Scenograph: Fitting Real-Walking VR Experiences into Various Tracking Volumes

Sebastian Marwecki, Patrick Baudisch
Hasso Plattner Institute, Faculty of Digital Engineering, University of Potsdam, Germany
{firstname.lastname}@hpi.uni-potsdam.de

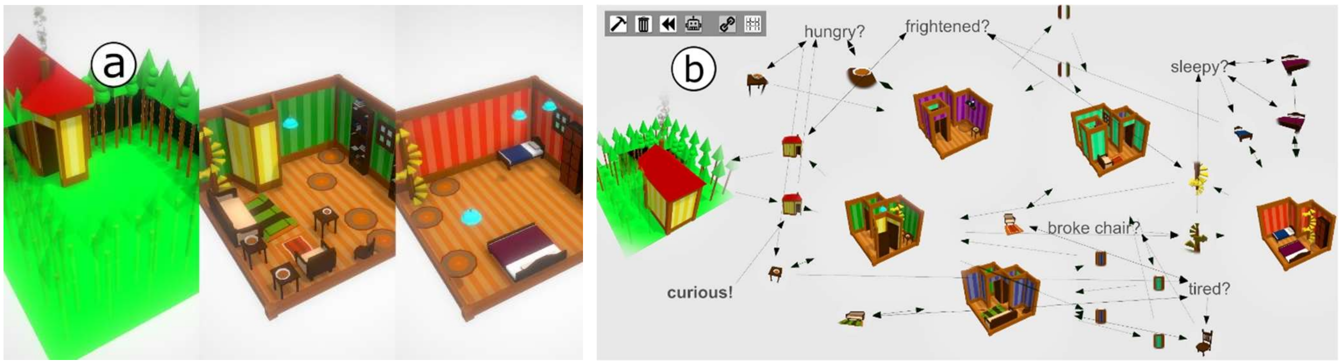

Figure 1 (a) Traditionally, designers of real-walking VR experiences have specific tracking volumes in mind. This rendition of the fairy tale 'Goldilocks' consists of three 25m² locations filled with interactive assets. Users change locations using corridors as portals [26]. Unfortunately, specifying the tracking volume prevents the experience from running on smaller tracking volumes. (b) Scenograph addresses this through a tracking volume-*independent* representation of real-walking experiences. This allows Scenograph to instantiate experiences for tracking volumes of different size and shape. Here we used Scenograph to map 'Goldilocks' to an L-shaped 8m² space. While maintaining the narrative structure, it splits the three locations into six smaller ones, each fitting the new tracking volume.

ABSTRACT
When developing a real-walking virtual reality experience, creators generally design virtual locations to fit a specific tracking volume. Unfortunately, this prevents the resulting experience from running on a smaller or differently shaped tracking volume. To address this, we present a software system called Scenograph. The core of Scenograph is a tracking volume-independent representation of real-walking experiences. Scenograph instantiates the experience to a tracking volume of given size and shape by splitting the locations into smaller ones while maintaining narrative structure. In our user study, participants' ratings of realism decreased significantly when existing techniques were used to map a 25m² experience to 9m² and an L-shaped 8m² tracking volume. In contrast, ratings did not differ when Scenograph was used to instantiate the experience.

UIST '18, October 14–17, 2018, Berlin, Germany

ACM ISBN 978-1-4503-5948-1/18/10…$15.00
https://doi.org/10.1145/3242587.3242648

Author Keywords
Virtual reality; real-walking; locomotion.

INTRODUCTION
The most immersive approach to experiencing virtual reality is to allow users to walk around in the tracking volume in a way that maps the virtual world one-to-one to the tracking volume. This approach, known as *real-walking* [30], can lead to higher immersion than in-situ walking (e.g., *treadmills* [5], *walking in place* [28]) and related VR navigation techniques (e.g., *teleportation* [3]).

The typical workflow for designing real-walking experiences is to initially determine the size and shape of the available tracking volume, such as the designer's research lab or some standardized installation like *The Void* [34], and then design the virtual world for that volume accordingly. Because of this, real-walking experiences tend to be specific to the size and shape of the tracking volume they were designed for.

Recent tracking technologies such as Oculus [16] or Vive [33] and online platform such as Steam allow users to bring VR experiences into their preferred environments (i.e., their living room), instead of having to go to the VR setup. Since designers of VR experiences cannot anticipate the amount or shape of users preferred space, the current approach of designing experiences that are tailored to the tracking space becomes impossible.

Creators of VR experiences at home are thus faced with a choice: (1) to artificially narrow down their market by designing for niche tracking volumes or (2) abandon real-walking – creators have picked the latter. Only 0.3% of Steam games use 4m x 4m or more [27], with games like [29] being the exception, and *all* Steam applications require rectangular shape. We find ourselves in a situation where users have paid for a VR system capable of real-walking but have essentially no real-walking contents – they miss out on the extra immersion potentially available.

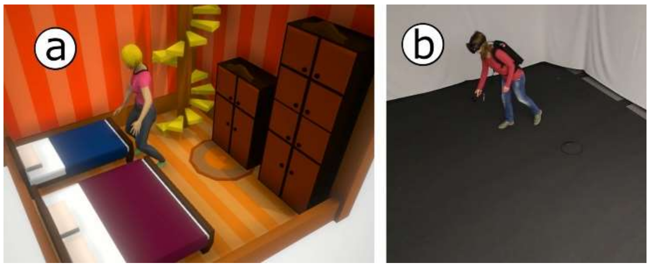

Figure 2: (a) Our adaptation of the fairy tale 'Goldilocks and the Three Bears', in which Goldilocks maliciously enters the home of the three bears, eats their porridges, sits on their chairs and sleeps in their beds. (b) The user in our tracking space of 5m x 5m.

While researchers investigated how to reduce space demands for real-walking [32], their proposed techniques are still based on tailoring the experiences to a specific tracking space. Techniques like *redirected walking* [17], *flexible spaces* [31], or *VirtualSpace* [12] considerably reduce space requirements, but ultimately do not address the problem that applications still assume a tracking volume with well-defined size and shape (e.g., *VirtualSpace* requires $16m^2$ to let individual requirements sink to $4m^2$). A substantial step in the right direction is *Oasis* [23], which enables customization of virtual worlds. Users scan their tracking volume with a depth camera, and from this data, Oasis creates a static virtual location that fits into the space.

