Procedurally Generated Virtual Reality from 3D Reconstructed Physical Space

Misha Sra[1], Sergio Garrido-Jurado[2], Chris Schmandt[1], and Pattie Maes[1]

[1]MIT Media Lab, Cambridge, MA, 02142 USA
[2]University of Córdoba, Córdoba, 14071 Spain
sra@media.mit.edu, i52gajus@uco.es, geek@mit.edu, pattie@media.mit.edu

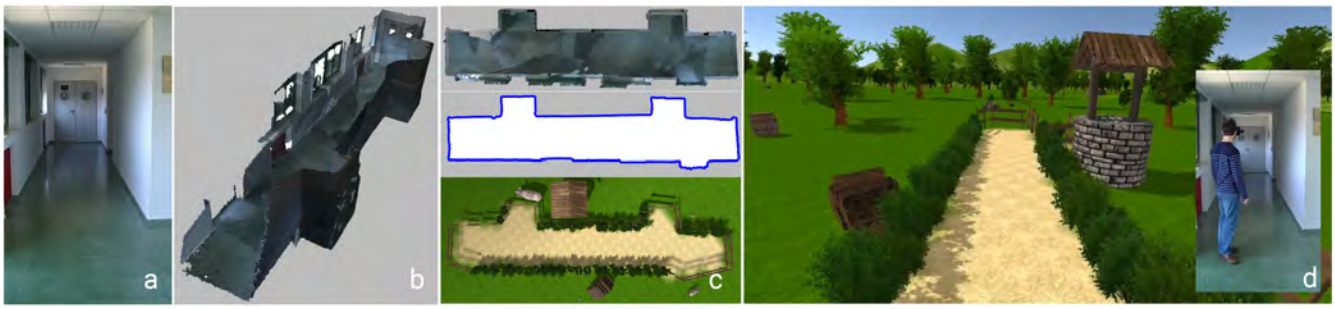

Figure 1: *Different steps of the proposed system. (a-b) We start with creating a 3D map of the real environment. (c) We detect the walkable area in the input 3D map to determine where the user can move freely. The generated virtual world is created according to the estimated walkable area in the point cloud. (d) Inset shows a user navigating the generated virtual environment by walking in the real environment, while visually experiencing it through a Tango HMD.*

Abstract

We present a novel system for automatically generating immersive and interactive virtual reality (VR) environments using the real world as a template. The system captures indoor scenes in 3D, detects obstacles like furniture and walls, and maps walkable areas (WA) to enable real-walking in the generated virtual environment (VE). Depth data is additionally used for recognizing and tracking objects during the VR experience. The detected objects are paired with virtual counterparts to leverage the physicality of the real world for a tactile experience. Our approach is new, in that it allows a casual user to easily create virtual reality worlds in any indoor space of arbitrary size and shape without requiring specialized equipment or training. We demonstrate our approach through a fully working system implemented on the Google Project Tango tablet device.

Keywords: Virtual Reality; Mobile Computing; Procedural Generation; Computer Vision; 3D Reconstruction; Depth Cameras; Locomotion; Obstacle Avoidance; Tracking

Concepts: •**Human-centered computing** → **Virtual reality; Interaction techniques;**

VRST '16,, November 02-04, 2016, Garching bei München, Germany
ISBN: 978-1-4503-4491-3/16/11…$15.00
DOI: http://dx.doi.org/10.1145/2993369.2993372

1 Introduction

Following their introduction in the 1960s, head-mounted virtual reality systems have mainly focused on visual and aural senses [Sutherland 1968; Cruz-Neira et al. 1992]. In order to enhance immersion in the virtual world, researchers have since pursued the addition of movement and haptic sense through motion platforms, exoskeletons, and other hand-held devices [Burdea and Coiffet 2003]. Realism of locomotion, a fundamental requirement for action in both real and virtual environments, had been a challenge to achieve until redirected walking, a technique that introduces a rotational gain in order to imperceptibly rotate the user away from the boundaries of the tracking space [Razzaque et al. 2001], was introduced. Redirected walking has made possible natural and unconstrained walking in VEs without using mechanical locomotion devices. However, the technique requires a relatively large physical space and is thus not suitable for the typical home or office environment.

We present a novel pipeline to generate an interactive VR experience using the physical environment as a template that allows natural and unconstrained walking in a visually-immersive virtual world. Our approach incorporates the concept of *passive haptics* [Insko 2001], i.e., receiving feedback from touching a physical object that is registered to a virtual object through object detection and tracking. In stark contrast to previous approaches built on passive haptics, where physical objects were constructed and placed in the real environment (RE) [Insko 2001], our system automatically detects existing objects in the real world and places corresponding virtual objects in the VE. Users receive full haptic feedback by interacting with the real world object through its virtual proxy. Prior research suggests there are benefits to physical replication of objects in immersive virtual environments where use of actual objects significantly increased self-reported solidity and weight, not only of the object touched but also others in the scene [Hoffman 1998].

While advances in consumer VR technology e.g., HTC Vive have made it easy to accurately capture users' motions over

room-sized areas using external tracking devices, the walkable area is ultimately restricted by the size of the tracked space and constrained to a fixed regular shape. We overcome this room-scale limitation by using a mobile device that allows walking to build and experience virtual worlds that can span the size of an entire house or a whole office floor. Our system is currently limited to spaces with a constant floor height, i.e., we do not process stairs or ramps.

The key contributions of our work are the following:

- A novel framework for using the physical environment as a template for automatically generating an immersive virtual world that conforms to any indoor physical space.
- An object detection and tracking pipeline for incorporating interaction, with physical objects through their virtual counterparts, in the VR experience.
- An end-to-end mobile application with the first unique combination of 3D mapping, obstacle detection, object detection and tracking, automatic VE generation, and haptic feedback.

We believe our framework is the first of its kind to deliver an easy and automated mechanism for procedurally creating interactive VEs without any input physical space size or shape limitations. Our system goes beyond the initial 3D mapping of the real world environment that detects planar surfaces (walls, floors, tabletops etc.) and adds object recognition and tracking which is missing from existing devices and experiences that also map the real world e.g., Hololens[1], Occipital Bridge Engine[2]. Users can create their own VR experiences, turn their living rooms into space stations, walk through the Grand Canyon or go for a stroll on Mars within minutes. We contemplate the use of our system in developing VR gaming and storytelling, education, remote tourism, architectural walkthroughs, training, and simulation. For example, space in a museum with passive haptic objects could be used to immerse visitors in any historical time period. Safety training through simulated rescue operations in replica environments is another area of interest. Generating remote environments for watching sunsets from the comfort of a user's home could provide relaxing escapes from reality.

