Towards Cyber-Physical Holodeck Systems Via Physically Rendered Environments (PRE's)

Veljko Krunic, Richard Han
University of Colorado, Boulder
krunic@ieee.org, Richard.Han@colorado.edu

Abstract

We present an early vision of a cyber-physical environment in which computer controlled rendering of physical surfaces, terrains, and environments is achieved by manipulating grids of "moving physical pixels" or "moxels", whose heights can be raised and lowered on command. A user would be free to walk within such a dynamically deformable physically rendered environment (PRE). The system would be able to create on demand the floor, ceiling and sides of the "Holodeck" in which the person resides, and vary the slipperiness of the surfaces to provide authentic tactile feedback in real time. A user would be able to conduct a walking tour of a remote site, the experience of climbing a steep mountain, or navigation through a maze - all within the bounds of the Holodeck. Multiple users could be networked together to create a distributed cyber-physical system to enable team-based distributed training. We present early results in a Holodeck-like simulator, HoloSim.

1. Introduction

The "Holodeck" of StarTrek fame presented the vision of a room in which an immersive cyber-physical world is created around a user by a computer, which is capable of creating arbitrary objects and terrain of high realism in real time. In addition, the user was free to move through such an environment, and objects could be materialized seemingly out of thin air. While this complete vision is still science fiction, this paper offers a pathway towards achieving a Holodeck by describing how the technology of today can be used to generate physically rendered environments (PREs)[1] of emulated three-dimensional terrains with geometric and tactile realism.

Figure 1 shows a PinPoint toy that consists of a grid of rod-like physical pixels that are free to move in the up/down direction. If a user presses their hand on the bottom of the grid, the physical pixels would raise in the approximate 3D shape of the hand, as shown in the figure.

Figure 1. PinPoint toy with the impression of a human hand rendered via moving physical pixels, or "moxels".

By augmenting this pixel-based motion with computer controlled actuation of the physical pixels in real time, dynamic surfaces, terrains, and even forms of motion, e.g. waves, would be capable of being rendered, as shown in Figure 2. We term computer-actuated moving physical pixels as *moxels*[1]. Over the size of a room, deformation of the ground, walls, and ceiling would simultaneously create entire 3D physically rendered environments within which a user could stand and move. The realism would be further enhanced by equipping the tip of each moxel with a surface, such as a ball bearing, disc or ring, whose coefficient of friction could be varied. This would provide tactile feedback that approximates the slipperiness of actual physical surfaces. Combining the PRE with complementary technology such as virtual or augmented reality, e.g. immersive graph-

1 Our colleague Dale Lawrence coined the term "moxel" for a moving pixel after hearing our idea

1545-0678/08 $25.00 © 2008 IEEE
DOI 10.1109/ICDCS.Workshops.2008.31

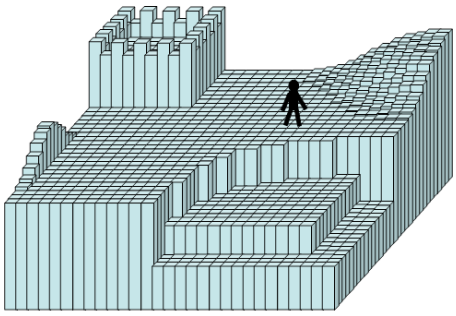

Figure 2. A physically rendered environment (PRE) or Holodeck consisting of a grid of computer-controlled physical moxels or pixels whose displacements can be raised and lowered.

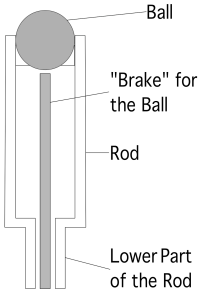

Figure 3. Each moxel consists of a rod whose height can be raised and lowered. The tip of each moxel provides a mechanism for controlling the coefficient of friction, in this case a ball whose rotation can be adjusted or braked.

ics visualization via projection and/or head-mounted helmet displays [8], would enable the creation of cyber-physical systems that provide unprecedented levels of tactile realism and freedom of movement to simulate telepresence.

As shown in Figure 2, a user would be able to stand within this computer controlled physically rendered environment on top of the moxels themselves. The user's freedom of movement would not be limited, i.e. the user would be free to kneel, roll, jump or even climb on dynamically rendered surfaces.