In this paper, we present a system, Scenograph, that pushes this idea further, but instead of mapping static location to arbitrary tracking volumes, we map *experiences*. By making real-walking experiences independent of any particular tracking volume, Scenograph provides a crucial component for making real-walking experiences available to consumers.

SCENOGRAPH

Scenograph is a software system that offers a tracking volume-independent representation of real-walking experiences. Instead of designing for a tracking volume of specific size and shape, Scenograph lets designers specify an experience independent of the tracking volume. The virtual world is then automatically generated by Scenograph (while applying space compression techniques like [26]), so that users can run the experience in their individual tracking volume. We demonstrate this process through an example application based on the 19th-century fairytale 'Goldilocks and the Three Bears' ([22], see Figure 2 for our design). Naturally, Scenograph allows for the design of any application that can be procedurally generated, 'Goldilocks' is a good example as the narrative unfolds within one connected environment.

The interface to Scenograph is an editor, in which application designers define the unfolding of their real-walking experience. An experience is the designed arrangement of possible interaction sequences, where users switch between different virtual locations, or *scenes* to be more general. Users experience those scenes by real-walking as each scene has a designed arrangement of objects that users walk between. Scenograph encodes that experience in a bipartite graph (specifically, a petri-net). The instantiated graph maintains the arrangement of virtual scenes and objects.

Goldilocks contains three scenes, a 'dark forest', the 'three bears' home', and 'upstairs bedroom' (see Figure 1). The 'home' scene is connected to three 'porridges' and three 'chairs'. Corridors are used as portals to switch between scenes making the switch unperceivable (like in [26]). Logical elements enable story progression when the user interacts with certain objects, e.g., after eating 'little bear's porridge', the user changes into the 'tired' state, so that users can now interact with the 'chairs'.

The tracking volume-independent representation

Internally, Scenograph represents the experience as a petri-net, as such it has *transitions* and *nodes*. Nodes are either *spatial* or *logical*. The *spatial* nodes are the scenes of the experience. The *logical* nodes are the states that enable story progression. The two kinds of nodes are connected by *transitions*, the virtual objects in the scenes. Transitions pass tokens between input and output nodes. The first 'door' for example, passes tokens from the 'forest' (spatial) and 'curious' (logical) nodes to the 'home' (spatial) and 'hungry' (logical) node. In Figure 3 we see that the 'porridge' then takes these tokens away from the 'hungry' state, but keeps it in the 'house', as the user remains there. A petri-net is suitable for encoding any direction a narrative may take (see Figure 4). This data structure can be expressed in any environment the application is developed in, here we used Unity3D/C# (see implementation section).

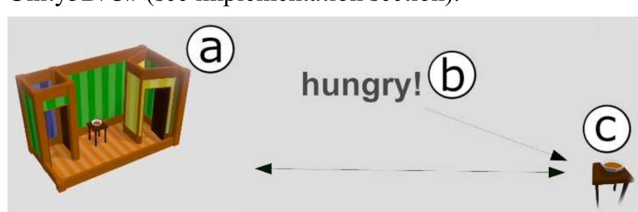

Figure 3: Scenograph encodes all possible interaction sequences in a graph. (a) Spatial nodes define the virtual scenes, here a part of the 'three bears' home'. (b) Logical nodes express states of the user and of objects, here whether the main character Goldilocks is 'hungry'. (c) Transitions, here a 'porridge', let users switch states and scenes, they populate the virtual scenes as objects. The 'porridge' can be interacted with, as both its requirements, 'home' and 'hungry', are satisfied.

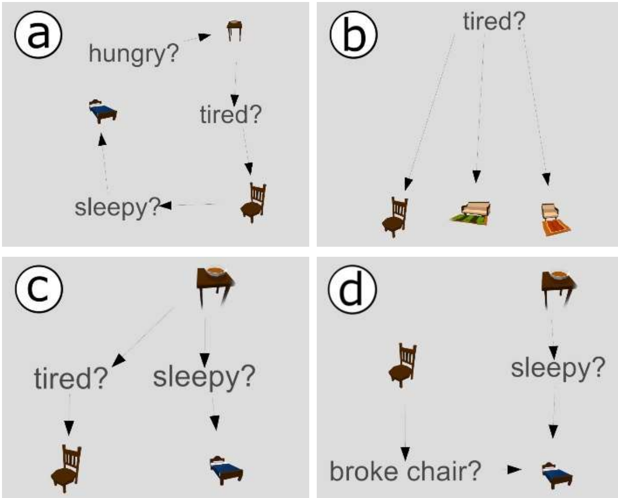

Figure 4: Here are examples of different narrative arrangements, which the petri-net representation allows. (a) Sequence: the 'porridge', 'chair' and 'bed' are accessed in a set order. (b) Conflict: the user needs to decide between one of the 'chairs'. (c) Concurrency: eating the 'porridge' allows using the 'chair' and the 'bed'. (e) Synchronization: sitting on the 'chair' and eating the 'porridge' is necessary before sleeping in the 'bed'. Other progressions are also possible ('confusion', 'merging', etc.).

Scenograph adapts the scenes to the available space by splitting the nodes into multiple instances – this is the core value of the system. As seen in Figure 1, limiting the tracking volume from $25m^2$ to $8m^2$ results in splitting the 'home' node into four nodes. Figure 5 shows in detail how transitions get re-linked to maintain narrative structure. Scenograph takes the petri-net and the available tracking volume as input and transforms them into the new layout.

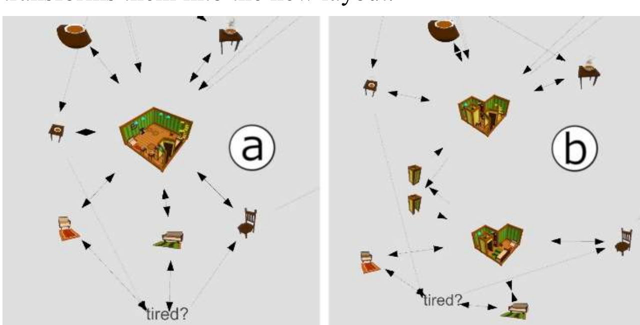

Figure 5: Scenograph splits the 'home' node into two as the designed for $25m^2$ get reduced to an L-shaped $12m^2$. (a) This node has six transitions (three porridges followed by three chairs). (b) The porridges are placed in the first (upper) node, the chairs in the second, as they are interacted with *after* the porridges.

