THE BIT DOME: CREATING AN IMMERSIVE DIGITAL ENVIRONMENT WITH A KINECT-BASED USER INTERFACE

Zane R. Cochran
Mathematics and Computer Science Department
Berry College
Mount Berry, Georgia 30149
512-576-7243
zane@zanecochran.com

ABSTRACT
This paper presents a unique way to create an inexpensive immersive environment that implements a non-standard interface using the image-based 3D capabilities of the Xbox Kinect. This method results in a compelling interaction that engages students in an immersive experience fostering further exploration into the underlying technology of the installation.

INTRODUCTION

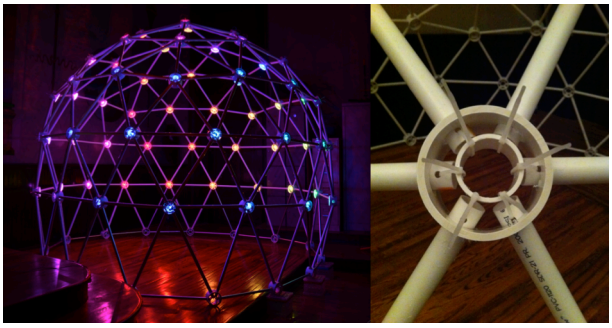
Using state-of-the-art technologies to create interest in computer science and effectively teach its principles is becoming an increasingly important part of education at several levels of learning [1]. Because computer science education can be effectively taught through the principles of exploration and active engagement, the increased use of immersive digital environments has created a demand for inexpensive and manageable systems in which to deliver these experiences [1, 6]. Furthermore, these environments are best leveraged when paired with image-based 3D reconstruction hardware, such as the Xbox Kinect to allow for a user's body to naturally interact with their surroundings.

While often times considered synonymous with a traditional understanding of virtual reality, the primary objective in an immersive digital environment is not to simulate reality, but rather provide an abstraction of a simulated environment in which the user feels that they can affect some change on the digital elements around them. With this in mind, we developed a digital environment, named the Bit Dome, as part of a student-driven physical computing project.

To simplify the construction of the dome structure, the Bit Dome makes use of a number of readily available and inexpensive materials, making it appropriate for an extended student project in a physical computing course. Principally, the system relies upon basic construction materials and minimal hardware including RGB LEDs, an inexpensive microcontroller, an Xbox Kinect and a computer. With these simple items, we developed an installation that is as visually stunning as it is functionally impressive.

THE BIT DOME

The Bit Dome is a semi-spherical geodesic dome with 61 RGB lights evenly distributed throughout the space. At one end of the dome is an Xbox Kinect that allows a user to intuitively interact with the dome through direct interaction with physical elements inside the space and a variety of natural body gestures that simplifies the user interface [10].



a. b.

Figure 1. (a) The open surface of the Bit Dome allows people outside the structure to observe. (b) A close up view of this prototype's free-floating hubs.

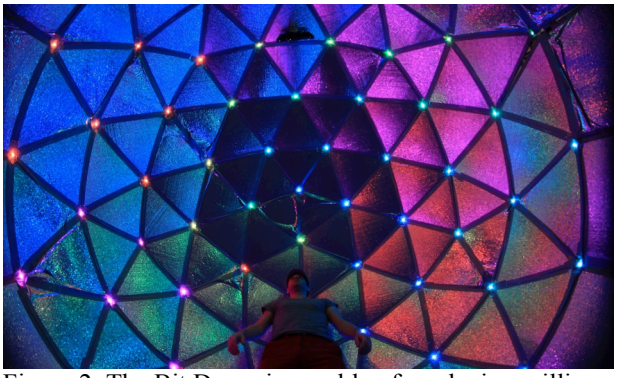


Figure 2. The Bit Dome is capable of producing millions of vibrant visualizations. A user (bottom) interfaces with the dome by standing opposite an Xbox Kinect (top).

Hardware Configuration

A number of hardware considerations were made to ensure that the Bit Dome remained practical and versatile because the Bit Dome was designed to be a low-cost immersive environment developed by students for a physical computing class. The principle design directive for the final structure focused on economy, simple construction and scalability [6].

The Physical Bit Dome Structure

The geometry of a geodesic dome is known for its strength, making the dome easily scalable to large sizes. Its near-spherical shape maximizes capacity while minimizing the amount of material needed for its construction. While not an implicit requirement of immersive digital environments, the dome encapsulates its user, generating the illusion of an isolated experience.