Figure 3 illustrates a single physical pixel or moxel. On top of the moxel is either a ball or a surface whose coefficient of friction can be varied. In the case of a ball, its ability to rotate and/or rate of rotation would be adjustable, e.g. by braking, to give varying degrees of resistance. When both the height of each moxel and the spin of each moxel's ball are computer controlled, we would control not only the shape of the terrain, but also the slipperiness of the surface under the users feet. This would allow simulation of an icy cave, or a rough climbing surface.

Key examples of motivating applications of the PRE include the following:

1. Remote tours of landmarks and facilities that are distant in place and/or time. For example, a user could climb the steps of the Great Pyramid, hike the Great Wall, navigate through a remote factory assembly plant, or walk through a famous golf course (potentially in concert with playing a simulated golf game).

2. Multiple remote users could be networked together to create a distributed cyber-physical system to enable team-based distributed training. Possibilities include training of firefighters, SWAT teams, search-and-rescue operations, and military combat.

3. The physical size of the PRE could be as small as a table or as large as a building. In the latter case, physically proximate teams could conduct training in a physically morphing environment, e.g. firefighting where walls collapse behind a group, lasertag, or even diving in a water-filled PRE.

4. In training or group-based exercises, the cyber-physical PRE can introduce physical obstacles at critical times, e.g. icy conditions, forcing tripping, etc. This would improve training of people's reactions to falls in combat or other life-threatening conditions, similar to how pilots are trained in flight simulators[13].

5. Interactive gaming in one's living room. The PRE could be programmed to simulate a virtual environment, say a subterranean cave that never existed, a fairy tale castle, or a futuristic city.

6. Assistive technology. A scaled down version of this system the size of a box could be used as assistive technology, allowing a blind person to perceive 3D depth and shape.

This paper is *early* work that is intended to raise interest in the vision of computer controllable deformable terrains. Many questions about practical implementation of this vision, its limitations, price and practical applicability still remain open. However, we believe that a prototype of this vision is achievable with the technology available today, both on the software and hardware side.

2. System Architecture

To realize the vision of a Holodeck through physically rendered environments, we intend to employ a high-level system architecture such as depicted in Figure 4. The system consists of both a software component for issuing commands to control the displacement of the moxels, and actuation technology for raising/lowering the moxels. Building such a system will require substantial innovations both in computer science and electro-mechanical engineering.

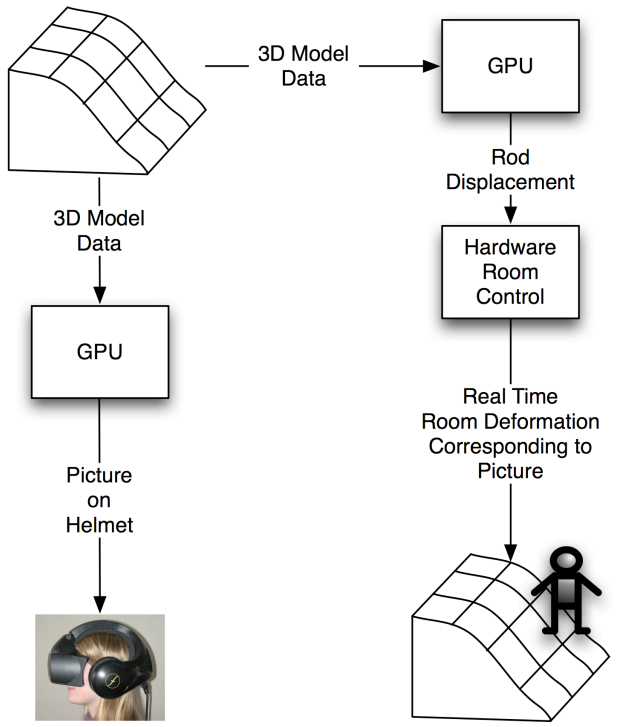

Figure 4. System Architecture

The system begins by creating a 3D model in software of the environment to be rendered. The 3D model must contain physical characteristics of surfaces being modeled, including shape, texture and slipperiness. From the 3D model, we extract the sequence of actions needed to render the physical surfaces in the environment. From the same 3D model, we can generate both graphical images that are shown within a user's helmet-mounted display as well as corresponding physical terrains that are rendered within the PRE, thereby providing an even deeper sense of immersion. Thus, two coordinated rendering paths emerge from the same core 3D model.