The end-user has no knowledge of Scenograph's data structure. The system merely requires a specification of the user's tracking volume in the form of a polygon. This specification can be provided by a range of tracking technologies. The designer provides the volume-independent representation of the experience.

Authoring an experience

To create an experience, the designer must define all possible interaction sequences, i.e., the connections between spatial nodes, logical nodes, and transitions. This specification follows a bipartite graph structure, as nodes and transitions can only connect to each other, not to themselves. After creating the necessary virtual objects of the experience (3D models, animations, etc.), the designer connects transitions to nodes and vice versa by clicking on the corresponding objects in the editor's graph representation (Figure 6).

Figure 6: The designer authors the Goldilocks experience by connecting nodes and transitions. Here, the user interacts with the 'beds' after 'little bear's chair'. Consequently, the designer connects the 'chair transition to a state ('sleepy'), which is a mandatory state for the 'bed' transitions.

Scenograph can also attribute multiple transitions to the same virtual object. In Figure 6, for example, one side of the 3D model of the big bed is considered 'mama bear's bed', the other side 'papa bear's bed'. A staircase can be used for 'going upwards' and 'going downwards', etc.

The classic Goldilocks fairy tale is sequential (like most stories). Goldilocks eats the porridges, sits on the chairs, then lies on the beds. She always starts with the item of papa bear, then mama bear and finally the small bear. In our rendition of Goldilocks, we instead chose to allow for user decisions: any porridge is edible until small bear's porridge is eaten, sitting on any chair is possible until the user sat on small bear's chair, etc. A sequentially told story in Scenograph would provide an easy to solve problem (cut off the story when we run out of space, then split the location node). Scenograph usefulness increases with the complexity of the narrative, for example, when user decisions are involved, since the problem of where to split the nodes is then non-trivial. On the other end of the spectrum, if there is no logical connection at all (all objects can be interacted with anytime without consequences or story progression), then the decision where to split the node would be arbitrary.

Defining spatial constraints

Scenograph requires its applications to declare the space requirements for its virtual objects the transitions are paired with. Space requirements entail the objects length and width as hard constraints, and placement preferences (close to a wall, middle of the room, etc.) as soft constraints with a cost function.

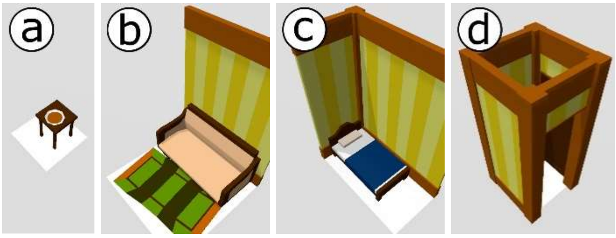

Figure 7: (a) A 'porridge' is a virtual object that requires 1m², (b) 'papa chair' needs 4m² and a wall, (c) 'little bed' 1m x 2m and a corner, (d) a generic corridor-portal connecting scenes takes 1m² and a corner.

Defining available space

Scenograph requires a specification of the given tracking volume in the form of a polygon, as well as the resolution into which the space gets virtualized, which is provided by the application. In our lab setup we have 5m x 5m available, and our example applications is designed for a resolution of 1m x 1m so that Scenograph tessellates the space into 5 x 5 tiles (Figure 8).

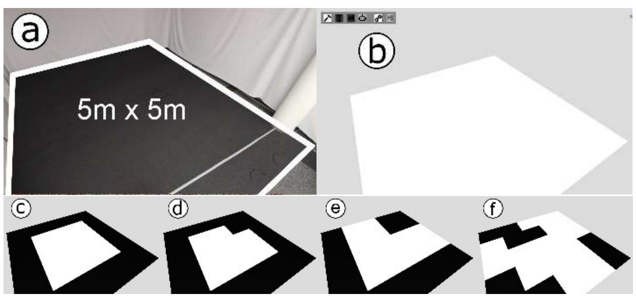

Figure 8: (a) Our tracking volume of 5m x 5m. Since the 'Goldilocks' resolution is 1m x 1m this space is virtualized into (b) 25 tiles. Defining different tracking volumes result in (c) 9 tiles, and (d) L-shaped 8 tiles, the configurations we used for our user study. Many other sizes and shapes are possible (e, f).

The generated chunks of tiles are allocated to the system for the generation of the virtual scenes. Different physical setups will thus result in different scenes (Figure 9). Note that each experience requires a minimum size based on its largest virtual object, e.g., for Goldilocks this is the sofa with 2m x 2m. Scenograph allows to switch setups at runtime, for example if space gets occupied or freed up suddenly. In this case Scenograph re-instantiates the experience, however, it maintains the current logical and spatial nodes (e.g., 'frightened' and 'upstairs bedroom').

Figure 9: Scenograph takes various tracking volumes as input, such as (a) this living room. (b) Scenograph then instantiates the experience.

Testing spatial constraints for given available space

Scenograph needs to determine if all nodes can support their transitions, i.e., if the virtual objects' can be packed together onto the space the scene is given. The system offers different packing algorithms, ("best-fit", using simulated annealing), or random placement ("first-fit", using random placement). Given a smaller or differently shaped tracking volume, the packing algorithm might not find a solution and the node is then not able to support its transitions. In Figure 5 we cannot pack three 'porridges' and three 'chairs' into L-shaped 12m², thus Scenograph splits the 'home' node into two.

Splitting nodes to match spatial constraints

To determine the number of splits per node and the distribution of transition onto split nodes Scenograph uses divisive hierarchical clustering. The distance computation between transition pairs, required for our hierarchical clustering, uses a simple semi-decision technique (distance of 1 if two transitions can be interacted with in any order, 2 if one transition needs to be interacted with after the other, 3 for neither). Scenograph now needs to evaluate which clustering to take for each node.

Scenograph cannot linearly iterate through each node separately to find the right clustering, as some virtual objects need to be instantiated on the same tiles in more than one scene (transitions with different spatial nodes as input and output, such as a door). This means that Scenograph needs to consider all possible clusterings for all nodes in parallel, making the packing problem 3-dimensional (width, depth, occurrence in nodes).