We demonstrate our system through the generation of four different virtual worlds based on 3D scans of two different physical spaces.

2 Related Work

The work presented in this paper attempts to simplify and automate the process of creating interactive VEs by using the real world as a template. Our system allows anyone who can use a mobile device to be able to build a VR experience compared to existing VR authoring tools that require programming and 3D modeling skills. We summarize below a few most directly related works.

2.1 3D reconstruction and object detection

The appearance of low-cost range sensors, such as the Microsoft Kinect, has provided easy access to fast and robust 3D reconstruction. KinectFusion [Newcombe et al. 2011] and its variants [Peasley and Birchfield 2013; Qayyum et al. 2013] are among the most popular techniques for hand-held scanning of indoor scenes. More recently, Googles Project Tango[3] tablet with an integrated depth camera and inertial sensors has been used for virtual and augmented reality applications [Nassani et al. 2015]. Similar to 3D reconstruction, object detection has also benefited from the availability of low-cost depth cameras. Many new approaches that use RGBD data from depth sensors have been proposed for object detection [Lai et al. 2013; Alexandre 2012], and they generally provide more robust detection than possible with 2D images. The Sliding Shapes detector [Song and Xiao 2014] extends the 2D sliding window approach for object detection in images to depthmaps. A window is moved along the 3D space in a point cloud and evaluated by a classifier at each position. An ensemble of exemplar SVMs is used to decide if the window contains an object.

2.2 Real walking in VR

Natural walking is a desired feature in many VR applications [Usoh et al. 1999] but remains a challenge because of space and tracking requirements. Redirected walking makes natural walking in VEs possible by tracking and manipulating the user's real world trajectory [Razzaque et al. 2001]. The recently introduced consumer HTC Vive[4] system allows a user to move in a small tracked space with a maximum size of $15' \times 15'$. For small spaces, low cost depth sensors have been used to track users [Sra and Schmandt 2015]. The idea of applying 3D analysis to determine walkable area is outlined by Nescher [Nescher et al. 2016] although its implementation is not presented. Change blindness, a perceptual phenomenon that occurs when a person fails to detect a visual change to an object or scene was used to allow a user to walk through a virtual environment that is an order of magnitude larger than the physical space [Suma et al. 2011].

2.3 Passive Haptics in VR

Passive haptics have been shown to both enhance immersion in VR and also make virtual tasks easier to accomplish by providing haptic feedback [Hinckley et al. 1994]. Adding representations of real objects, that can be touched, to immersive VE enhanced the feeling of presence in those environments [Hoffman 1998]. Low-resolution physical models made of styrofoam and plywood were found to significantly improve presence [Insko 2001]. TurkDeck used "human actuators" to operate physical props in real-time [Cheng et al. 2015]. Substitutional Reality pairs every physical object surrounding a user to a virtual counterpart [Simeone et al. 2015]. In Annexing Reality [Hettiarachchi and Wigdor 2016], the system detects simple geometry primitives (e.g., cylinders, cones) from objects placed on a table to match known virtual objects.

2.4 Adaptive Systems

HTC Vive is a recently introduced consumer VR device that allows developers to create experiences using natural locomotion in a room sized space. The Vive Lighthouse tracking system requires users to manually trace out a play area, clear of furniture and obstacles, using the included hand-held devices. If the area is oddly shaped, the largest square or rectangle (maximum size 15'x15') that can fit in the space is chosen and the VR app is loaded accordingly. Hololens and Occipital Bridge Engine, both augmented reality (AR) devices, analyze the user's physical environment to find flat surfaces like a wall or a couch seat to determine potential locations for placing virtual objects in the real world, as described in FLARE [Shapira et al. 2014]. SnapToReality [Nuernberger et al. 2016] presents a related AR system where edges and planes in the physical environment are detected for aligning virtual content placement.

We designed our system to overcome the limitations of existing VR

[1] Hololens. https://www.microsoft.com/microsoft-hololens/en-us

[2] Occipital. http://structure.io/developers#bridge

[3] Project Tango. https://get.google.com/tango/

[4] HTC Vive. https://www.htcvive.com/

experiences that incorporate passive haptics or natural locomotion. Unlike the Vive, the physical space in our system where the user walks can be regular or irregularly shaped e.g., square or an octagon with a hole with no limitation on size. The user does not need to prepare a play area in their living room by moving furniture out of the way as our system uses any given physical space as input for procedurally building a virtual world that conforms to that space. While the Vive is tethered to a PC and the user needs to stay within the Lighthouse tracked volume, users in our system are free to move from room to room to generate large virtual worlds. This is possible because the Tango provides pose in 6DOF and does not need an external tracking system.

Even though we analyze the physical environment for planar surfaces, similar to the Hololens or Occipital, we use the output as a template to create a complete virtual environment, with visual indications of where a user can and cannot go. Unlike AR systems, since the user cannot see the physical world at all in a VR experience, we must build our virtual world to provide safe travel for the user. We go further, and add interaction with automatically detected real world objects through their virtual counterparts, something neither of the above mentioned devices do. While current object recognition systems can achieve remarkable recognition performance for individual classes, e.g., chairs in our case, the simultaneous recognition of multiple classes remains a major challenge. We focus on recognizing only one class of objects as our primary goal is to demonstrate user interaction with daily objects in their natural environment through passive haptics.

Our target space is home or work environments where it is impractical to radically rearrange furniture, build matching props, or use "human actuators". We have not found any approach like ours in the literature that automatically generates a VE using the physical space as input with WA segmentation and object detection and tracking for passive haptics.

3 System Overview

In this section we describe our virtual reality generation system, designed for creating interactive VEs that allow users to walk and interact with objects in the real world while being visually immersed in the virtual world. The main idea is to use the physical world as a template for the VE so as to create a correspondence in scale, spatial layout, and object placement between the two spaces. Figure 1 shows a user immersed in one of the generated experiences. The user can freely walk, sit, bend down, or turn and tilt their head. Tango continuously tracks the user's position and provides them with a first-person view into the VE.