In our implementation, a frame-based construction was used because it resulted in a naturally open display, allowing spectators outside of the environment to observe the natural interaction between the installation and its user (Figure 1a). A benefit of this design is forgiving tolerances in the size of its base components because it is constructed from various lengths of polyvinyl chloride (PVC) pipe that interconnect at 61 hubs. These hubs allow each pipe to float to a natural resting position, regardless of small difference in the length of materials (Figure 1b).

Lighting, Audio and Control Hardware

The primary feedback from the Bit Dome is received through patterns of colorful light emitted from the environment's 61 RGB lights (Figure 2). While 61 pixels may seem prohibitively small, a restricted number of outputs simplifies the hardware and software and makes programming the dome accessible to students. To control each pixel, a Rainbowduino (an Arduino-based microcontroller) was acquired to translate commands received serially from a computer into light arrays in a multitude of colors, thus generating a rich user experience [9].

In addition to visual feedback, there are also speakers positioned around its perimeter that allow for prerecorded messages, music or sound effects to play at set times during the experience. Trial-and-error observation showed that users were able to more easily navigate the dome's various functions when prompted by audible instructions. This also opened up the opportunity to incorporate music and effects in the software to enrich the immersive experience.

Adding Depth Imaging with an Xbox Kinect

In order to interact with the software and ultimately the lights, an input device that was capable of tracking a visitor's location in the dome and body position was critical. The Xbox

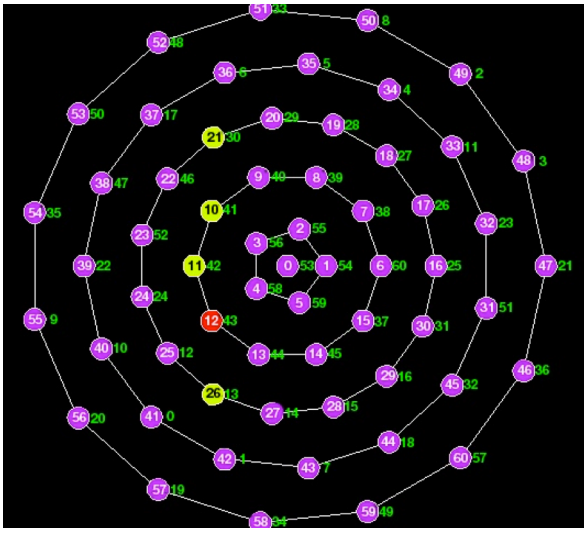
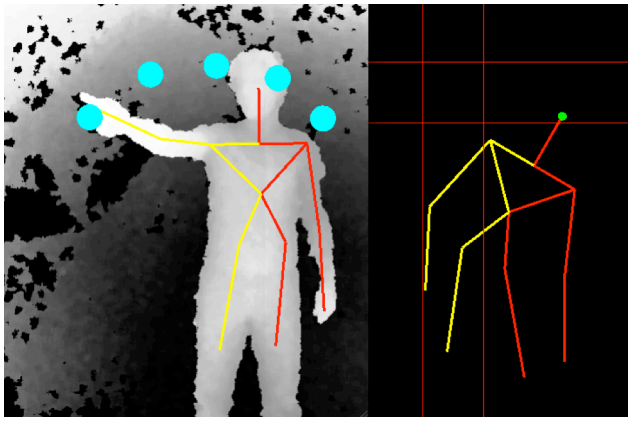


Figure 3. A mapping of virtual lights (numbers in circles) versus physical lights (adjacent numbers). Lights in yellow and red represent interactive menu lights, with red denoting the current selection.



a. b.

Figure 4. (a) Static body tracking allows users to directly interact with fixed physical objects (lights) in the environment from a preset position. (b) Dynamic tracking allows the user to freely roam the space and influence a variety of behaviors in the software.

Kinect was chosen because it is adept at capturing the desired user input and interaction is relatively easy with widely available software libraries [3, 2]. Incorporating it as an input device also encourages greater involvement in the experience and makes interacting with the dome more enriching because users must use their entire body to control the environment [3, 5].

Software Components

There are two main software components that are necessary to make the Bit Dome function seamlessly. The first is a modified version of the firmware that was specifically developed for the Rainbowduino. The second is the main software that runs locally on a computer that interfaces with both the Kinect and the Rainbowduino.

The Rainbowduino firmware was optimized to allow the hardware to be controlled externally over a USB connection from a Java program. This firmware processes 24-bit 8x8 RGB frames at over 30 frames per second [4]. Because the control software that runs on the connected computer was developed in Processing (a language and programming environment based on Java), this was the best choice for controlling the Bit Dome's LEDs.