Scenograph iterates through all possible clustering in an informed manner. The number of clusters and therefore the potential connected scenes to consider is exponential in the number of virtual objects. Our hierarchical clustering does not reduce this amount, it merely sorts all potential clusterings based on the conceptual distance between transitions. The number of virtual objects and thus of potential clusters is different for each scene. For example, our three nodes have one transition ('forest' has a door leading to 'home'), eight transitions ('home' has 3 'porridges', 3 'chairs', 1 'door', 1 'stairway') and three transitions ('upstairs bedroom' has 2 'beds', 1 'stairway'), leaving $2^0 + 2^7 + 2^2 = 512$ possibilities. Each possibility corresponds to a certain clustering depth per node, which we represent using a mixed radix numeric system (e.g., $1_1 2_6 1_3$ corresponds to splitting the second node in two). We iterate through this numeric system linearly first based on the checksum of this clusterings number (to reduce the number of nodes) and then on its order within the hierarchical clustering (maximizing proximity of virtual objects that are also conceptually close).

Creating additional transitions between nodes

After finding the right split of nodes, additional transitions need to be generated to connect them (Figure 10). This problem has multiple solutions (e.g., sequence, loop, full connection, etc.). However, as a tradeoff, added transitions lead to a linear decrease of available space per cluster (to a maximum of number of clusters – 1 for full connection). This makes this decision a design problem and consequently the application designer defines how scenes should be connected, our example application uses a sequence.

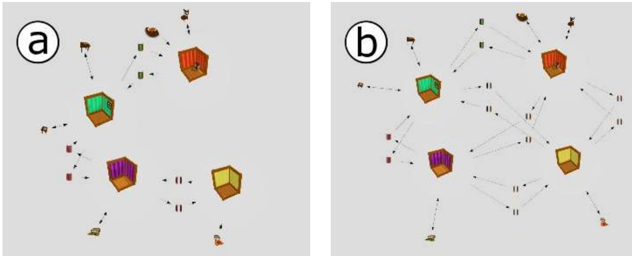

Figure 10: These four $4m^2$ nodes can be linked in (a) a sequence, (b) completely, a loop or any other connection, which the application designer decides on.

Instantiating the experience

The virtual scenes are now generated. Each virtual object is placed onto the space the packing algorithms allocated for it. Afterwards, the rest of the scene is generated. While an application may create the visuals for each scene itself, Scenograph offers some default procedural generation algorithms to create the scene automatically. The application just provides the decorative objects, walls, floors, etc., together with placement constraints (e.g., a wall element with a window cannot be used next to occupied tiles). Scenograph loads and unloads scenes dynamically depending on the users' interaction with the virtual objects or where they walk. Scenograph uses corridors similar to "impossible spaces" to connect scenes [24], which serve as portals or locks. The L-shaped $12m^2$ space in Figure 5 can thus be used twice to fit the 'porridges' as well as the 'chairs'. Less overlap makes the technique less perceptible. However, since we focus on small spaces, Scenograph here fully overlaps the scenes.

Summary of the algorithm

The process of our system can be summarized as followed:

```
s_n = Application.GetSpatialNodes()
for all spatial nodes s_n
    d_n = GenerateDendrogram (s_n)
r = Application.GetResolution()
t = GetAvailableSpace(r)
threshold = Application.PackingThreshold
iterate through cluster possibilities
    c_m = CurrentClustering
    s_nc_m = SplitNodesForClustering(d_n, c_m)
    if(not PackingPossible(s_nc_m))
        reject s_nc_m
    if(PackingValue(s_nc_m) >= threshold)
        pick s_nc_m
for all s_nc_m
    e_n= GenerateScene(s_nc_m)
LinkScenes(e_n, Application.Preference)
```

Extension: Binding in props

Naturally, any requirements of physical objects next to floor space can be defined in the system. Figure 11, for example, shows how Scenograph places 'papa chair' on the position of an actual physical sofa instead of just empty space (if a prop is not bound to a virtual object, its floor space is made unavailable to the application). This concept of dynamically binding in passive haptics is known from Simeone et al. [20], who "substituted" physical objects with virtual counterparts. *Oasis* [23] also refers to this aspect. However, this technique requires a similar physical counterpart for each virtual object, thus narrowing down the scope of possible tracking volumes, which we here want to widen. With the same logic of mapping virtual objects onto arbitrary spaces, future work should investigate mapping virtual objects onto arbitrary objects (see discussion section).

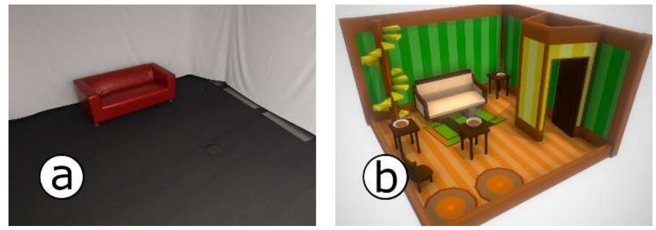

Figure 11: Our system supports defining other physical requirements than floor space, here binding in an (a) actual sofa for (b) 'papa bear's chair'.

Implementation

The system was implemented in C#, the example application and editor interface in Unity3D. ALGLIB [1] was used for clustering. To allow researchers to replicate our work, we provided the full source code online [19].

CONTRIBUTION

Scenograph allows designers of real-walking virtual reality experiences to reach users with tracking volumes of arbitrary size and shape. The core of Scenograph is a tracking volume-independent representation of real-walking experiences. This representation is not spatial until instantiated into tracking volumes of specific size and shape. Scenograph thereby lays the groundwork for real-walking experiences to reach the consumer market.

RELATED WORK

This work borrows from the following fields: real-walking in VR, space compression techniques, preservation of interaction sequences and automated layouting.

Real-walking in VR and space compression techniques

While walking in VR can be enabled by *treadmills* [5], it is more often simulated with techniques such as *walking in place* [28] or *teleportation* [3]. However, real-walking, a one-to-one mapping of physical to virtual motion, leads to the highest user satisfaction [30]. Real-walking has a high space demand, which researchers have tried to reduce with several techniques (for an overview see [32]). *Resetting* [35] rotates and virtually repositions users once they hit the tracking volume's borders. *Seven league boots* [10] scales the virtual motion. In one way or another, these and related techniques perceptibly interrupt or alter the one-to-one mapping of physical to virtual motion, which leads to a reduction of the immersive quality of walking.

With *redirected walking* Razzaque et al. [17] break this one-to-one mapping unperceivably and fold long walking paths into limited tracking space, thus lowering the required amount of space for real-walking. Redirected walking benefits from the a priori knowledge of the virtual environment, onto which the walking paths can be mapped [4, 6, 9, 25]. Even with the constraint of a priori setting of walking paths, this approach is sub-perceptible only for large tracking volumes (e.g., 4m x 10m [17]). For smaller spaces, complementary fallback techniques such as *resetting* [35] are needed, which disrupt the walking experience.