Figure 2: *Examples of user interaction in the virtual world. (a) Proximity based interaction with virtual elements (virtual world as seen from a first person perspective). (b) Interaction with a real chair through passive haptics (virtual world as seen from a third person perspective).*

One main aspect of our system is that it provides the user a physical/haptic experience along with a visual and auditory one. Our system achieves this through real-walking in the VE and *passive haptics* i.e. whenever users touch a particular object in the virtual world, they also touch a corresponding object in the real world. Figure 2b illustrates a user feeling the realness of a virtual chair in our generated VE by touching the corresponding real chair and sitting down.

The system does an automatic detection of walkable areas from input 3D mapping data. This is unlike existing approaches for real walking in a VE that are costly and time-consuming, require a tracking system and a tracked space, and are created manually. Our use of object detection for passive haptics is different from previous systems where the RE is constructed with low resolution physical props to match the design of the VE [Insko 2001].

The software pipeline (Fig. 4) includes: (i) building a 3D map of the RE using the Tango device, (ii) analyzing the 3D data to determine WAs, free of obstacles like walls or furniture, (iii) using the depth data of the RE to do object detection and tracking, (iv) using the mapped WA to procedurally generate a VE, (v) placing virtual models of detected objects in the VE, (vi) tracking the user and the objects in real-time as they experience and interact with the VE through an HMD.

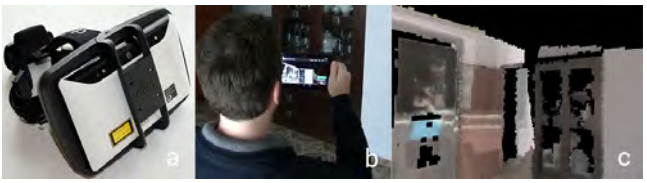

Figure 3: *(a) Tango in a virtual reality headset. (b) A user walking with the Tango device to create a 3D model of a room. (c) The generated 3D model. Scanning a* 3×4 *m full room took less than 3 minutes.*

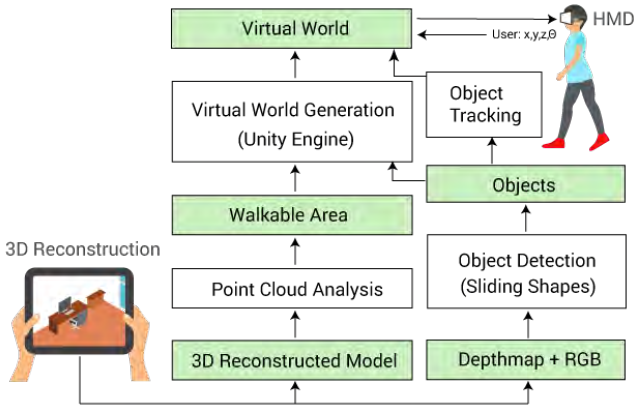

Figure 4: *Full process diagram starting with a 3D scan of the real environment (bottom left) to the user walking with an HMD in the generated virtual environment (top right). Green boxes represent data while white ones represent sub processes in our system.*

Our system allows two types of interactions with the procedurally generated VE, namely interaction with digitally created elements and interaction with elements that have real world counterparts. Digitally created elements e.g., wildlife in (Fig. 2a) respond to the user's presence thus interaction with them is based on the user's location in the VE as well as the user's proximity to them. For interacting with objects that have real world counterparts, e.g., the chair (Fig. 2b), once the user puts on the HMD, the physical

objects are never seen directly, only felt. Thus interaction with them happens through their virtual representations.

4 Technical details

In this section, we describe the key components of our system.

4.1 Walkable Area Detection

The main input for our system is a 3D reconstruction of the physical environment. For acquiring input data, we use the Tango device which features a motion and depth sensing camera to create a 3D map of the environment. The integrated sensors continuously return the 3D position and orientation of the device producing a registered point cloud of the environment in real-time. A point cloud created through any other 3D scanning technique or device would also be valid input for our system.

Figure 3 shows a sample reconstruction using Tango. We detect the walkable area in the reconstructed 3D model and use it to procedurally generate a virtual world.

Figure 5 illustrates how we determine the walkable area, the space that is free of obstacles like furniture, in the 3D map of a living room. We focus on detecting open space and use boundary elements (Section 4.2.2) to create a visual barrier between the detected open space and the obstacles like furniture or walls. We begin by pre-processing the input point cloud, $\mathcal{P} = \{\mathbf{x}_i\} \in \mathbb{R}^3$, which involves removing isolated components with bounding diameters smaller than an empirically determined minimum size of 2 cm. Using the pre-processed point cloud, we separate the floor from the rest of the elements in the RE, i.e. furniture, walls and other obstacles. The floor isolation is achieved by fitting a plane $\mathcal{Q}$ to the points in $\mathcal{P}$ using RANSAC sample consensus.

The floor point cloud, $\mathcal{P}_{floor}$, is composed of those points in $\mathcal{P}$ with a distance to plane $\mathcal{Q}$ shorter than a threshold ε, i.e., $\mathcal{P}_{floor} = \{\mathbf{x}_i\} \in \mathcal{P} \mid \mathfrak{D}(\mathcal{Q}, \mathbf{x}_i) < \varepsilon$, where $\mathfrak{D}$ is the orthogonal distance function between a plane and a point. A minimum distance of $\varepsilon = 0.05$ m was employed in the processing of the 3D reconstructed models of our indoor scenes.

The rest of the points in $\mathcal{P}$ belong to potential obstacles. Objects that are above the user's head, like the ceiling, do not impact the WA and are ignored since there is no possibility of collision with them. We define the point cloud of obstacles as $\mathcal{P}_{obst} = \{\mathbf{x}_i\} \in (\mathcal{P} - \mathcal{P}_{floor}) \mid \mathfrak{D}(\mathcal{Q}, \mathbf{x}_i) < h$, where h is the minimum height. The value of h can be incremented if we want to consider the possibility of a user jumping in the VE. Figures 5b and 5c show $\mathcal{P}_{floor}$ and $\mathcal{P}_{obst}$ respectively.

A top view of the WA provides a simplified representation while maintaining the important information needed to replicate the space in the generated VE. To get the top view image, I_k, of a point cloud $\mathcal{P}_k$, we project the 3D points onto the floor plane $\mathcal{Q}$.

Figures 5(d-e) show the top binary images for the floor and obstacle point clouds, i.e., I_{floor} and I_{obst}.

By combining these two binary images we obtain a top view of the WA in the RE. We combine the images as $I_{WA} = I_{floor}(1 - I_{obst})$.