The example in the figure demonstates deformation of only one plane, e.g. the ground plane, but the concept is straightforward to extend to deforming other edges of the room beside the floor, e.g. ceiling and walls. Thus, the physical rendering engine may be drawing six different surfaces, or even more if internal PRE objects are factored into the scene.

The Graphics Processing Unit (GPU) normally takes specific graphics commands and renders them on screen. The GPU renders specific types of compute-intensive commands more quickly than generic processors. The next section proposes ways to adapt GPUs to support PRE.

3. Basis of Software Control

This section describes the basis of software control of the grid of moxels.

3.1. Z Buffer and GPU Usage

Our first observation is that the Z-Buffer found in typical graphical systems can be exploited to help us identify and render physical surfaces. In a standard graphical environment, where a viewer is looking at a rendered 3-D scene, the Z-Buffer contains the distance of each pixel from the viewer. If we position the point of view of the user to look up from the bottom of the scene, and use orthogonal projection to render the scene, then the Z-Buffer would contain the distance of each pixel from the floor, i.e. the Z-Buffer contains the height of each moxel on the physical surface that we desire to render. Given a standard 3D graphical model, we need only specify the point of view of the user as being from the bottom looking up in an orthogonal projection, and read out the values from the Z-buffer, and then raise each moxel in the PRE to the appropriate height from the floor.

An important outcome of this observation is that it allows us to use hardware acceleration available in conventional GPUs to calculate moxel position, as well as standard APIs like OpenGL [9] for controlling this calculation.

3.2. Adapting to Physical Limitations

Each moxel is a physical entity and subject to physical laws of inertia. While the Z-Buffer approach can help us calculate the final position of the moxel, it does not account for how the moxel reaches that final position from its current position. The moxel itself is subject to the inertia, as well as physical limitations of the physical system implementing it that need to be accounted for.

Our solution is to utilize programable pixel (fragment) shaders, allowing us to perform per pixel calculations [10]. We would perform quantization of the response function into a texture, and then pass to the fragment shader that texture and a variable representing the interpolation step that we want the fragment shader to perform.

3.3. Slipperiness Simulation

Our discussion thus far has related to how to control displacement of the moxel. When it comes to controlling slipperiness of terrain, typically no information presented to the OpenGL pipeline for the purpose of scene rendering could help us to determine the slipperiness of the surface.

Fortunately, we can encode slipperiness of the surface in the texture, and then use the fragment shader to vary slipperiness of each ball on top of each moxel in accordance with the value of "slip texture" for that pixel. Implementation of this is fairly straightforward in the fragment shader, and calling program can be modified to provide surface slipperiness information. That information could be passed in the form of the coefficients of frictions in a texture map equivalent, that then would be used by the Holodeck's fragment shader. In the long run, material libraries in the modeling packages artists are using to specify look of the objects [15] could be extended to include a slipperiness map of the surface, too.

3.4. Integration with Rendering Engines

Interception of OpenGL calls using [11] and employ a system like AspectC++ [12] is beneficial. For the basic integration of Holodeck in simple graphic applications is straightforward and not much more then call interception is needed. For integration with the complete rendering engine [21] might be more involved, as multipass rendering techniques, e.g. shadow maps [22], or multipass rendering using shaders [21] will not reproduce realistic Holodeck pictures even if all OpenGL calls are instrumented. This is because the results of some passes, such as for shadow map generation [22], are not directly drawn in the scene but are used for intermediate steps in rendering the final scene.

To resolve this problem, it is necessary to modify the rendering engine [21] so that calls that are not intended to affect the screen buffer are not rendered in the Holodeck. This could be achieved by combination of reading Z-buffer at the end of finally rendered scene (as opposed to real-time OpenGL call interception), excluding some rendering passes from Holodeck environment and incorporating additional textures intended only for Holodeck use (so that e.g. bump-mapped or displacement mapped surfaces have "rough" appearance in Holodeck although their Z-buffer values are "flat"). Finally, cyber-physical systems like Holodeck provides one more area where parallelism could be accounted for.