Adapting the virtual world to reduce space demand

Instead of reducing space demand by breaking the mapping of physical to virtual motion, designers can alter or influence the virtual world. *VirtualSpace* [12] packs more people into limited space, reducing the space demand to $4m^2$ per user. Collisions are avoided by design; VirtualSpace requires its applications to adhere to an API to dynamically set virtual obstacles and goals that redirect their respective users. However, this technique still assumes a tracking volume of specific size for its applications (in this case $16m^2$). In *impossible spaces*, Suma et al. [26] lets virtual rooms unnoticeably overlap to compress space. The layout can be dynamically generated, such as in Vasylevska et al.'s [31] *flexible spaces*, or combined with redirected walking [11]. As impossible spaces uses change blindness [24], it requires large amounts of space to be sub-perceptible (9m x 9m [26]). However, even when the overlap is noticeable this technique still enables enjoyable experiences, like the commercially available game "unseen diplomacy", a game with relative success [29] that uses only a relatively small space (4m x 3m). The major drawback here is that this game, like VirtualSpace, was designed with a fixed tracking volume in mind. As argued in the introduction, this makes the game unsuitable for a lot of setups; either it does not make use of possible surplus space of the user or it does not even run at all for tracking volumes that do not fit the required 4m x 3m.

Procedurally generating the virtual world

An important step towards altogether circumventing the problem of a virtual scene not fitting into the tracking volume has been taken in *Oasis* [23]. Sra and colleagues adapted virtual scenes to fit rooms of arbitrary shape. Conceptually, this widens the possibilities for space setups, also to mobile scenarios. This concept has been reused [21] and is also known as procedural content generation (PCG).

Techniques using PCG for real-walking, however, do not take the creation or maintenance of a coherent narrative into account. This points to a problem of PCG in general – it is designed to quickly generate variance in the virtual scenes and is usually not used for story focused experiences. Translating this problem into real-walking applications for VR: when PCG is applied to generate a virtual scene there are two constraints, tracking volume as well as maintaining the virtual narrative.

Preserving a virtual narrative while using procedural content generation

Maintaining a virtual experience using PCG has been applied and is called "experience-driven" PCG [36]. However, does not extend to real-walking but mostly to arcade style games only. For example, Mourato et al. [13] created variations of 2D games (platformers) based on the analysis of original game segments. A main challenge is the "quantification of [user] experience," according to Yannakakis et al. [36]. Nevertheless, PCG has often been applied to space layouting [8], however without real-walking, so that we hope to point to a possible research direction within that area.

Automated layouting in other areas

Preserving experience, or, more specifically, functionality, has been successfully achieved in another area: GUIs. *Supple++* [7] is a tool to automatically generate user interfaces. Functionality here is defined as enabling all "traces", all possible interaction sequences. The tools' objective is to incorporate 2D space constraints while preserving functionality. This directly translates to our constraints, tracking volume limitations and preservation of the narrative and virtual experience. Similar to our virtual object placement, it is "impossible to compute the cost of a layout without knowing all the interface elements comprising that layout". This then requires "modifications to the optimization algorithm". As in Supple (and similar GUI generating systems [2, 15]), we use a high-level structure to maintain all "traces". We widened this representation to petri-nets to able to allow for any kind of narrative, the arrangement of interactions.

Operating Systems

Our system conceptually borrows from the field of operating systems; the software, here our 'Goldilocks' application, is abstracted so that it can run on arbitrary physical hardware, in this case, arbitrary physical tracking volume. Specifically, it *links* compiled and assembled objects together and *loads* them onto physical space, the hardware. This abstraction of hardware and software allows for hardware changes (space variance) and code survivability (preservation of the experience). In this regard, Scenograph performs as an integral part of operating system for real-walking in VR and enables development of software independent of the hardware.

USER STUDY

Scenograph can fit experiences into tracking volumes with different physical sizes and shapes. To validate this, we let Scenograph instantiate our 'Goldilocks' experience for different tracking volumes. We compared these instantiations to two commonly used locomotion techniques for small tracking volumes (see Figure 12): *motion scaling,* the changed mapping of users' physical to virtual motion (similar to *Seven-League-Boots* [10]), and *teleportation*, a form of artificial locomotion [3]. We hypothesized that Scenograph would receive higher ratings of realism and enjoyment.

Interface conditions

We compared six conditions.

In *large-square*, we contained every scene of the narrative in a single volume by allocating a total of 5m x 5m to the application. No split of any location nodes and no motion scaling needed to be applied. This condition emulates the best possible situation as every scene can incorporate all virtual objects.

In *small-square-Scenograph* we allocated 3m x 3m, emulating a smaller tracking volume where splitting the experience is necessary as not all virtual objects fit into the space.

In *small-shape-Scenograph* we allocated the small space in the form of a L-shape ($8m^2$), emulating variable shape of users' tracking volume.

In *small-square-scaled* we rendered the same $25m^2$ virtual world as in *large-square* but applied a motion mapping of 1:5/3 to all translations of the participant so that only 3m x 3m are needed.

In *small-shape*-scaled we also rendered the $25m^2$ virtual world, but for 2m x 2m tracking volume, the smallest quadratic shape fitting into the L-shape of *small-shape-Scenograph*, for a 1:5/2 motion mapping.

In *small-teleport*, we rendered the same $25m^2$ virtual world as in *large-square* but enabled a teleport mechanism: users could point to any virtual position and teleport themselves to it. The limited tracking volume, here our *small* 3m x 3m, was visualized like in most real-walking systems (such as Vive [33], or [18]) by rendering so-called "chaperone bounds".

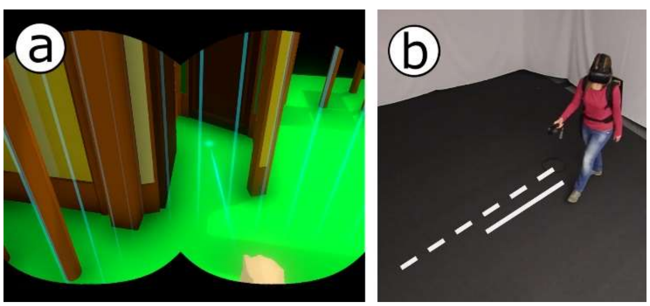

Figure 12: (a) Our first control condition implements a *teleport* functionality and displays chaperone bounds to keep users from leaving the tracking volume. (b) Our second control condition, *scaled motion*, changes the mapping of physical motion (solid line) to virtual motion (dashed line).