I_{WA} represents the areas in the 3D reconstructed model where a floor has been detected and there are no obstacles like walls or furniture (Fig. 5f). The rest of the space outside this area is occupied by an obstacle or its status is unknown. We use this binary image as a guide for generating the VE. To account for errors in reconstruction and tracking, we apply an erode filter to shrink the WA. This increases the distance between the WA and the obstacles in the environment for a safer immersive experience. To segment the exact boundaries of the WA for the VE generation process, we detect contours in I_{WA} (Fig 5g) and only keep the one which outlines the largest WA in the environment. When present, we keep internal contours representing holes in the WA, as they can lead to the generation of interesting VEs such as inland lakes or craters.

4.2 Virtual World Generation

After analysing of the 3D reconstructed model we generate the VE as described below.

We use the estimated WA as the central element for our procedural map generation. The visual aesthetic of the virtual world is determined by the design of the available set of models, textures, etc.

The generated VE is composed of three types of elements: (i) static elements, (ii) WA boundary elements, and (iii) other virtual world elements. Each type of element serves a specific purpose. It is either visual, interactive, functional or a combination thereof and impacts the user's immersive experience as described in each of the following sections.

4.2.1 Static elements

Static elements are visual elements that stay at the same position, orientation, and scale following each generation of the VE. Some examples of static elements in our generated VE are the skybox, mountains in the distance, and the ground terrain (see Figure 6a). In our design, these elements are usually visible to the user from a distance and influence only the visual feel of the virtual space.

4.2.2 Boundary elements

An important feature of our generated VE is the capacity of the user to move freely within the mapped WA without fear of colliding with any object or wall in the real world. This feature calls for design techniques that prevent the user from walking into occupied areas while they are visually-immersed in the VE.

We use virtual elements, referred to as boundary elements, that indicate areas in the virtual world the user should not access. Boundary elements are both visual and functional and, hence, the type (e.g., fence, shrub, water) of virtual items used is important. Since people do not usually walk through obstacles in the real world, we use common sense knowledge of real world behaviors as well as simple design techniques from video games to place appropriate boundary objects in generated VEs.

The output VE can be an outdoor environment (e.g., a forest or a Martian landscape) or an indoor one (e.g., a cave or a space station), independent of the type of physical environment of the user. We focus on creating virtual worlds that look and feel much bigger than the 3D mapped indoor spaces they are based on.

A concern in designing visually expansive spaces is that even though the users are aware that they should not pass through the boundary barriers, there is some chance that they may touch or lean on those barriers, resulting in them touching furniture or walls in the real world. The disconnect between touching, for example, a fence and touching a piece of furniture in the real world would be disruptive for immersion. A potential solution would be to use passive haptics to replace all obstacles in the real world with corresponding virtual counterparts, e.g., virtual walls for physical walls similar. This, however, would constrain the generated VEs to indoor environments. To reconcile generating large open spaces

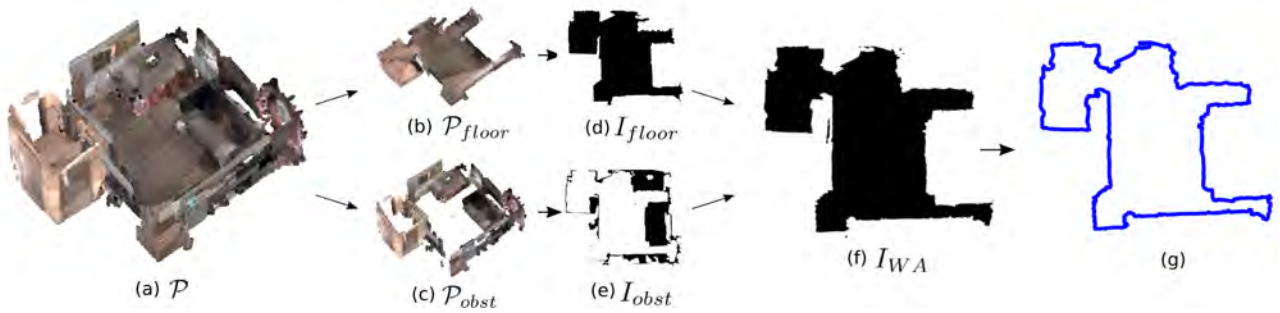

Figure 5: *Walkable area detection. (a) 3D reconstructed model. (b-c) Floor and obstacle point clouds. (d-e) Top views of the floor and obstacles. (f) Segmented walkable area using d and e. (g) Contour of detected walkable area.*

while still providing tactile feedback, we employ passive haptics for the interactive objects in the scene e.g., a chair in this demo.

Figure 6: *Virtual world generation for environment in Figure 1. (a) Static elements like terrain and mountains are placed first. (b) Boundary elements like fences, and shrubs are positioned next. (c) Finally, rest of the world elements are filled in to create a VE that feels alive.*

In the farm VE in Figure 6b, a wooden fence and shrubs are placed around the WA to prevent users from leaving that space. The elements visually and functionally blend in with the design of the VE and do not command any particular attention. Since, fences and shrubs are commonly used in the real world for enclosing a space, we expect the user to behave in a manner similar to their real world behavior when they encounter them in our VE. We believe, this can help prevent some, if not all, potential mishaps by keeping users within the WA.

The design and type of these boundary elements varies based on the design of the virtual world. For example, the water in Figure 2b or lava in Figure 2c acts as a virtual boundary and prevents users from walking into walls or furniture in the real world. Other boundary element examples are the blackness of space, a canyon ledge, red rope barriers with post stanchions in a virtual museum or rocks in a virtual cave. In addition to boundary elements, we also differentiate between the floor texture of the WA from the rest of the open space in the VE to reinforce the spatial demarcation.

To prevent collisions with real world obstacles, the HTC Vive includes a ”chaperone” system. It works by displaying a wall-like blue grid in the user’s virtual vision when they are in close proximity to the boundaries of their configured play area. If the user gets closer still to the boundary, the forward-facing camera gives them a sort of thermal view of their surroundings. There is still the potential obstacle of having a cord trailing the user around the room. We did not find any studies that explore if and how the ”chaperone” system affects immersion in VR. However, in our experience some users chose to disable it for greater immersion, even at the risk of potential collisions with real world obstacles.