4. HoloSim - Simulating the Holodeck

We have constructed a simulator called the HoloSim as a first step towards realizing our PRE-based Holodeck system. HoloSim is able to show the visual impression of a physical environment rendered by moxels, as well as how animation would look like in that environment. The simulator is currently using pre-calculated moxel positions. We are planning to extend the simulator so that it integrates the complete graphics rendering engine. The HoloSim will be used to test all software control algorithms prior to deployment in a fully realized PRE.

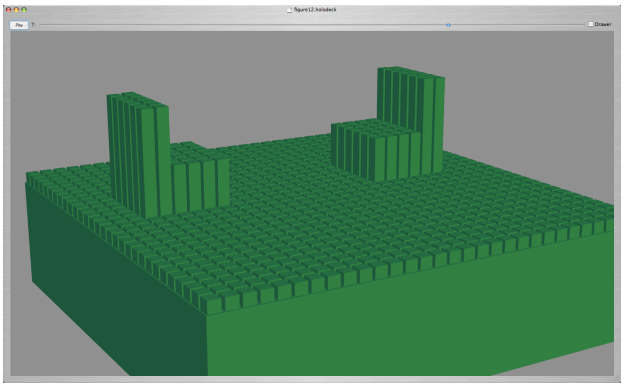

Figure 5. HoloSim rendering two thrones.

Figure 5 illustrates the HoloSim rendering a surface within the PRE using a grid of physical moxel-like pixels. In this case, two thrones are rendered facing each other.

Figure 6 shows a sequence of images depicting how the HoloSim can simulate movement. The two thrones are displaced over time so that they "collide" with each other, but pass through each other without causing damage. This highlights one of the unique properties of the PRE; objects at the same time have both material and non-material properties. Videos of this "collision" are available at http://www.cs.colorado.edu/~rhan/PRE/video/motion.mov.

This example illustrates that HoloSim can be used to investigate a variety of cyber-physical phenomena, including:

1. What are the best software control algorithms for rod movement? Should we choose algorithms that allow rods to reach their final position the fastest? Or algorithms that would minimize discontinuity between rods, to minimize sharp edges?
2. How would the physical system look like without a helmet? Would running the simulator inside a CAVE[16] provide insights on how a completed physical system would look like?
3. The simulator would be the appropriate place to investigate the best ways to integrate with the rendering engine.
4. Approaches towards fail-safeness and physical safety could be tested in a simulator environment.

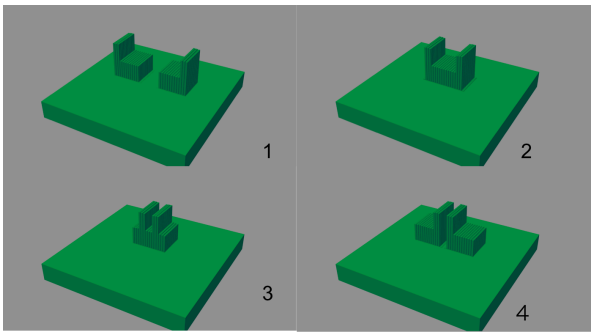

Figure 6. Collision of two thrones in PRE.

Finally, the majority of the simulator code could be used as a basis for the software control module of the physical system once it is realized.

5. Related Work

Intel's Dynamic Physical Rendering (DPR) [3] and programmable matter [4] are visions of using small physical elements called *catoms*, e.g. balls, to create arbitrary 3D shapes. Catoms are able to orient among themselves in arbitrary positions. PRE and DPR are natural complements. DPR is able to create arbitrarily shaped objects, but is likely to have difficulty creating large ground areas on which a user can stand, due to the computational cost and unclear physical strength of rendered catoms. In contrast, PRE's sofware control algorithms are fast and the PRE is capable of supporting the physical weiight of human users. An ideal environment is likely to combine the PRE approach for ground areas, and the DPR approach for smaller and more complex objects in the room.