Apparatus
The study took place in our 5m x 5m empty lab space (Figure 8). Participants wore a backpack PC for tetherless VR. For tracking we used the VIVE system [33]. The experience we used was the 'Goldilocks' application described earlier. Participants could interact with the virtual objects of the application (*transitions*, see system section) with their hand controller.

Task and procedure
We gave participants a summary of the story and let them experience the VR application before testing, so it was familiar prior to all conditions. Each participant then traversed once through each condition for a total of six sessions. The order of the conditions was randomized for each user. Each session took about two minutes to complete. After each session, participants filled in a questionnaire.

Measurements
The questionnaire contained two statements to complete on a 1-7 Likert scale: "I enjoyed the experience (not at all-very much)" and "Moving around felt (artificial-natural)" for our measures *enjoyment* and *realism*.

Participants
We recruited twelve participants from our organization (six female, six male, mean age 23.6 sd 3.6 years). Seven of the participants had previous experience with VR, two of whom with real-walking. The remaining five participants had never tried VR before.

Hypotheses
Our hypotheses compared Scenograph to both control conditions: For *small* spaces, we hypothesized the *Scenograph* conditions would be perceived as more realistic and enjoyable than the *scaled* conditions for both *square* (**H1)** and *shape* (**H2**). The *Scenograph* conditions would be perceived as more realistic and enjoyable than the *teleport* condition (**H3, H4**). The *small-square-Scenograph* condition would be perceived as less realistic and enjoyable than the *large-square* condition (**H5**).

Results
This section reports Bonferroni-corrected *p*-values. T-tests (applied as suggested by [14]) are one-sided.

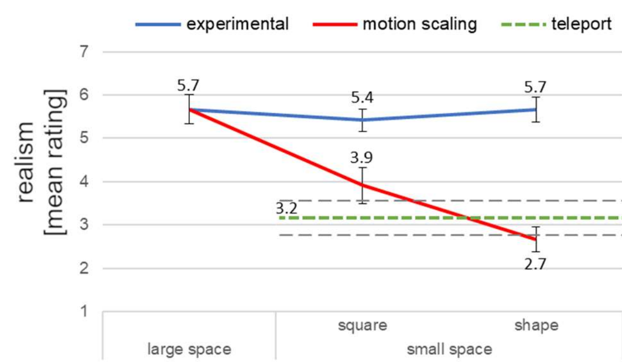

Figure 13: Participants rated the *Scenograph* conditions more realistic than both control conditions *scaled motion* and *teleport* (bars show std. error).

Realism. Figure 13 shows our main finding. As hypothesized, the *Scenograph* condition outperformed the *scaled* conditions in both space setups *square* ($p<.05$, $t(11) = 3.22$), supporting **H1**, and *shape* ($p<.001$, $t(11) = 9.88$), supporting **H2**. *Scenograph* conditions performed better than the *teleport* control (both $p<.01$, $t(11) = 6.23$), supporting **H3** and **H4**. Surprisingly, no difference was found between *large-square* and *small-square-Scenograph* ($p=.73$, $t(11) = 1.27$), so **H5** cannot be supported in terms of realism (underlining the systems effectiveness).

Enjoyment. Enjoyment was generally high, the *large-square* space condition unsurprisingly scored the highest (mean 5.8, sd 0.9) together with *small-teleport* (5.8 sd 1.1). The *Scenograph* conditions performed almost equally well for *small-square* (5.6 sd 0.9), and *small-shape* (5.6 sd 1.1). The *scaled* conditions scored lower for both *small-square* (4.8 sd 1.5) and *small-shape* (4.3 sd 1.5), so that the difference in mean scores was significant for *small-shape* (p<.05, t(11) = 3.61), supporting **H2**. We did not find support for the remaining hypotheses in terms of enjoyment, notably again **H5** (p=.80, t(11) = 1.2).

Meters Walked. For making our results comparable for future studies, we want to report on the distance walked by the users. On average, users walked 37.0m in the *large-square* condition, 28.0m in *small-square-Scenograph*, 29.0m in *small-shape-Scenograph*, 22.3m in *small-square-scaled*, 14.2m in *small-shape-scaled*, 13.4m in *small-teleport*.

Qualitative feedback

Participants found walking experiences generated by Scenograph to be most "realistic" (P3, P7, P8) as movement felt most "natural" (P3, P7). One participant stated that changing rooms was sometimes "irritating" (P2), another found the experience got "harder" (P6).

Participants found teleportation to be "very artificial, but cool" (P6, P12). Others stated that the chaperone bounds broke immersion (P8, P9). One participant argued for its naturalness due to its "familiarity" (P1).

Motion scaling was perceived as "non-intuitive" (P5, P6, P7) and "uncomfortable" (P2, P4, P12). Participants liked the experience better if it was "less fast" (P10). Some participants enjoyed it as it was "like in a movie, but not natural" (P9), though "it became more natural over time" (P8).

DISCUSSION

Our main finding is that Scenograph can create real-walking experiences in VR for tracking volumes of any size and shape. When comparing Scenograph's experiences to two commonly used locomotion techniques, namely *instant teleportation* [3] and scaling the mapping of physical to virtual motion (e.g., [10]), experiences were rated to be more realistic, leading to the conclusion that Scenograph degrades more gracefully with limitations of tracking volume. The collected qualitative feedback supports the findings.

Space compression impacts realism differently than enjoyment. Both teleportation and motion scaling still provided enjoyable experiences to the participants. Also, the size of the tracking volume did not measurably impact enjoyment (comparing 9m^2 to 25m^2). The number of participants might have played into not discovering the hypothesized differences, but it seems that even when space compression is high and becomes more perceivable to the user (teleport, stronger motion scaling, higher overlap of impossible spaces) it affects realism first. Only when the compression becomes very high (motion scaling for L-shaped 8m^2) does it also impact enjoyment. Based on our results, we can thus only make claims about Scenograph's impact on realism.

