Space Folding in Virtual Reality

Joe Strout
Colorado State University
Fort Collins, United States
joe@strout.net

ABSTRACT

UPDATED—December 13, 2019. Abstract here will describe the VR space folding technique, summarizing background, methods, results, and conclusion. Aim for about 150 words.

Author Keywords

virtual reality;travel;nonphysical spaces

CCS Concepts

•**Human-centered computing** → **Human computer interaction (HCI);** *Haptic devices;* User studies; Please use the 2012 Classifiers and see this link to embed them in the text: https://dl.acm.org/ccs/ccs_flat.cfm

INTRODUCTION

Real walking is widely regarded as the most natural and direct travel technique. It provides vestibular cues, requires no training, promotes spatial understanding, and is especially efficient at maneuvering tasks. However, it is not widely adopted in consumer VR applications primarily because of the limits of the user's physical area. Current consumer headsets are designed for indoor use only, and most users do not have an indoor space larger than a few meters on a side.

While many approaches have been developed to "compress" a larger virtual space into a smaller physical one (see Related Work), various limitations make these difficult to apply to a room-scale VR setup of say 3 by 4 meters area, or limit the amount of extra space gained to a modest factor of less than 2X.

Here we introduce an approach to spatial compression that allows for any compression factor, works in a space as small as 2 by 3 meters, and works with completely natural walking, without redirection. Of course there are always trade-offs; the technique presented works best for indoor virtual scenes, and makes no attempt to keep the user from noticing the spatial manipulation. We present an experiment to show that this does not detract from the user's ability to navigated the virtual environment, or in overall user satisfaction.

CHI'20, April 25–30, 2020, Honolulu, HI, USA

ISBN 978-1-4503-6708-0/20/04…$15.00

DOI: https://doi.org/10.1145/3313831.XXXXXXX

RELATED WORK

Here we review previous work. This could be part of the Introduction section, or it could be a separate section as we have it here.

Vaslevska & Kaufman [7] reviewed several techniques for compressing virtual environments to enable natural locomotion within a limited physical space.

Field and Vamplew [1] extended earlier redirected walking work with additional algorithms to redirect the user to the center of the physical space, and to keep the user on circular paths that anticipate possible future turns. (No actual VR experiments were used in testing this system.)

Experiments comparing four redirected walking algorithms were carried out by Hodgson and Bachmann [2]. Subjects walked within a 25m x 45m facility under either normal conditions, or while using redirection. The authors also did simulations of agents moving at 1m/s, considered typical of VR users. Their experiments determined that a 35m x 35m space would be needed for such locomotion in an unconstrained virtual environment.

Nilsson et al reviewed 15 years of redirected walking research, summarizing the state of the field in 2018 [3].

A very different approach to compressing a virtual environment is change blindness redirection, explored by Suma et al [5]. With this technique, the layout of some part of the VE is changed when the user isn't looking at it. For example, a doorway may be moved from one wall to another while the user is looking away or their view is occluded. The technique produces a powerful illusion of a large VE, but requires the user to follow particular paths; it does not allow for free exploration.

Suma and Krum introduced the term *impossible spaces* to describe a closely related technique, in which rooms are spatially expanded while not in view, thus producing a self-overlapping architecture [6]. The authors were intent on maintaining the illusion of a natural space, and found that rooms could overlap by as much as 56% before users began to notice. The present work could be considered an extreme version of impossible spaces, in which the objective to keep users from noticing is abandoned, and a much higher overlap factor is obtained.