The main control software was developed in Processing because of the language's emphasis on interactive visualizations and its vast community of support. Students, artists and researchers often use Processing to create a broad variety of image-based interactions [8]. While this installation could be programmed in many languages, Processing enables rapid development of programs because of its simple programming environment and accessible libraries that make interfacing with both the Kinect and Rainbowduino simple.

Low-Level Functions

Because the Rainbowduino firmware is normally used for controlling an 8x8 grid of pixels, it was necessary to consider how lights are ordered in software as a continuous array of 64 lights, versus how they are physically configured as 61 lights arranged in six concentric circles. To accomplish this, a mapping was created that allowed the software to easily reference a particular light given its physical location in the dome (Figure 3). Also, the relationship between lights positioned above, below or adjacent to any given light was also established. The position,

Figure 5. System calibration using a five-point calibration process to orient the user's body in the software program.

neighbors and color attributes of each light were packaged in the software as an individual object that could easily be referenced by a currently running program.

To interface between Processing and the Kinect, we employed the Open Kinect software library. It enables body recognition and the tracking of 12 points of physical articulation that can be easily accessed and used. These reference points allow the software to quickly reconstruct a skeleton of the user's body in a virtual space and empowers student developers the opportunity to quickly develop interactive programs.

The Bit Dome relies upon two different methods of tracking a user's body to determine key body gestures: calibrated and relative. The calibrated method depends upon determining if a user is pointing at a specific light in the dome from a preset location, whereas the relative method determines the body's relation between its various parts (e.g. distance between hands and torso) or location inside the dome [7]. Because the relationship between a user's hand and the physical location of a light in the dome will vary based upon the user's height, arm length and posture, it is necessary to perform a one-time calibration for each unique user.

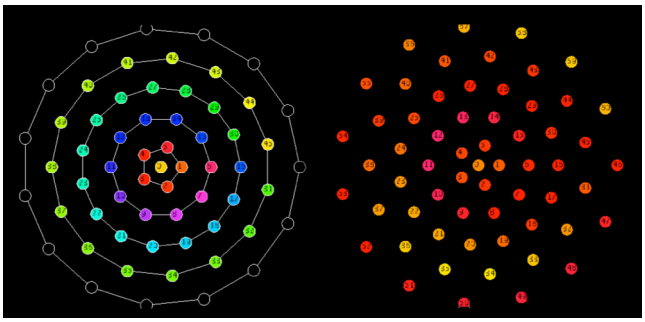
Calibrated tracking exploits the relationship between the location of a user's hand while standing in a predetermined spot in the dome and the placement of several key lights within the environment (Figure 4a). Once this relationship is established, it becomes intuitive to use natural gestures, e.g. pointing, to navigate through the system's many available programs.

Relative body tracking gives the user freedom to move about the dome and affect broad changes in the dome, however, because of the absence of an absolute reference point, users are unable to interact directly with physical objects in the dome. However, this type of tracking allows users a greater variety of interpreted gestures that can affect simulations or even games as they jump, lean, duck, or move their limbs to control an interaction (Figure 4b).

High-Level Functions

The main control software functions in three finite states: dome calibration, menu selection and program execution. By design, when the software begins, it enters a calibration state. After calibration, the software enters the menu selection state where the user points at five designated lights to navigate to one of the programs that can be run in the dome (Figure 3). Upon selection of a program, the dome enters the program execution state that continues to run until the user makes a preset gesture to exit the program and return back to menu selection.

To perform the calibration, the dome audibly instructs the user to stand on a specially marked spot on the floor and sequentially point at the five lights directly in front them. By saving the coordinates of the user's hand during this sequence, it is possible for the program to later



a. b.

Figure 6. (a) A simple scene made using the Dome Painting program. (b) The intensity of lights affected by tracking a user's body throughout the environment.

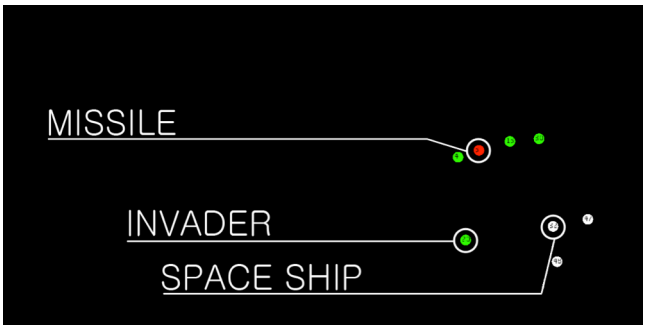


Figure 7. A virtual representation of Space Invaders as it is displayed in LEDs in the Bit Dome.

determine if the user is pointing at a menu option when it is running in its menu selection state. With the calibration finished, the user can naturally point at these same lights to control the functionality of the dome (Figure 5).