Before automatically placing the boundary elements in the scene, we simplify the contour of the WA by performing a polygonal approximation. The type of boundary element to be placed is

Figure 7: *Four different virtual worlds generated for the same real environment (Figure 5a) using different visual styles and procedural generation rules.*

selected using contour parameters like angle or segment length. For example, in Fig 6, fences look natural when placed along straight edges but intersect awkwardly when placed at contour corners with acute angles. We therefore include softer elements like shrubs for placement in the corners. They are smaller, fit in the tight space and even if two shrubs intersect, they do not look unnatural.

For some VEs, polygonal approximation is a necessary step needed to produce a more pleasant and natural looking visualization. In other VEs, we may not need any boundary elements and polygonal approximation. For example, in the floating island VE (Fig. 7b) we use water as a boundary element and the noisy contours do not effect the visual outcome negatively. In fact, the jaggedness adds to the visual realism of the floating island.

4.2.3 Virtual world elements

Once the static and boundary elements are in place, we add the rest of the elements in the environment as shown in Figure 6c. These elements impact the user’s visual and interactive experience in the VE. Placement of the world elements is a design task. VE generation consists of an optimization process to obtain the virtual world that best accomplishes defined rules.

Techniques for procedural map generation usually employ a set of rules that describe the desired features in the output map. We create a set of rules that define spatial relationships between sets of virtual

elements. The rules also take into account proximity of items to the WA and their orientation relative to the WA. For example, in the farm VE, the door of the house always faces the fenced area. We created 25 different rules for the placement of elements in the farm virtual world shown in Figure 6c. Some of the rules employed are (see Appendix A for the full list):

1. Virtual elements do not intersect with the WA.
2. Animals are placed close to the WA.
3. House faces the WA.
4. No trees between the house and the WA (so that visibility is not impacted).
5. Baby boars close to the mother boar.
6. Pail close to the well.
7. Horse close to the shelter, etc.

To generate an aesthetically pleasing virtual world map, we need to optimize the position of the elements based on the designed rules. We chose to use a genetic algorithm (GA) [Whitley 1994] because it allowed us to model the optimization function more explicitly than other optimization approaches. We also wanted to use a more sophisticated method than a simple heuristic/random approach. A near optimal solution provided by the GA was preferred because it created a different VE after each execution.

We employed a GA with elitism, i.e. the best individual remains in the next generation, and a maximum number of iterations as stop criteria. Each individual is composed of the poses of N elements, i.e., all the virtual elements in the scene. Each pose is a 2D transformation matrix describing the translation, position, and scale of each virtual element as viewed from the top. The values for the initial population are generated randomly. The crossover operator is performed by two individuals from the previous generation selected by fitness ranking. A uniform crossover approach [Syswerda 1989] is performed so that the N virtual elements of each parent are randomly distributed between the two new children. The mutation is randomly applied to each new individual, producing a small modification in position, rotation or scale. Finally, the fitness function evaluates the optimality of an individual (a virtual world composed of N virtual elements) with respect to the set of rules. For each rule r_i we define a function $f_i \in [0, 1]$ that provides a score for that rule based on the current individual. For example, a function to evaluate a minimum distance between two elements (e.g. rule 6) is expressed as a ramp function. The fitness function for an individual is then defined as the weighted average of each of the rule functions f_i, that assign a different weight, w_i, to each rule depending on its importance: $\sum_{i=0}^{N} w_i * f_i$

We use the best individual from the last generation to place the virtual elements in the generated VE.

Figure 7a shows the virtual world generated for the living room environment in Figure 5a. By altering the style (meshes, textures, shaders) and the rules we generate a totally different world for the same real environment (Figs.7b-c-d).

4.3 Object Detection and Tracking

Object detection in the real world for use in our VE is a challenging task due to variations of viewpoint, occlusion, self-occlusion and sensor noise to name a few. We use the Sliding Shapes object detector [Song and Xiao 2014]. This method slides a window along the 3D space of a depth map. Each position is evaluated using an ensemble of SVM classifiers to determine if the window contains a trained object. The positions with a score over a threshold are considered and non-maximum suppression is applied to remove duplicate detections. Figure 8 shows the input and output of the detection process.

Figure 8: *Example of chair detection using the Sliding Shapes detector (a) RGB image. (b) Depth image (c) Detected chairs.*

Similar to many classification techniques, Sliding Shapes requires an initial training step. It uses a large set of synthetically generated object models of typical indoor objects (e.g. chairs, tables, etc). Hence, objects that are considered for recognition and thereby interaction in our VE need to be selected a priori in order to perform offline training and classification. An automated system could ideally detect which objects are available in the room, select some of them for inclusion in the design, and choose object models which would be appropriate candidates as their virtual counterparts, using a rule-based system.

We capture several depth maps of the user's environment during the 3D scanning process for classifying objects in the scene. The final output after classification is the position, orientation and scale of a detected object that we use to position a similar object model in our VE at the beginning. The point cloud is also used for object tracking when the user is in VR mode.

In our example output (Fig. 2b), we detect a chair in the real world and place a corresponding chair in the VE at the same scale, position and orientation. We compare the object in the scene with 880 instances of chairs during classification to recognize not only that a particular object is a chair but also what type of chair e.g., an office chair with wheels and armrests. This allows us to better match a virtual counterpart to the detected object and also enables detection of a very wide range of chairs.

Detected objects are tracked in real-time when the user is in VR mode. We store the point cloud of the detected object in its original position and look for the object in each subsequent depth frame. Real-time tracking is achieved using ICP [Besl and McKay 1992] which provides the relative 3D transformation between two poses from two point clouds. We compare the depth map provided by Tango with the stored point cloud of the object to be tracked (e.g., chair). If a change in pose is detected, we update the virtual object's position accordingly. To improve performance and robustness, we only perform ICP when the user is looking at the object and we crop the input depth map to the volume surrounding object at its current position. ICP is unable to converge for large or fast object movements, especially since it is running on a mobile device, unlike the GPU based implementations like KinectFusion [Newcombe et al. 2011]. While we have built marker-based tracking to overcome this limitation, the need for adding markers in the user's environment takes away from the goal of creating a fully automated system for VR world generation.

5 User Experience

One remarkable feature of our system is its capability to support user interaction with physical and virtual elements. We describe two interaction scenarios to highlight the possibilities of the VEs generated using our approach (Fig. 2).

Upon start of the application, the user enters a virtual world from a fixed position in the physical world that was mapped earlier and observes the generated outdoors through the Tango HMD. Figure

2a shows how our system allows the user to walk in the RE while being visually immersed in the farm world. Fig 2b shows how the system allows the user to feel the chair on the floating island. The objective behind adding passive haptics i.e., when users touch or manipulate an object in the virtual world, they simultaneously also touch or manipulate a corresponding object in the physical world, is to embrace the domestic environment filled with furniture and objects by making it part of the users' experience.