While many spatial compression studies focus on measuring the ability of users to detect the manipulation, Peck et al [4] instead compared several measures of navigability for redirected

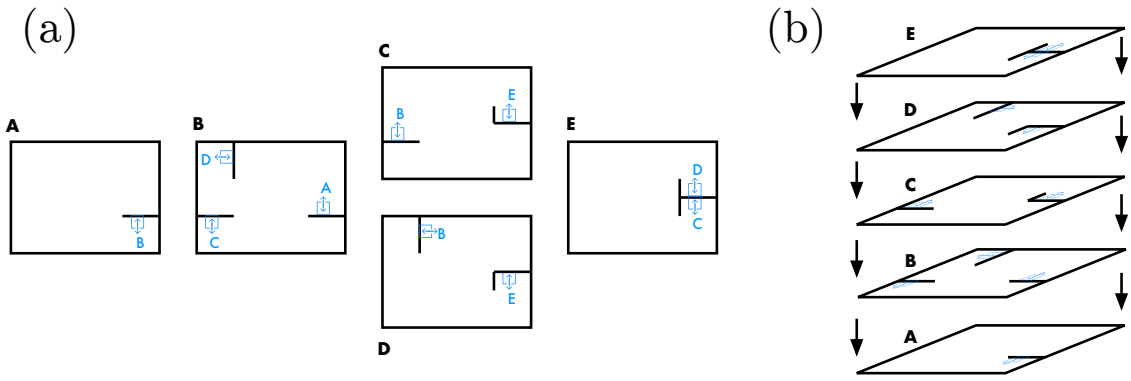

Figure 1. Illustration of the space folding technique. Each "room" of the virtual environment contains portals that connect to other rooms, allowing both viewing and travel in both directions. The layout shown here was used in the pilot study. In (a) the five rooms are shown in top-down view, separated for clarity; (b) illustrates how these virtual spaces are "folded" into one physical space.

walking (with distractors to create redirection opportunities), walking-in-place, and joystick locomotion. (In the latter case, travel speed was proportional to joystick deflection, with a top speed of 3 mph.)

METHODS

Technique

An illustration of the VR Space Folding technique is shown in Figure 1. The central idea is relatively simple: the virtual environment (VE) is divided into spaces ("rooms"), joined by portals. Stepping through a portal takes the user to a different virtual space, which occupies the same physical space as the previous room. In this way, an arbitrarily large VE can be "folded" into a restricted physical environment.

While the core idea would work with opaque portals, the experience is enhanced by portals that can be seen through. For example, when standing in the southeast corner of the physical space while in room *A* of Figure 1 (assuming north is towards the top of the diagram), and looking north, one would see part of room *B*, just as if looking through an ordinary arch or open doorway. Stepping through that doorway, and then turning around and looking south, one would see the corner of room *A* just vacated.

The technique places two important constraints on the placement of the connecting portals:

1. Because the user must be able to physically walk through them, portals may not be placed on the outside boundary of the physical space. Instead they must turned perpendicular to that boundary, or otherwise moved away from the outside edge so that the user has room to walk on both sides of each portal.

2. Each portal has a specific location in physical space, which must be the same as its position in *both* virtual spaces it connects. We cannot, for example, connect the west side of one room to the east side of another.

However, within these constraints, portal placement is otherwise free. In this pilot study we have placed all portals orthogonal to the room walls, but that is not necessary. Within a single virtual room, a portal is one-sided, as in the aforementioned portal between rooms A and B; this is traversed only northward from room A, or southward from room B. But two portals may be placed back-to-back, as illustrated by the portals from room E to rooms C and D. In this example, looking through the archway from the south one would see room C; but looking through it from the north, one would see room D.

The set of connected virtual rooms form an undirected, connected graph. The graph size is the number of rooms, and its order is equal to the largest number of portals in any room. In most applications this order is likely to be relatively low, as each portal takes up roughly $2m^2$ of physical space. We can therefore enumerate the possible room graphs. An order 1 graph (i.e. no more than one portal per room) is always exactly two rooms. There are two possible order 2 (two portals per room) graphs, regardless of size: either all rooms connected in a line, or the end rooms also joined to form a loop.

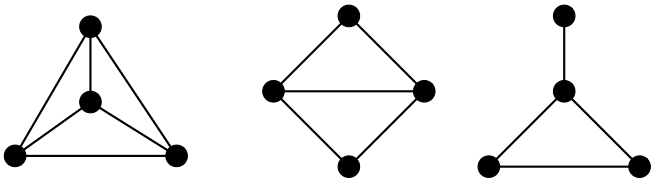

Figure 2. All possible order-3 graphs of 4 nodes. These are the possible topologies for a VE with 4 rooms and up to 3 doors per room.

