigation.

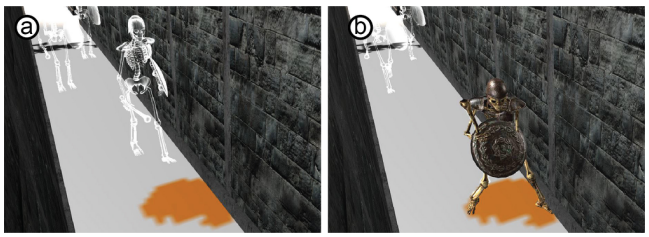

Figure 12: Virtual skeleton knights cause the user to avoid a newly detected physical obstacle (orange pixels) by moving to block the user's path.

We are also looking into incorporating VRoamer with strategical games such as Portal. However, as we stated in the previous section, backtracking is one of the VRoamer's limitation and this type of games usually requires backtracking to solve puzzles in previous rooms. In the future, we attempt to combine redirected walking to redirect the user for backtracking when the previous space is occupied.

Another use case is to walk a virtual art gallery. A future system could also include physical object recognition to represent real-world constraints and semantics in VR, such as object sturdiness (e.g., walls, doors). Recognizing such objects based on RGB would also improve our tracking and obstacle representation, e.g., by detecting tripping features such as carpets.

8 Conclusion

We presented VRoamer, a system that generates VR experiences *on-the-fly* when users walk in *previously unseen* large indoor environments. VRoamer brings procedural generated VR experience from a known, controlled space to an unknown, uncontrolled space. In contrast to redirected walking techniques that fold virtual spaces, VRoamer extends the use of physical spaces. We have explored design considerations when generating virtual content on the fly with a real-time inside-out tracking system. With VRoamer, we speculate on bringing more physical spaces for the use of immersive experiences.

References

[1] E.A. Suma, Z. Lipps, S. Finkelstein, D. `M. Krum and M. Bolas, "Impossible Spaces: Maximizing Natural Walking in Virtual Environments with Self-Overlapping Architecture," *IEEE Trans. Vis. Comput. Graphs.,* vol. 18, no. 4 pp. 555-564, 2012.

[2] G. R. Wichman, "A Brief History of Rogue," *http://www.digital-eel.com/deep/A_Brief_History_of_Rogue.htm*, 1997.

[3] M. Sra, S. Garrido-Jurado, C. Schmandt and P. Maes, "Procedurally generated virtual reality from 3D reconstructed physical space," in *Proceedings of the 22nd ACM Conference on Virtual Reality Software and Technology*, 2016, pp. 191-200.

[4] Unity, "Unity – Game Engine," *unity3d.com*, 2018.

[5] Intel Corporation. "Intel RealSense," *realsense.intel.com*, 2018.

[6] J. Chen, S. Paris and F. Durand, "Real-time edge-aware image processing with the bilateral grid," in *ACM SIGGRAPH 2007 Papers*, 2007.

[7] S. Izadi, D. Kim, O. Hilliges, D. Molyneaux, R. Newcombe, P. Kohli, J. Shotton, S. Hodges, D. Freeman, A. Davison and A. Fitzgibbon, "KinectFusion: real-time 3D reconstruction and interaction using a moving depth camera," in *Proceedings of the 24th annual ACM symposium on User interface software and technology*, pp. 559-568, 2011.

[8] R. Gal, L. Shapira, E. Ofek and P. Kohli. "FLARE: Fast layout for augmented reality applications," in *Proceedings of the 13th IEEE International Symposium on Mixed and Augmented Reality*, pp. 207-212, 2014.

[9] S. Marwecki, M. Brehm, L. Wagner, L. Cheng, F. Mueller and P. Baudisch, "VirtualSpace - Overloading Physical Space with Multiple Virtual Reality Users," in *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, p. 241:1--241:10, 2018.

[10] M. Usoh, K. Arthur, M. C. Whitton, R. Bastos, A. Steed, M. Slater, and F, P. Brooks, Jr., "Walking > walking-in-place > flying, in virtual environments," in *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, pp. 359-364, 1999.

[11] S. Razzaque, "Redirected Walking" *Ph.D. Dissertation*, University of North Carolina at Chapel Hill, Chapel Hill, NC, USA. Advisor(s) Fredrick P. Brooks, Jr., 2005.

[12] Microsoft Corporation. "Windows Mixed Reality,". https://developer.microsoft.com/en-us/windows/mixed-reality, 2018.