In a generated VE, the detected object appears as a 3D model of the real one. It thus, presents the same affordances and allows the same interaction as its physical counterpart. The concept of affordance introduced by Gibson [Gibson 1979] refers to the interaction possibilities perceived by the observer of an object. For instance, a chair affords sitting.

5.1 Physical Object Interaction

In Figure 2b, we use a virtual chair that is similar in style to the rest of the elements in the generated virtual world. Because it is not a replica of the physical chair, its virtual and physical properties differ from an aesthetic and tactile perspective. The altered physical properties do not affect the way the objects functionality is perceived [Simeone et al. 2015]. For example, in the VE, the material of the chair may appear to be made of wood or stone. This affects the way the object is perceived in terms of temperature, weight, texture, and hardness but not functionally; the user still expects to be able to use the chair for sitting. The visual discrepancy can be resolved by using a high-fidelity chair model that more closely matches the real chair. Prior work shows as long as the discrepancy between the visual information and the haptic information does not get too large, the visual information will dominate [Warren and Cleaves 1971] and the discrepancy will not adversely impact the user's experience in virtual reality. In the chair example in Figure 2b, the user gets a more engaging tactile experience of sitting in a virtual chair while simultaneously sitting is the matching physical proxy. The example scene of a chair on a floating raft may seem contrived but the goal of that specific scene was to reiterate that obstacles like tables and walls in the room, that are outside the WA, are represented by a boundary element, e.g., water and not a virtual counterpart. Only the selected physical objects in the room have a surrogate virtual object with which the user can interact.

The presence of virtual representations of real world objects does not limit the flexibility of the generated VE as long as the number of detected and interactive objects is kept low. This low limit allows the design of interactions with some specific objects while still maintaining the idea of the virtual world being totally different from the real world.

It must be noted that since the user can get close to a detected interactive object, like the chair in Fig. 2b, the object needs to be included in the WA segmentation. We include the chair in the area by projecting the bounding box of the detected chair onto the estimated floor plane image.

5.2 Virtual Object Interaction

Some virtual elements in the VE respond to the user's presence, making the world feel alive. For example, in the sample world shown in Figure 2a, virtual animals flee when the user approaches or startles them. Though we demonstrate only one, triggers based on changes in the user's position and orientation can be easily included for a richer virtual experience. Possible triggers include, detecting if the user is looking at the floor, if they are bending or running, or if they have entered a particular area.

6 EVALUATION AND DISCUSSION

6.1 Implementation Details

The point cloud analysis, object detection and virtual world generation process was done on a Windows 10 computer with an Intel Core i7-6700K 4.0GHz processor and 16GB RAM. Object tracking was done on the mobile device. The virtual world was generated using the Unity game engine and textures, models and other 3D elements were downloaded from the Unity Assets Store.

Total time required to process a point cloud and generate a virtual environment was about 2-4 minutes for each physical space we tested, including the time taken to do the initial 3D mapping. The time consuming pre-process was object recognition, which varied from 20 minutes to 1 hour depending on the complexity of the object being detected. This performance could be easily improved by reducing the large set of instances we tested against, 880 in case of the chair, though that may reduce accuracy. When we tested with 20-40 instances, the detection time went down to a couple of minutes. Our current code is not parallelized nor running on a GPU so we believe the detection time will go down even further with some engineering and optimization. We determined that since object recognition would be done once at the beginning, it would not impact the VR experience adversely by not running in real-time. However, keeping the physical object and its virtual counterpart in sync during VR mode was needed to avoid collisions and thus real-time object tracking was implemented. We achieved a frame rate of 25-30 fps during immersion with one object being tracked.

The GA employed a population of 1000 individuals, a mutation probability of 0.05 and a maximum number of 1000 iterations. These parameters, along with the weights w_i of the fitness function were determined experimentally in order to fulfill the proposed rules.

We use the Tango tablet both as the 3D mapping device and as an HMD (Fig. 3). It is self contained, mobile, provides tracking in real-time, has higher computational resources than other similar tablet devices, and has integrated sensors that are necessary for accomplishing our goals.The Tango SDK provides functionality to perform 3D reconstruction and user tracking. The Constructor[5] tool for Tango was used for 3D reconstruction in our examples. All other steps, i.e., WA detection, VE generation, object recognition and tracking, were specifically implemented for this work. Any other HMD which allows user tracking could be used for the immersive experience. Since all we need is a standard point cloud of the physical environment, data from non-HMD devices like the Microsoft Kinect or the Intel RealSense camera would be acceptable for generating the VE. However, for object tracking, it would be necessary for the HMD to have a depth sensor in order to apply ICP in real time.

6.2 Qualitative analysis

We evaluated our system in different real environments and with several different users to test its functionality and capabilities. One evaluation task of paramount importance was safety of the system as users walk around while their senses of sight and sound are occluded. Some users ignored the boundary elements, as we expected, and had to be steered away from the physical obstacles. We propose the addition of a tiered warning system using audio, haptic, and visual feedback where the warning mode and duration change from the least invasive to the most, as the user gets closer to the boundary. Qualitative user feedback was positive and almost