Once we allow order 3 (up to three portals per room), the design possibilities expand considerably. The smallest order-3 graph has four nodes, and there are exactly three such graphs, shown in Figure 2. Any arrangement of four rooms, with no more than three portals per room, will be isomorphic to one of these. Similarly, with five rooms, the layout will be isomorphic to one of the eight graphs shown in Figure 3. Larger or higher-order graphs are of course possible too. While there is in principle no limit to the size or order of the graph layout one could use, order 3 is the smallest order which allows a wide variety of possible designs, and will be taken as a minimal requirement when considering implementation details.

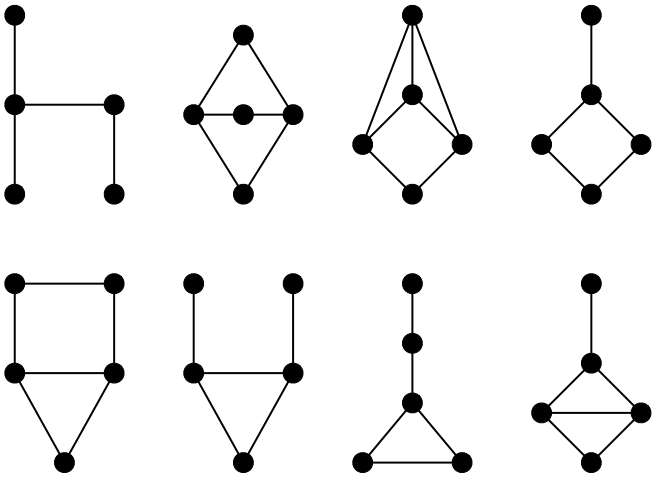

Figure 3. All possible order-3 graphs of 5 nodes. The room layout shown in Figure 1 and used in the pilot study corresponds to the top-right graph above.

Implementation Details

To implement the space folding technique in VR requires a portal solution that meets the following requirements:

1. It must work in VR on the platform of choice, allowing users to see through the portal with proper stereo.

2. It must allow for at least three portals in the same virtual space. Note that it is not strictly necessary to allow for a portal to be visible through another portal; proper arrangement of occluders (e.g. walls) can avoid the need for that. However, a portal solution that allows for such secondary visibility would reduce constraints on the design of the space.

For this pilot study, the Unity game engine and Oculus Quest headset were chosen. However, after seeking and evaluating all available off-the-shelf portal solutions, none were found that could meet these criteria. Most portal solutions did not work in VR at all, or did not work on the Quest; and one allowed for only a single portal. A custom portal solution was therefore written for this study.

The portals combine a custom shader with a script to configure materials using that shader for each portal in view. The shader limits its drawing to a certain region of the world, specified by material properties. These include *CutPoint* (a 3D vector), and *CutSign* (a 4D vector). For each axis (X, Y, and Z), the shader will use the corresponding element of *CutSign* to control how drawing is limited relative to *CutPoint*:

1: draw only geometry *ahead* of *CutPoint* on this axis 0: don't cut on this axis (i.e. draw everything) -1: draw only geometry *behind CutPoint* on this axis

In addition, the shade has properties *LineBound*1 and *LineBound*2, each a 4D vector, which define two lines in the world. The fourth element of *CutSign* controls drawing relative to these lines:

1: draw only to the right of *LineBound*1 and left of *LineBound*2 0: don't limit drawing by the line bounds -1: draw only to the left of *LineBound*1 and right of *LineBound*2

Each virtual room is assigned a unique material that uses the custom shader. A script on each portal then configures the materials for its "local" and "remote" rooms as follows: *CutPoint* is set to the center of the portal; the line bounds are set to the left and right edges of the portal; and *CutSign* is set so that the remote material is drawn only for the frustum of space through the portal, and the local material is drawn everywhere except that same frustum.

The effect of this shader can be seen in Figure 4.