[13] M. Serino, K. Cordrey, L. McLaughlin, and R. L. Milanaik, "Pokémon Go and augmented virtual reality games: A cautionary commentary for parents and pediatricians," *Current Opinion in Pediatrics*. 2016.

[14] D. Waller, E. R. Bachmann, E. Hodgson, and A. C Beall, "The HIVE: A huge immersive virtual environment for research in spatial cognition," *Behav. Res. Methods*, vol. 39, pp. 835–843, 2007.

[15] H. Chen and H. Fuchs, "Supporting Free Walking in a Large Virtual Environment: Imperceptible Redirected Walking with an Immersive Distractor," in *Proceedings of the Computer Graphics International Conference*, 2017, p. 22:1--22:6.

[16] Z.-C. Dong, X.-M. Fu, C. Zhang, K. Wu, and L. Liu, "Smooth assembled mappings for large-scale real walking," *ACM Trans. Graph.*, 2017.

[17] Q. Sun, L.-Y. Wei, and A. Kaufman, "Mapping virtual and physical reality," *ACM Trans. Graph.*, 2016.

[18] K. Vasylevska, H. Kaufmann, M. Bolas, and E. A. Suma, "Flexible spaces: Dynamic layout generation for infinite walking in virtual environments," in *IEEE Symposium on 3D User Interface 2013*, 2013.

[19] D. Lindlbauer and A. D. Wilson, "Remixed Reality: Manipulating Space and Time in Augmented Reality," in *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, p. 129:1--129:13, 2018.

[20] F. Steinicke, G. Bruder, J. Jerald, H. Frenz, and M. Lappe, "Estimation of detection thresholds for redirected walking techniques," *IEEE Trans. Vis. Comput. Graph.*, 2010.

[21] B. G. Witmer and M. J. Singer, "Measuring presence in virtual environments: A presence questionnaire," *Presence Teleoperators Virtual Environ.*, 1998.

[22] C. Hirt, M. Zank, and A. Kunz, "Preliminary Environment Mapping for Redirected Walking," in *Proceedings of 25th IEEE Conference on Virtual Reality and 3D User Interfaces*, 2018.

[23] M. Zank and A. Kunz, "Optimized graph extraction and locomotion prediction for redirected walking," *2017 IEEE Symposium on 3D User Interfaces* (3DUI), pp. 120-129, 2017.

[24] T. Nescher, M. Zank and A. Kunz, "Simultaneous mapping and redirected walking for ad hoc free walking in virtual environments," *2016 IEEE Virtual Reality* (VR), pp. 239-240.

[25] C. Hirt, M. Zank, and A. Kunz. "Geometry Extraction for Ad Hoc Redirected Walking Using a SLAM Device". In *Augmented Reality, Virtual Reality, and Computer Graphics*, pp. 35–53, 2018.

[26] C. Hirt, M. Zank, and A. Kunz. "Real-time wall outline extraction for redirected walking," in *Proceedings of the 23rd ACM Symposium on Virtual Reality Software and Technology*, p. 72:1--72;2, 2017.

[27] The Void, "The Void", *thevoid.com*, 2018.

[28] M. Hendrikx, S. Meijer, J. Van Der Velden, and A. Iosup, "Procedural Content Generation for Games: A Survey," *ACM Trans. Multimed. Comput. Commun. Appl.*, vol. 9, no. 1, p. 1:1--1:22, Feb. 2013.

[29] M. Sra, S. Garrido-Jurado and P. Maes, "Oasis: Procedurally Generated Social Virtual Spaces from 3D Scanned Real Spaces," in *IEEE Transactions on Visualization and Computer Graphics*, vol. 24, no. 12, pp. 3174-3187, 1 Dec. 2018.

[30] S. Marwecki and P. Baudisch, "Scenograph: Fitting Real-Walking VR Experiences into Various Tracking Volumes," in *Proceedings of the 31st Annual ACM Symposium on User Interface Software and Technology*, 2018, pp. 511–520.

[31] E. Hodgson, E. R. Bachmann, D. Vincent, M. Zmuda, D. Waller, and J. Calusdian, "WeaVR: a self-contained and wearable immersive virtual environment simulation system," *Behav. Res. Methods*, vol. 47, no. 1, pp. 296–307, Mar. 2015.

[32] E. R. Bachmann, M. Zmuda, J. Calusdian, X. Yun, E. Hodgson, and D. Waller, "Going anywhere anywhere: Creating a low cost portable immersive VE system," in *2012 17th International Conference on Computer Games (CGAMES)*, 2012, pp. 108–115.