[5]Tango Constructor. https://developers.google.com/tango/tools/constructor

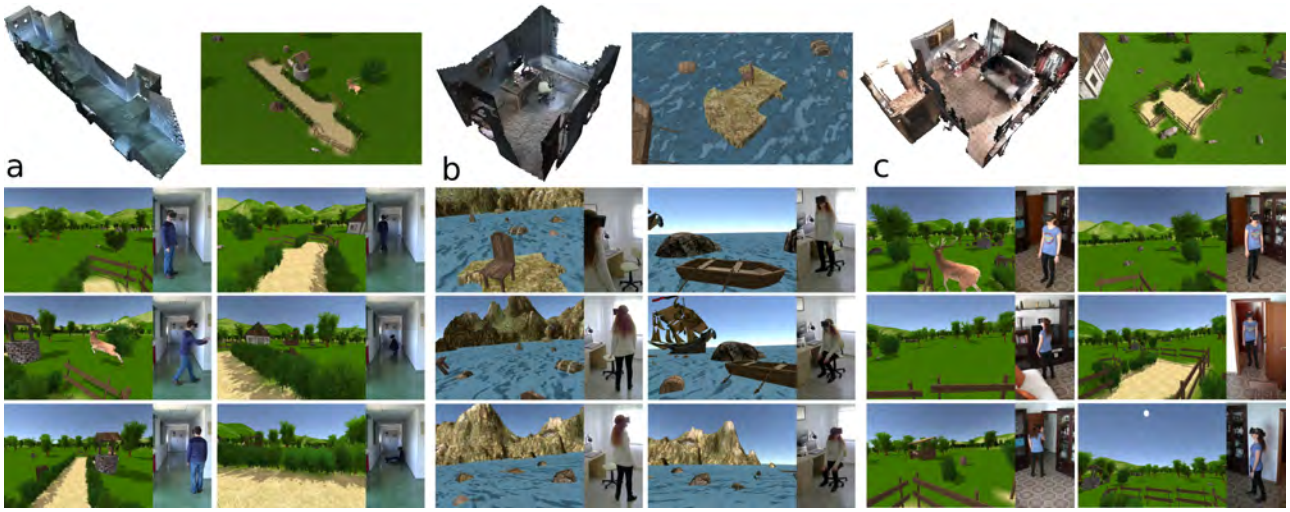

Figure 9: *Examples showing user immersion in three different real environments. (a) Hallway. (b) Office. (c) Living room.*

invariant. Test subjects were very excited, which is possibly caused by the novelty effect combined with a high level of presence in the fully immersive VR system. All subjects were tentative about sitting in the virtual/physical chair and expressed surprise on being able to do so. In more than 20 tests, no subjects reported any nausea. This surprisingly good result may be explained by the lack of conflict between visual and vestibular cues.

In Figure 9 we present the results using our approach for three different physical environments. Each scene is a visually-immersive real-walking VE that the user experiences through an HMD. The user can move, walk, bend down in the environment while their pose (position, orientation) is continuously tracked by the Tango device. Figure 9b shows interaction with a real chair.

The ability to generate larger than room-scale VEs and walk freely in them provides a significant improvement compared with existing commodity virtual reality systems, such as HTC Vive. By using boundary elements that perceptually imply a border e.g. fences, the user clearly understands the demarcation between the space where they can safely move and the rest of the environment that is physically out of bounds but still accessible visually.

A simple addition would be a teleportation system for moving the WA to any location in the virtual environment (as in [Bowman et al. 1997]). This would allow users to travel and explore much larger virtual spaces using relatively small physical spaces available for walking.

6.3 Limitations

Our system has a few limitations that are worth mentioning.

The Tango device provides position and orientation data that is used in both the 3D reconstruction and motion tracking phases. In some cases, incorrect estimation of the user's pose can generate a displacement between their real and virtual world positions. The drift usually appears when the motion tracking system does not have reliable data to perform correct tracking, for example, because of fast movements, dimly lit spaces, relatively empty spaces with no features like blank walls and floors, or proximity of the device to obstacles such that camera images are not available.

Other than using boundary elements, we did not implement additional mechanisms for warning a user when they get close to a physical obstacle. We think an audio signal could provide a first warning as a user gets within a certain distance of an obstacle followed by a haptic and finally a visual indicator if they get closer still. It needs to be determined how the warnings may impact the user's sense of presence in the VE.

A user's sense of presence in the VE may be disrupted if they accidentally or purposefully touch the boundary elements. Since the boundary elements separate the WA from the obstacles in the physical space, touching them would mean touching a wall or a piece of furniture. However, since the boundary elements are not virtual counterparts of these real world objects, the disconnect between touching a virtual shrub and a physical table may be jarring. We chose not to include virtual counterparts of all physical objects because we were interested in generating large open virtual spaces for the available indoor spaces instead of limiting ourselves to generating virtual spaces that have a 1:1 mapping with the available physical space, especially since that has been explored by [Sra and Schmandt 2015].

Currently, we do not track other people or pets in the RE who could collide with or startle the VR user. We expect individuals not wearing an HMD to easily avoid colliding with the person wearing an HMD but pets may need to be tracked or temporarily removed from the space.

Since one of the first steps in WA detection is estimating the planar floor in the environment, the proposed system is limited to environments that do not include, ramps or stairs.

7 Conclusion and future work

In this paper, we described a novel VR generation system that uses the real world as a template and allows natural walking in the generated virtual world and incorporates haptic feedback. The system, thus, creates a highly immersive environment by combining egocentric scene viewing with proprioception, vestibular, and tactile feedback.

Our system is the first to allow casual users to quickly and easily create immersive and interactive VEs that can be experienced in an HMD. In order to achieve our design goals, we devised a procedural virtual world generation framework using 3D reconstruction and object recognition data. We demonstrated the procedural generation

through the automatic creation of a variety of virtual scenes for the same physical environment.

Besides overcoming the previously presented limitations, there are a few aspects of the system that we can improve upon in future work.

An immediate improvement would be to employ more sophisticated procedural map generation techniques for creating the virtual world. Other important potential improvements to optimize system performance and subjective experience are: (i) Expanding object detection and tracking to a wider range of objects beyond furniture and potentially to other people and pets. (ii) Adding a tracked full-body avatar that moves in sync with the user's real body movements. This may require using either a motion capture suit or some external skeleton tracking device like the Kinect and could limit the size of the walkable area. (iii) Providing predefined theme-based sets of 3D models (e.g., space, forest, fantasy) for giving the user an option to choose the theme for the generated VE. (iv) Real-time VR generation, where the virtual world unfolds and layers itself over the real world as the user walks around wearing an HMD. (v) The possibility of allowing multiple users to share the same virtual environment, even if they do not share the same physical space is an interesting and challenging problem that we are currently tackling. (vi) A user study including simulator sickness and presence questionnaires. (vii) A more detailed performance evaluation for the current system.

References

ALEXANDRE, L. A. 2012. 3d descriptors for object and category recognition: a comparative evaluation. In *Workshop on Color-Depth Camera Fusion in Robotics at the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Vilamoura, Portugal*, vol. 1, Citeseer, 7.

BESL, P. J., AND MCKAY, N. D. 1992. Method for registration of 3-d shapes. In *Robotics-DL tentative*, International Society for Optics and Photonics, 586–606.

BOWMAN, D. A., KOLLER, D., AND HODGES, L. F. 1997. Travel in immersive virtual environments: An evaluation of viewpoint motion control techniques. In *Virtual Reality Annual International Symposium, 1997., IEEE 1997*, IEEE, 45–52.