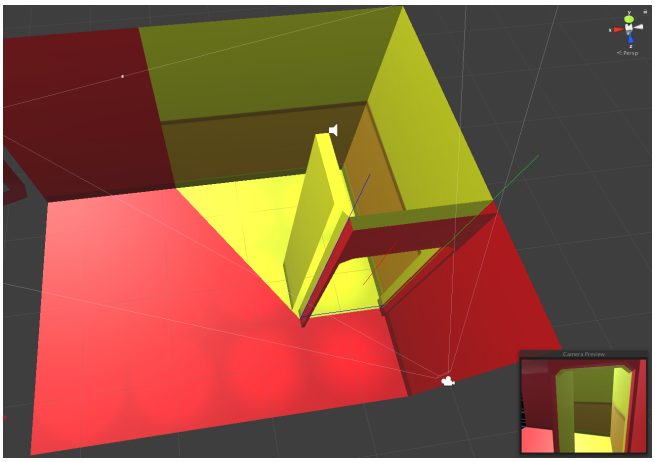

Figure 4. Effect of the world-limit shader. Player's viewpoint is indicated by the camera icon in the corner of the space. The local material is here colored red, and the remote (seen through the portal) material is colored yellow. Inset shows the view from the player's perspective.

Pilot Study

RESULTS

DISCUSSION

- Limitations of shader approach used.

- Other technical limitations.

- Limitations of the pilot study.

- Implications for VE design; when this approach is (and is not) appropriate. - needs at least 3 x 2 meters - works best for indoor environments - very large VE's might get tedious, but - shortcuts (normal portals, plus teleporters/elevators) are possible

CONCLUSION

- Summary of approach.

- Summary of findings.

- Possible directions for future work.

ACKNOWLEDGMENTS

Here thanking everybody who helped out.

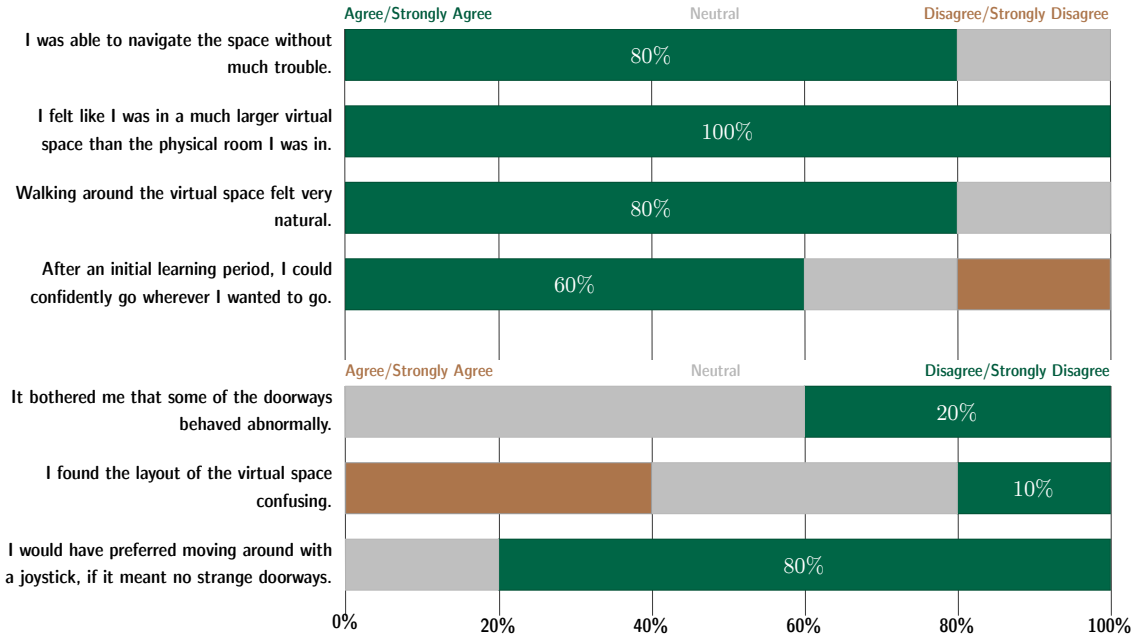

Figure 5. Post-trial survey responses collected in the pilot study. Positive (top) and negative (bottom) statements have been grouped here, but were interleaved in the actual questionnaire.