[33] N. C. Nilsson, T. Peck, G. Bruder, E. Hodgson, S. Serafin, M. Whitton, F. Steinicke, and E. S. Rosenberg, “15 Years of Research on Redirected Walking in Immersive Virtual Environments,” *IEEE Comput. Graph. Appl.*, vol. 38, no. 2, pp. 44–56, Mar. 2018.

[34] J. N. Templeman, P. S. Denbrook, and L. E. Sibert, “Virtual Locomotion: Walking in Place Through Virtual Environments,” *Presence: Teleoper. Virtual Environ.*, vol. 8, no. 6, pp. 598–617, Dec. 1999.

[35] E. Bozgeyikli, A. Raij, S. Katkoori, and R. Dubey, “Point &Teleport Locomotion Technique for Virtual Reality,” in *Proceedings of the 2016 Annual Symposium on Computer-Human Interaction in Play*, 2016, pp. 205–216.

[36] H. Noma, T. Sugihara and T. Miyasato, "Development of Ground Surface Simulator for Tel-E-Merge system," *Proceedings IEEE Virtual Reality 2000* (Cat. No.00CB37048), New Brunswick, NJ, USA, 2000, pp. 217-224.

[37] J. M. Hollerbach, R. Mills, D. Tristano, R. R. Christensen, W. B. Thompson, and Y. Xu, “Torso Force Feedback Realistically Simulates Slope on Treadmill-Style Locomotion Interfaces,” *Int. J. Rob. Res.*, vol. 20, no. 12, pp. 939–952, 2001.

[38] H. Iwata, H. Yano and F. Nakaizumi, "Gait Master: a versatile locomotion interface for uneven virtual terrain," *Proceedings IEEE Virtual Reality 2001*, Yokohama, Japan, 2001, pp. 131-137.

[39] H. Iwata, H. Yano, H. Fukushima, and H. Noma, “CirculaFloor: A Locomotion Interface Using Circulation of Movable Tiles,” in *Proceedings of the 2005 IEEE Conference 2005 on Virtual Reality*, 2005, pp. 223–230.

[40] N. Nilsson, S. Serafin, F. Steinicke, and R. Nordahl, “Natural Walking in Virtual Reality: A Review,” *Comput. Entertain.*, vol. 16, pp. 1–22, 2018.

[41] E. A. Suma, S. Clark, D. Krum, S. Finkelstein, M. Bolas and Z. Warte, "Leveraging change blindness for redirection in virtual environments," *2011 IEEE Virtual Reality Conference*, Singapore, 2011, pp. 159-166.

[42] A. L. Simeone, E. Velloso, and H. Gellersen, “Substitutional Reality: Using the Physical Environment to Design Virtual Reality Experiences,” in *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, 2015, pp. 3307–3316.

[43] A. L. Simeone, "Substitutional reality: Towards a research agenda," *2015 IEEE 1st Workshop on Everyday Virtual Reality* (WEVR), Arles, 2015, pp. 19-22.

[44] A. Hettiarachchi and D. Wigdor, “Annexing Reality: Enabling Opportunistic Use of Everyday Objects As Tangible Proxies in Augmented Reality,” in *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, 2016, pp. 1957–1967.

[45] J. F. Garcia, A. L. Simeone, M. Higgins, W. Powell, and V. Powell, “Inside Looking out or Outside Looking in?: An Evaluation of Visualisation Modalities to Support the Creation of a Substitutional Virtual Environment,” in *Proceedings of the 2018 International Conference on Advanced Visual Interfaces*, 2018, p. 29:1--29:8.

[46] A. L. Simeone, I. Mavridou and W. Powell, "Altering User Movement Behaviour in Virtual Environments," in *IEEE Transactions on Visualization and Computer Graphics*, vol. 23, no. 4, pp. 1312-1321, April 2017.

[47] L. Shapira and D. Freedman, "Reality Skins: Creating Immersive and Tactile Virtual Environments," *2016 IEEE International Symposium on Mixed and Augmented Reality* (ISMAR), Merida, 2016, pp. 115-124.

[48] M. Orlowski, "A New Algorithm for the Largest Empty Rectangle Problem," Algorithmica, 1990, volume 5, pp. 65-73.

[49] L.-P. Cheng, E. Ofek, C. Holz, H. Benko, and A. D. Wilson, “Sparse Haptic Proxy: Touch Feedback in Virtual Environments Using a General Passive Prop,” in *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*, 2017, pp. 3718–3728.