The study design has some limitations. No set of control conditions exists that can sufficiently represent the broad spectrum of available locomotion techniques. We chose our control conditions as mere representatives of those existing techniques; motion scaling as one instance of locomotion techniques which distort the mapping of physical to virtual motion (instead of, e.g., *redirected walking* [17]) and teleportation as one instance of virtual locomotion techniques (instead of, e.g., *walking-in-place* [28]). It can be argued that virtual locomotion always comes at the cost of perceived realism and that the motion mapping can be optimized, as for example done in *seven-league-boots* [10]. A great way of optimizing is by knowing about the intended destination of travel, which make these techniques particularly suitable for sequential narratives (*seven-league-boots* [10] scales motion and redirected walking [17] alters the yaw rotation towards the direction of travel). For that reason, we imagine that these or similar techniques can play into future iterations of the system. As of now, our study showed that Scenograph is a working system that supports end-users in any tracking volume and compares well to the chosen control conditions.

Scenograph does not improve all real-walking scenarios. The system uses the degrees of freedom that an experience provides. For example, it may not matter to the Goldilocks designer where the "chairs" are in relation to the "porridges" and Scenograph can place them in different scenes. However, if the designer specifies that each chair must be next to its porridge, then the system cannot optimize as there are no degrees of freedom to use. The same holds for any experience with strong spatial constraints, say in a soccer simulation where the goal must be in a fixed relation to the other goal or to the corner flags. Also, if the narrative is unknown before (again: soccer for example), the system has no information to work with.

This work builds on insights of Sra at al.'s *Oasis* [23] and *impossible spaces* [26]. We similarly believe that the only way real-walking can be achieved is by altering or altogether generating the virtual world, a dynamic *mise-en-scène*. Otherwise anytime physicality adapts to virtuality, breaking the motion mapping is needed in one way or another, and realistic perception will be diminished. Of course, further advances are needed to be made in the field of procedural content generation before it is possible to apply it to any kind of virtual experience.

For future work, we imagine integrating not only variance in tracking volume, but variance in the number of users within the tracking volume. While *VirtualSpace* [12] already addressed this this issue, the arcade-style applications did not contain a narrative. Building on *Supple++* [7], we also consider binding in user capabilities and preferences for achieving tighter or looser packing of virtual objects or speeding up narrative and altering story arcs. While binding in physical props (e.g., "substitutional reality" [20]) and automatic layouting (e.g., "flexible spaces" [31]) has already been addressed with regard to real-walking, future work should address this also with regard to narrative structure.

CONCLUSION

We presented Scenograph, a software system that supports the design of real-walking experiences, which can adapt to any tracking volume's size and shape. Scenograph achieves this by representing the experience in a petri-net. Based on this representation and a given tracking volume the system generates a set of virtual scenes that guarantees the preservation of the experience for each user's individually available space. This strategy provides more realistic real-walking experiences than commonly applied techniques for variance in tracking volume, as we demonstrated in our user study. Real-walking experiences are typically designed with a specific tracking volume in mind. Scenograph allows users with constrained tracking volume to also have these experiences.

ACKNOWLEDGEMENTS

We thank all who supported this work, especially Maximilian Brehm, and all study participants for their time.

REFERENCES

1. Alglib, cross-platform numerical analysis and data processing library. Retrieved April 1, 2018 from http://www.alglib.net/
2. Blaine A. Bell and Steven K. Feiner. 2000. Dynamic space management for user interfaces. In *Proceedings of the 13th annual ACM symposium on User interface software and technology* (UIST '00), 239–248. https://doi.org/10.1145/354401.354790
3. Evren Bozgeyikli, Andrew Raij, Srinivas Katkoori, and Rajiv Dubey. 2016. Point & Teleport Locomotion Technique for Virtual Reality. In *Proceedings of the 2016 Annual Symposium on Computer-Human Interaction in Play* (CHI PLAY '16), 205-216. https://doi.org/10.1145/2967934.2968105
4. Haiwei Chen, Samantha Chen, and Evan Suma Rosenberg. 2018. Redirected Walking Strategies in Irregularly Shaped and Dynamic Physical Environments. In *25th IEEE Conference on Virtual Reality and 3D User Interfaces* (VR '18). *Workshop on Everyday Virtual Reality*.
5. Rudolph P. Darken, William R. Cockayne, and David Carmein. 1997. The omni-directional treadmill: a locomotion device for virtual worlds. In *Proceedings of the 10th annual ACM symposium on User interface software and technology* (UIST '97), 213-221. http://doi.org/10.1145/263407.263550
6. Zhi-Chao Dong, Xiao-Ming Fu, Chi Zhang, Kang Wu, and Ligang Liu. 2017. Smooth assembled mappings for large-scale real walking. *ACM Trans. Graph.* 36, 6, Article 211 (November 2017), 13 pages. https://doi.org/10.1145/3130800.3130893
7. Krzysztof Z. Gajos, Jacob O. Wobbrock, and Daniel S. Weld. 2007. Automatically generating user interfaces adapted to users' motor and vision capabilities. In *Proceedings of the 20th annual ACM symposium on User interface software and technology* (UIST '07). ACM, New York, NY, USA, 231-240. https://doi.org/10.1145/1294211.1294253
8. Mark Hendrikx, Sebastiaan Meijer, Joeri Van Der Velden, and Alexandru Iosup. 2013. Procedural content generation for games: A survey. ACM Trans. Multimedia Comput. Commun. Appl. 9, 1, Article 1 (February 2013), 22 pages. http://dx.doi.org/10.1145/2422956.2422957
9. Christian Hirt, Markus Zank, and Andreas Kunz. 2018. Preliminary Environment Mapping for Redirected Walking. In *25th IEEE Conference on Virtual Reality and 3D User Interfaces* (VR '18). IEEE. https://doi.org/10.3929/ethz-b-000253659
10. Victoria Interrante, Brian Ries and Lee Anderson. 2007. Seven League Boots: A New Metaphor for Augmented Locomotion through Moderately Large Scale Immersive Virtual Environments. In *Symposium on 3D User Interfaces*, Charlotte, NC, 2007, pp. https://doi.org/10.1109/3DUI.2007.340791
11. Eike Langbehn, Paul Lubos, and Frank Steinicke. 2018. Redirected Spaces: Going Beyond Borders. In *25th IEEE Conference on Virtual Reality and 3D User Interfaces* (VR '18). Demo.
12. Sebastian Marwecki, Maximilian Brehm, Lukas Wagner, Lung-Pan Cheng, Florian Mueller, and Patrick Baudisch. 2018. VirtualSpace - Overloading Physical Space with Multiple Virtual Reality Users. In *Proceedings of the 2018 Conference on Human Factors in Computing Systems (CHI '18).* https://doi.org/10.1145/3173574.3173815
13. Fausto Mourato, Fernando Birra, and Manuel Próspero Dos Santos. 2013. The Challenge of Automatic Level Generation for Platform Videogames Based on Stories and Quests. In *Proceedings of the 10th International Conference on Advances in Computer Entertainment - Volume 8253* (ACE 2013), Dennis Reidsma, Haruhiro Katayose, and Anton Nijholt (Eds.), Vol. 8253. Springer-Verlag New York, Inc., New York, NY, USA, 332-343. http://dx.doi.org/10.1007/978-3-319-03161-3_24
14. G. Norman: Likert Scales, level of measurements, and the "laws" of statistics. In *Advances in Health Sciences Education* 15, 5, 625-632, 2010. https://dx.doi.org/10.1007/s10459-010-9222-y
15. Jeffrey Nichols, Brad A. Myers, Michael Higgins, Joseph Hughes, Thomas K. Harris, Roni Rosenfeld, and Mathilde Pignol. 2002. Generating remote control interfaces for complex appliances. In *Proceedings of the 15th annual ACM symposium on User interface software and technology* (UIST '02). ACM, New York, NY, USA, 161-170. DOI: https://doi.org/10.1145/571985.572008
16. Oculus System. Retrieved April 1, 2018 from https://www.oculus.com/