CirculaFloor [2] is a locomotion interface that uses a set of movable tiles to achieve omni-directional motion while providing the illusion of infinite distance. CirculaFloor simulates only flat surfaces and can't simulate slipperiness, so PRE and CirculaFloor are complements as combining a moxel-based surface with the tiles of CirculaFloor allows for extension of the PRE that is capable of simulating unlimited environments.

Omnidirectional treadmills [5] give a user the impression of walking through an effectively unlimited environment. The Sarcos treadmill [7] combines an angled treadmill, mechanical tether and a CAVE-like [16] immersive graphical interface to provide the impression of a user moving on infinite potentially angled terrains, at the price of limiting freedom of movement (no kneeling or rolling for user). Additionally, it is not possible to simulate fine grained texture like gravel under a user's feet or slipperiness of the surface.

An alternative approach that gives a user the physical illusion of walking arbitrary distances is offered by devices like the GaitMaster [6]. The device has two pedestals or pedals on which a user stands, one for each leg. This type of device confines human locomotion, too.

Augmented reality systems combine physical objects with computer generated imaging [8]. Many concepts from augmented reality such as helmet-based split-view immersive environments [8] are quite complementary with the vision of physically rendered environments. In some cases, the user's entire vision may be immersed by the helmet, while the PRE provides the realistic physical feedback or "feel" of the surfaces.

The designers at amusement parks, e.g. Imagineers at Walt Disney [19], as well as the designers of sophisticated stage sets [18] explore effects that are distantly related to the PRE-based concepts that we are proposing but are typically very specific to one particular environment.

Haptic interfaces [20] allow computer stimulation of the user's sense of touch but are not generally not concerned with locomotion in an environment. Finally, somewhat related work is a Braille enabled terminal [17]. The system that we are proposing could enhance a Braille terminal, as it would not only enable reading of the Braille alphabet, but would also enable perception of new 3D shapes.

6. Future Work

This is still early work that is presenting an ambitious concept. We have raised some of the technical challenges, but they are not meant to be exhaustive. Areas for future work include electromechanical issues, reliability and human perception of motion. Best software algorithms for control and avoiding issues of spatial and temporal "physical" aliasing as well as dealing with physical moxel characteristics (inertia, durability, prevention of collision, user physical safety) are open areas for future research. Many of these issues could be answered by enhancing the current simulator.

In addition to the questions above, once when the physical system is constructed, we would have a novel environment that could be used in a variety of training roles. Understanding human interactions in cyber-physical environments within the context of distributed team training is an area for future PRE research.

To enhance the interactivity of the PRE, our goal is to convey the effect of infinite distance via subtle displacements that re-center a user while they're walking or climbing. This would enable the perception of far greater distance than the actual physical dimensions of the PRE. Our intent is to investigate translation effects in at least two ways, by combining Holodeck based moxel surfaces with the set of tiles used to recenter the users as in CirculaFloor [2] and by combining the Holodeck surface with the treadmill-based approaches.

7. Summary

This paper has presented an ambitious original concept for realizing the vision of a Holodeck through physically rendered environments (PREs). Computer controlled actuation of the height of physical pixel-like moxels arranged in a grid allows rendering of surfaces and terrains. The PRE is designed to support a human user, who will be able to walk over these surfaces, and roll and kneel as well. The coefficient of friction, or slipperiness, of the tip of each moxel would be computer-controlled, to give the user different impressions of slipperiness. We have presented how software control of a PRE-based Holodeck can be achieved by adapting existing graphics pipelines. We have developed a simulation environment called HoloSim that mimics the rendering actions of a true PRE, enabling us to study the most promising approaches prior to building the full physical system. We believe the HoloSim, suitably modified, will grow into the software front end for controlling the hardware back end.

We have presented a pathway towards realizing the Holodeck via a PRE, a powerful cyber-physical system with strong research potential and significant practical applications.

References

[1] V. Krunic, R. Han, "Towards Physically Rendered Environments", Technical Report CU-CS-1033-07, Department of Computer Science, University of Colorado, September 2007. Available at http://www.cs.colorado.edu/department/publications/reports/docs/CU-CS-1033-07.pdf.