BURDEA, G., AND COIFFET, P. 2003. Virtual reality technology. *Presence: Teleoperators and virtual environments 12*, 6, 663–664.

CHENG, L.-P., ROUMEN, T., RANTZSCH, H., KÖHLER, S., SCHMIDT, P., KOVACS, R., JASPER, J., KEMPER, J., AND BAUDISCH, P. 2015. Turkdeck: Physical virtual reality based on people. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology*, ACM, 417–426.

CRUZ-NEIRA, C., SANDIN, D. J., DEFANTI, T. A., KENYON, R. V., AND HART, J. C. 1992. The cave: audio visual experience automatic virtual environment. *Communications of the ACM 35*, 6, 64–73.

GIBSON, J. J. 1979. *The Ecological Approach to Visual Perception*. Houghton Mifflin.

HETTIARACHCHI, A., AND WIGDOR, D. 2016. 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*, ACM, New York, NY, USA, CHI '16, 1957–1967.

HINCKLEY, K., PAUSCH, R., GOBLE, J. C., AND KASSELL, N. F. 1994. Passive real-world interface props for neurosurgical visualization. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, ACM, 452–458.

HOFFMAN, H. G. 1998. Physically touching virtual objects using tactile augmentation enhances the realism of virtual environments. In *Virtual Reality Annual International Symposium, 1998. Proceedings., IEEE 1998*, IEEE, 59–63.

INSKO, B. E. 2001. *Passive haptics significantly enhances virtual environments*. PhD thesis, University of North Carolina at Chapel Hill.

LAI, K., BO, L., REN, X., AND FOX, D. 2013. Rgb-d object recognition: Features, algorithms, and a large scale benchmark. In *Consumer Depth Cameras for Computer Vision*. Springer, 167–192.

NASSANI, A., BAI, H., LEE, G., AND BILLINGHURST, M. 2015. Tag it!: Ar annotation using wearable sensors. In *SIGGRAPH Asia 2015 Mobile Graphics and Interactive Applications*, ACM, New York, NY, USA, SA '15, 12:1–12:4.

NESCHER, T., ZANK, M., AND KUNZ, A. 2016. Simultaneous mapping and redirected walking for ad hoc free walking in virtual environments. In *IEEE Virtual Reality Conference 2016*, IEEE, 239–240.

NEWCOMBE, R. A., IZADI, S., HILLIGES, O., MOLYNEAUX, D., KIM, D., DAVISON, A. J., KOHLI, P., SHOTTON, J., HODGES, S., AND FITZGIBBON, A. 2011. Kinectfusion: Real-time dense surface mapping and tracking. In *IEEE ISMAR*, IEEE.

NUERNBERGER, B., OFEK, E., BENKO, H., AND WILSON, A. D. 2016. Snaptoreality: Aligning augmented reality to the real world. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, ACM, New York, NY, USA, CHI '16, 1233–1244.

PEASLEY, B., AND BIRCHFIELD, S. 2013. Replacing projective data association with lucas-kanade for kinectfusion. In *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, IEEE, 638–645.

QAYYUM, U., KIM, J., ET AL. 2013. Inertial-kinect fusion for outdoor 3d navigation. In *Australasian Conference on Robotics and Automation (ACRA)*.

RAZZAQUE, S., KOHN, Z., AND WHITTON, M. C. 2001. Redirected walking. In *Proceedings of EUROGRAPHICS*, vol. 9, Citeseer, 105–106.

SHAPIRA, L., GAL, R., OFEK, E., AND KOHLI, P. 2014. FLARE: fast layout for augmented reality applications. IEEE Institute of Electrical and Electronics Engineers.

SIMEONE, A. L., VELLOSO, E., AND GELLERSEN, H. 2015. 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*, ACM, 3307–3316.

SONG, S., AND XIAO, J. 2014. Sliding shapes for 3d object detection in depth images. In *Computer Vision - ECCV 2014 - 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part VI*, 634–651.

SRA, M., AND SCHMANDT, C. 2015. Metaspace: Full-body tracking for immersive multiperson virtual reality. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology*, ACM, 47–48.

SUMA, E. A., CLARK, S., KRUM, D., FINKELSTEIN, S., BOLAS, M., AND WARTE, Z. 2011. Leveraging change blindness for redirection in virtual environments. In *2011 IEEE Virtual Reality Conference*, IEEE, 159–166.

SUTHERLAND, I. E. 1968. A head-mounted three dimensional display. In *Proceedings of the December 9-11, 1968, fall joint computer conference, part I*, ACM, 757–764.

SYSWERDA, G. 1989. Uniform crossover in genetic algorithms. In *Proceedings of the 3rd International Conference on Genetic Algorithms*, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2–9.

USOH, M., ARTHUR, K., WHITTON, M. C., BASTOS, R., STEED, A., SLATER, M., AND BROOKS JR, F. P. 1999. Walking > walking-in-place > flying, in virtual environments. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, ACM Press/Addison-Wesley Publishing Co., 359–364.

WARREN, D. H., AND CLEAVES, W. T. 1971. Visual-proprioceptive interaction under large amounts of conflict. *Journal of experimental psychology 90*, 2, 206.

WHITLEY, D. 1994. A genetic algorithm tutorial. *Statistics and computing 4*, 2, 65–85.

A Sample rules for virtual environment generation

Full list of 25 rules used in the virtual world generation process for the farm world example in Figure 1 (see Section 4.2.3):

1. Virtual elements do not intersect with the WA.
2. Boars are placed close to the WA.
3. Deer are placed close to the WA.
4. Rabbits are placed far from the house.
5. Boars are placed far from the house.
6. Deer are placed far from the house.
7. Rabbits are placed far from the house.
8. Deer are placed close to trees or shrubs.
9. Boars are placed close to trees or shrubs.
10. Rabbits are placed close to trees or shrubs.
11. Baby boars are placed close to the mother boar.
12. Rabbits are placed close to each other.
13. House faces the WA.
14. No trees between the house and the WA.
15. No rocks between the house and the WA.
16. Shelter is placed close to the house.
17. Horse is placed close to the shelter.
18. Pail is placed close to the well.
19. Piles of wood are placed close to the house, or WA.
20. Barrels are placed close to house or WA.
21. Rocks are placed close to trees or shrubs.
22. Trees are not too close to WA.
23. Trees are not too close each other.
24. Shrubs are not too close each other.
25. Rocks are not too close each other.