After calibration, the Bit Dome's software enters the menu selection state. To simplify user interactions in the environment, menus and executable programs are represented by five lights opposite of where the user stands. These lights represent an unfixed set of commands that can be reassigned as the user traverses a tree of menu options. For example, a user could select a games menu by pointing at the fourth menu light. As soon as the software detects that the user is pointing at a menu item, it audibly announces the light's current function and changes the light's color to signify that it has been engaged. At this point, the software loads a submenu where the same five lights (now different colors to signify the functional reassignment) represent four games that can be played and one option to return to the main menu.

After a program is selected through the menu, control of the Bit Dome's lights is handed over to the program for manipulation. Programs have several ways in which they can change the state of the lights in the dome. Individual lights can be addressed and assigned colors, lights can be turned off completely, and can even be programmed to tween between two given colors over the span of several animation frames. To terminate a program, a user makes the escape gesture and the state of the software is returned to its previous location in the menu hierarchy.

SAMPLE INTERACTIONS

To help illustrate the wide variety of simple and engaging activities that can be created for the Bit Dome, three of the 16 programs developed for it are summarized here. These use simple principles, making creation of programs accessible to beginner and intermediate level students who have a basic understanding of programming languages and data structures.

Dome Painting

After creating a one-to-one correspondence between a user and the dome during the calibration state, it is possible to algorithmically determine the relationship between a user's hand and many lights in the dome. After establishing this relationship, users point at any light in the dome and cause it to light up (Figure 6a). This gives the user the illusion that they have the power to "paint" the dome with light.

Space Invaders

This program revitalizes an old favorite arcade game and reimagines it for use in an immersive environment. The user's ship is represented by three lights along the bottom edge of

the dome. Aliens, represented by green pixels descend down the sides of the dome by randomly selecting neighboring pixels that are below it (Figure 7). The ship maneuvers along the dome's perimeter when the user mimics turning a steering wheel left or right, determined by comparing the relative heights of the left hand to the right. The objective of the game is to prevent the aliens from reaching the bottommost level of the dome by positioning the ship underneath an alien and firing a missile—an activity actuated by pulling both hands close to the body.

Colors in Motion

Colors in Motion is a game that challenges users to freely explore the dome as music is played. The colors of the dome are directly affected by the intensity of movement and the total distance traveled by the user. When the user moves slowly throughout the space the dome shifts its colors toward the blue/violet portion of the spectrum. Rapid or intense movement, however, causes the dome to display vibrant hues of yellow, orange and red (Figure 6b).

CONCLUSION

Throughout this work, we have presented an implementation of an immersive digital environment that demonstrates a variety of skills learned during a typical physical computing course. It has further created opportunities for additional research and development through its open-ended functionality and continues to capture the attention of those who experience it.

ACKNOWLEDGMENTS

Our thanks to Dr. Nadeem Abdul Hamid for his many insights regarding this work.

REFERENCES

[1] Apostolellis, P., Daradoumis, T., Audience interactivity as leverage for effective learning in gaming environments for dome theaters, *Proceedings of the 5th European conference on technology enhanced learning conference on Sustaining TEL*, 451-456, 2010.
[2] Borenstein, G., *Making Things See*, Sebastopol, CA: O'Reilly Media, 2012.
[3] Cervantes, J.C., Vela, F.L.G., Rodriguez, P.P., Natural interaction techniques using Kinect, *Proceedings of the 13th International Conference on Interacción Persona-Ordenador*, Article 14, 2 pages, 2012.
[4] code.google.com/p/rainbowduino-v3-streaming-firmware, retrieved December 10, 2012.
[5] Francese, R., Passero, I., Tortora, G., Wiimote and Kinect: gestural user interfaces add a natural third dimension to HCI, *Proceedings of the International Working Conference on Advanced Visual Interfaces*, 116-123, 2012.
[6] Hole, J., Schull, J., Inexpensive immersive environments, *Proceedings of the 2nd International Conference on Immersive Telecommunications*, Article 16, 5 pages, 2009.
[7] Polacek, O., Klima, M., Sporka, A.J., Zak, P., Hradis, M., Zemcik, P., Prochazka, V., A comparative study on distant free-hand pointing, *Proceedings of the 10th European conference on Interactive TV and video*, 139-142, 2012.
[8] Processing.org, retrieved February 12, 2013.
[9] Seeedstudio.com, retrieved February 12, 2013.
[10] Villaroman, N., Rowe, D., Swan, B., Teaching natural user interaction using OpenNI and the Microsoft Kinect sensor, *Proceedings of the 2011 conference on Information technology education*, 227-232, 2011.