[2] H. Iwata, H. Yano, H. Fukushima, H. Noma: "CirculaFloor: A Locomotion Interface using Circulation of Movable Tiles", Proceedings of the 2005 IEEE Conference 2005 on Virtual Reality, 2005, pp. 223 - 230, ISBN: 0-7803-8929-8

[3] Intel Corporation: "Dynamic Physical Rendering", available at http://www.intel.com/research/dpr.htm, visited on September 1st, 2007.

[4] S. C. Goldstein, J. D. Campbell, T. C. Mowry: "Programmable Matter", Invisible Computing, June 2005, pp. 99-101.

[5] R. P. Darken, W. R. Cockayne, D Carmein: "The Omni-Directional Treadmill: A Locomotion Device for Virtual Worlds", ACM Symposium on User Interface Software and Technology, 1997, pp. 213-221

[6] H. Iwata, H. Yano, F. Nakaizumi: "Gait Master: A Versatile Locomotion Interface for Uneven Virtual Terrain", Proceedings of the Virtual Reality 2001 Conference (VR'01), 2001, pp 131, ISBN:0-7695-0948-7

[7] J.M. Hollerbach, Y. Xu, R. Christensen, S.C. Jacobsen: "Design specifications for the second generation Sarcos Treadport locomotion interface", Haptics Symposium, Proc. ASME Dynamic Systems and Control Division, DSC-Vol. 69-2, Orlando, Nov. 5-10, 2000, pp. 1293-1298.

[8] O. Bimber, R. Raskar: "Spatial Augmented Reality: Merging Real and Virtual Worlds", A K Peters, Ltd. (July 31, 2005)

[9] D. Shreiner, M. Woo, J. Neider, T. Davis: "OpenGL(R) Programming Guide: The Official Guide to Learning OpenGL(R), Version 2 (5th Edition)", Addison-Wesley Professional; 5 edition, August 1, 2005, ISBN:978-0321335739

[10] R.J. Rost: "OpenGL(R) Shading Language (2nd Edition)", Addison-Wesley Professional; 2 edition, January 25, 2006, ISBN: 978-0321334893

[11] G. Kiczales, J. Lamping, A. Mendhekar, C. Maeda, C. Lopes, J. Loingtier, J. Irwin, "Aspect-Oriented Programming, Proceedings of the European Conference on Object-Oriented Programming, 1997, vol.1241, pp.220242.

[12] O. Spinczyk, A. Gal, W. Schrder-Preikschat, "AspectC++: An Aspect-Oriented Extension to C++", Proceedings ofthe 40th International Conference on Technology of Object-Oriented Languages and Systems (TOOLS Pacific 2002), Sydney, Australia, 2002

[13] J. M. Rolfe (Editor), K. J. Staples (Editor): "Flight Simulation (Cambridge Aerospace Series)", Cambridge University Press; Reprint edition (May 27, 1988), ISBN: 978-0521357517

[14] Wikipedia entry on Wii, available at http://en.wikipedia.org/wiki/Wii, visited on August 23rd, 2007.

[15] Blender Material Library, available at http://www.blender.org/download/resources/#c2511, visited on August 23rd, 2007.

[16] Wikipedia entry on CAVE, available at http://en.wikipedia.org/wiki/CAVE, visited on August 23rd, 2007.

[17] List of commercially available Braille terminals, available at http://www.tiresias.org/equipment/eb7.htm, visited on February 25th, 2007.

[18] Wikipedia entry on Stagecraft, available at http://en.wikipedia.org/wiki/Stagecraft, visited on February 25th, 2007.

[19] Wikipedia entry on Walt Disney Imagineering, available at http://en.wikipedia.org/wiki/Walt_Disney_Imagineering, Visited on February 25th, 2007.

[20] Thomas H. Massie and J. K. Salisbury: "The PHANTOM Haptic Interface: A Device for Probing Virtual Objects", Proceedings of the ASME Winter Annual Meeting, Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems, Chicago, IL, Nov. 1994.

[21] W. Engel (editor): "ShaderX3: Advanced Rendering with DirectX and OpenGL", Charles River Media; 1 edition (November 2004), ISBN: 978-1584503576, pp. 499-519.

[22] A. Watt, F. Policarpo: "Advanced Game Development with Programmable Graphics Hardware", A K Peters, Ltd. (August 1, 2005), ISBN: 978-1568812403.