17. Sharif Razzaque, Zachariah Kohn, and Mary C. Whitton. 2001. Redirected walking. In *Proceedings of EUROGRAPHICS* 9, 105-106.

18. Anthony Scavarelli and Robert J. Teather. 2017. VR Collide! Comparing Collision-Avoidance Methods Between Co-located Virtual Reality Users. In *Proceedings of the 2017 CHI Conference Extended Abstracts on Human Factors in Computing Systems* (CHI EA '17), 2915-2921. https://doi.org/10.1145/3027063.3053180

19. Scenograph, Online Repository. Retrieved July 3, 2018 from https://github.com/sebastianmarwecki/Scenograph

20. Adalberto L. Simeone, Eduardo Velloso, and Hans Gellersen. 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 (CHI '15). ACM, New York, NY, USA, 3307-3316. DOI: https://doi.org/10.1145/2702123.2702389.

21. Keng Hua Sing and Wei Xie. 2016. Garden: A Mixed Reality Experience Combining Virtual Reality and 3D Reconstruction. In *Proceedings of the 2016 CHI Conference Extended Abstracts on Human Factors in Computing Systems* (CHI EA '16), 180-183. https://doi.org/10.1145/2851581.2890370

22. Robert Southey. 1839. The Story of The Three Bears. In *Fairy Tales and Other Traditional Stories*, Lit2Go Edition. Accessed April 1, 2018 from http://etc.usf.edu/lit2go/68/fairy-tales-and-other-traditional-stories/5105/the-three-bears/.

23. Misha Sra, Sergio Garrido-Jurado, Chris Schmandt, and Pattie Maes. 2016. Procedurally generated virtual reality from 3D reconstructed physical space. In *Proceedings of the 22nd ACM Conference on Virtal Reality Software and Technology* (VRST '16), 191-200. https://doi.org/10.1145/2993369.2993372

24. Evan. A. Suma, Seth Clark, David Krum, Samantha Finkelstein, Mark Bolas and Zachary Warte. 2011. Leveraging change blindness for redirection in virtual environments. In *2011 IEEE Virtual Reality Conference*, Singapore, 2011, pp. 159-166. https://doi: 10.1109/VR.2011.5759455

25. Qi Sun, Li-Yi Wei, and Arie Kaufman. 2016. Mapping virtual and physical reality. *ACM Trans. Graph.* 35, 4, Article 64 (July 2016), 12 pages. https://doi.org/10.1145/2897824.2925883

26. Evan A. Suma, Zachary Lipps, Samantha Finkelstein, David M. Krum, and Mark Bolas. 2012. Impossible Spaces: Maximizing Natural Walking in Virtual Environments with Self-Overlapping Architecture. *IEEE Transactions on Visualization and Computer Graphics* 18, 4 (April 2012), 555-564. http://dx.doi.org/10.1109/TVCG.2012.47

27. Steam VR blog post from 10 Nov, 2017. Retrieved April 1, 2018 from https://steamcommunity.com/app/358720/discussions/0/350532536103514259/?ctp=3

28. James N. Templeman, Patricia S. Denbrook and Linda E. Sibert. 1999. Virtual Locomotion: Walking in Place through Virtual Environments. In *Presence* 8(6), 598-617. https://doi.org/10.1162/105474699566512

29. Unseen Diplomacy VR real-walking game. Retrieved April 1, 2018 from http://store.steampowered.com/app/429830/Unseen_Diplomacy/

30. Martin Usoh, Kevin Arthur, Mary C. Whitton, Rui Bastos, Anthony Steed, Mel Slater, and Frederick P. Brooks, Jr. 1999. Walking > walking-in-place > flying, in virtual environments. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques* (SIGGRAPH '99), 359-364. https://doi.org/10.1145/311535.311589

31. Khrystyna Vasylevska, Hannes Kaufmann, Mark Bolas and Evan A. Suma. 2013. Flexible spaces: Dynamic layout generation for infinite walking in virtual environments. In *2013 IEEE Symposium on 3D User Interfaces* (3DUI'13), 39–42. https://doi.org/10.1109/3DUI.2013.6550194

32. Khrystyna Vasylevska and Hannes Kaufmann. 2017. Compressing VR: Fitting Large Virtual Environments within Limited Physical Space. In *IEEE Computer Graphics and Applications*, vol. 37, no. 5, pp. 85-91, 2017. https://doi: 10.1109/MCG.2017.3621226

33. Vive and Vive Trackers. Retrieved April 1, 2018 from https://www.vive.com/us/vive-tracker/

34. The Void. Retrieved April 1, 2018 from https://www.thevoid.com/

35. Betsy Williams, Gayathri Narasimham, Bjoern Rump, Timothy P. McNamara, Thomas H. Carr, John Rieser, and Bobby Bodenheimer. 2007. Exploring large virtual environments with an HMD when physical space is limited. In *Proceedings of the 4th symposium on Applied perception in graphics and visualization* (APGV '07), 41-48. https://doi.org/10.1145/1272582.1272590

36. Georgios N. Yannakakis and Julian Togelius. 2011. Experience-Driven Procedural Content Generation. *IEEE Trans. Affect. Comput.* 2, 3 (July 2011), 147-161. http://dx.doi.org/10.1109/T-AFFC.2011.6