REFERENCES

[1] Tom Field and Peter Vamplew. 2004. Generalised algorithms for redirected walking in virtual environments. (2004).

[2] Eric Hodgson and Eric Bachmann. 2013. Comparing four approaches to generalized redirected walking: Simulation and live user data. *IEEE transactions on visualization and computer graphics* 19, 4 (2013), 634–643.

[3] Niels Christian Nilsson, Tabitha Peck, Gerd Bruder, Eri Hodgson, Stefania Serafin, Mary Whitton, Frank Steinicke, and Evan Suma Rosenberg. 2018. 15 years of research on redirected walking in immersive virtual environments. *IEEE computer graphics and applications* 38, 2 (2018), 44–56.

[4] Tabitha C Peck, Henry Fuchs, and Mary C Whitton. 2011. An evaluation of navigational ability comparing Redirected Free Exploration with Distractors to Walking-in-Place and joystick locomotio interfaces. In *2011 IEEE Virtual Reality Conference*. IEEE, 55–62.

[5] 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*. IEEE, 159–166.

[6] 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 (2012), 555–564.

[7] Khrystyna Vasylevska and Hannes Kaufmann. 2017. Compressing VR: Fitting large virtual environments within limited physical space. *IEEE computer graphics and applications* 37, 5 (2017), 85–91.

```
Shader "Custom/WorldLimit" {
  Properties {
    _Color ("Color", Color) = (1,1,1,1)
    _MainTex ("Albedo (RGB)", 2D) = "white" {}
    _Glossiness ("Smoothness", Range(0,1)) = 0.5
    _Metallic ("Metallic", Range(0,1)) = 0.0
    _CutPoint ("CutPoint", Vector) = (0, 0, 0, 0)
    _LineBound1 ("LineBound1", Vector) = (0,0,10,10)
    _LineBound2 ("LineBound2", Vector) = (0,0,10,10)
    _CutSign ("CutSign", Vector) = (1, 0, 1, 0)
  }
  SubShader {
    Tags { "RenderType"="Opaque" }
    LOD 200

    CGPROGRAM
    #pragma surface surf Standard fullforwardshadows
    #pragma target 3.0

    sampler2D _MainTex;

    struct Input {
      float2 uv_MainTex;
      float3 worldPos;
    };

    half _Glossiness;
    half _Metallic;
    fixed4 _Color;
    float3 _CutPoint;
    float4 _LineBound1;
    float4 _LineBound2;
    float4 _CutSign;

    UNITY_INSTANCING_BUFFER_START(Props)
    UNITY_INSTANCING_BUFFER_END(Props)

    void surf (Input IN, inout SurfaceOutputStandard o) {
      float3 axisClip = (IN.worldPos - _CutPoint) * _CutSign * _CutSign.w;
      float4 line1Clip =
        ((IN.worldPos.x - _LineBound1[0]) * (_LineBound1[3] - _LineBound1[1])
         - (IN.worldPos.z - _LineBound1[1]) * (_LineBound1[2] - _LineBound1[0]));
      float4 line2Clip =
        ((IN.worldPos.z - _LineBound2[1]) * (_LineBound2[2] - _LineBound2[0])
         - (IN.worldPos.x - _LineBound2[0]) * (_LineBound2[3] - _LineBound2[1]));

      clip(_CutSign.w * max(axisClip, max(line2Clip, line1Clip)));

      fixed4 c = tex2D (_MainTex, IN.uv_MainTex) * _Color;
      o.Albedo = c.rgb;
      o.Metallic = _Metallic;
      o.Smoothness = _Glossiness;
      o.Alpha = c.a;
    }
    ENDCG
  }
  FallBack "Diffuse"
}
```

Figure 6. Source code for the world-limit shader used as the key component of the custom portal solution used in the pilot study.