Environments through Motion

Eric Lee

Advisor: Dr. Norman Badler

University of Pennsylvania

ABSTRACT

Technology is taking a path towards more intuitive control schemes for various applications such as playing games and authoring animation. Using gestures or motions simply feels more natural for operating certain tasks in technology. Motion capture is very popular for authoring agent animation in games, but there currently exists no motion system for authoring the virtual environments these agents live in. Motion control for environment design would be incredibly useful for prototyping environments when high level design is much more of a factor than fine-tuning details. Powerful procedural generation techniques exist for rapidly creating large scale environments, but these systems are marred by the need for discrete parameterizations. We want the focus to be on large scale design decisions and the ability to rapidly author and edit to discover what works best in a given virtual space.

We propose a system for the Unity3D game engine that takes advantage of the recently released Leap motion controller. The high precision and portability of the Leap make it an ideal choice of motion tracking technology for authoring environments while sitting at a desk. The focus of this system will be to provide virtual environment designers with motion controls using the Leap to manipulate environments and assets at a macro scale while allowing easy transition of control back to mouse and keyboard when more precision is required. Existing procedural generation techniques, particularly L-Systems, will be integrated as a backbone for some environment manipulation operations. The parameterization will be abstracted by the motion controls, keeping the focus on playing around in the virtual space like a sandbox rather than fiddling with discrete parameters. Ultimately, environment artists will have a system that will allow them to rapidly prototype environments using intuitive motions.

Project Blog: http://environmentsthroughmotion.blogspot.com/

1. INTRODUCTION

Authoring game environments in Unity3D requires a lot of micro-control of individual assets and how they link and interact with each other. Asset manipulation is done with the traditional mouse and keyboard which excels at editing details of a scene. Keyboard and mouse is certainly not a bad control scheme for larger, more experimental edits. However, there is room for an improved control scheme that encourages more experimental prototyping when designing game environments. Using mouse and keyboard, sometimes the macro level design gets lost when the artist focuses on detailing the environment without giving the environment as a whole enough attention.

Having a motion control system would both help optimize the environment authoring process as well as foster creativity in level design decisions when making large-scale edits is relatively simple and intuitive. The method proposed provides more freedom of control and is simple to integrate to an existing Unity setup. It takes advantage of the Leap Motion Controller, a 3” USB motion tracker that can record hand motions with 1/100th of millimeter accuracy. The Leap was initially developed because of the frustration of being limited to a mouse and keyboard for 3D modeling with the idea that virtual modeling should be comparable to modeling clay in the real world with your hands [LEA]. This project proposal then is a natural extension of the intended purpose of the Leap: to provide a more natural way of creating in the virtual space.

Contributions
This project makes the following contributions:

- A more natural control scheme for designing virtual environments suited towards prototyping
- An abstraction of existing procedural generation techniques while maintaining their power in large scale environment generation
- A more efficient environment design pipeline where large scale edits and design decisions can be handled via the Leap with simple transitions back to mouse and keyboard control for finer detail edits.

1.1 Design Goals

This project targets game designers and virtual world researchers. It is in particular meant for those who author in the Unity game engine, but the system can later be abstracted to work with other engines as a further goal. With this new control scheme, developers will be able to prototype their virtual environments in a more intuitive way. An important characteristic of this project is the ease at which designers will be able to switch between the Leap control scheme and the keyboard and mouse control scheme. Fulfilling this characteristic means that designers will not be inhibited by the Leap control scheme as they will be able to use the mouse and keyboard to make finer detail edits when desired. We also ultimately are contributing to the development of motion control as a more natural means of interfacing with certain technology.

1.2 Projects Proposed Features and Functionality

- Motion control scheme using the Leap for navigation and manipulation in the Unity scene editor specifically geared towards prototyping
- Abstracted L-System backbone for procedural generation functionality within the motion control scheme interface

2. RELATED WORK

Both motion controlled applications and environment generation have been explored for many different purposes, but the two have never really been tapped into for use together. Environment generation by motion is relatively unexplored (or at least undocumented) in field of motion controlled applications, but it is worth investigating the research that exists in both domains separated to understand the advantages of combining them together.

2.1 Motion Controlled Applications

When thinking about motion controlled interfaces in modern media, *Minority Report* instantly springs to mind with its applications controlled by Tom Cruise operating a surveillance system using his hands. John Underkoffler was the mind behind this system and developed a prototype himself. His device was capable of navigation through a file system but was limited by its large amount of data capture machinery in the form of gloves and multiple cameras [UND10]. Another popular device for building motion control applications upon is the Microsoft Kinect. The Kinect’s motion tracking technology has been used to great success, such as with the KinectFusion to surface map indoors scenes in real time [NIH*11]. Finally, motion tracking for actual player or other agent animation has been progressing to the point of requiring only a suit with sensors (not a traditional light suit) [EDW13]. This technique is following the direction towards more intuitive authoring methods but is meant for agents rather than environment design.

We see then that two of the primary limitations of previous motion control devices are the lack of portability of the system and need for higher precision (which will always be a factor) [ORL12]. The Leap controller embodies motion capture in a much more compact device and with much higher precision than existing devices [LEA].

2.2 Environment Generation Techniques

Rapid prototyping and generation of environments has been often been done procedurally. One major example is the L-System used for procedurally generating organic assets such as foliage [PRU03]. L-Systems have been expanded on to adapt for different kinds of environments such as sprawling cities [PAMU01]. However, these techniques are inhibited by technical setup or parameterization required. Parameterization still requires human input of discrete values to procedurally generate some mass environment. The advantage then that this Leap system will provide over traditional environment generation techniques is reducing the amount of input required by the author when simply prototyping an environment without the need for precise details. A motion of the hand to generate a quick forest of city will allow the focus to shift away from exact parameterization and back to large scale design decisions. This will therefore speed up the process of playing around in an environment sandbox.

Some basic procedural generation techniques will actually be leveraged as a backbone to add randomization to creating a large amount of certain assets, like trees or buildings. They will be integrated in a way where no parameterization through discrete values will be required of the user. The user’s hand motions will function as the parameters of the L-System to keep the environment design flow rapid and productive. If the user desires to fine tune the L-System inputs, control can always be shifted back to mouse and keyboard.

3. PROJECT PROPOSAL

This project aims to provide a motion based control scheme for designing game environments. The expected goal is that environment designers will be able to prototype a scene in a faster and more intuitive way, promoting experimentation for the level design.

3.1 Anticipated Approach

The approach to this project begins with prototyping a usage example for both the Leap Motion Controller and a Unity plugin. Thorough understanding of both technologies is essential before attempting to integrate the two. The Leap especially is a very recently released piece of technology without an extensively developed library for programmers to take advantage of and will require some exploration. We will first prototype a simple application that outputs the motion data that the Leap records including both built-in and self-defined gestures. We will then develop a prototype Unity plugin that features the user interface to the Leap motion control system. Then we will extend on the Unity plugin to have motion data from the Leap link with basic navigation features in Unity’s scene editor, starting with moving the perspective camera around. After confirming that the interfacing between Unity and the Leap functions as expected, the next task will be to build out controls of navigation and asset manipulation to different motions recorded by the Leap.

The big challenge here will be mapping out these controls in an intuitive and accurate way, since the Leap has three dimensions to work with. The mouse operates on two dimensions and the keyboard is binary, so mappings for Leap will require some thought a lot testing.

The next big step will be to implement procedural generation into the plugin in the form of an L-System. This L-System will be used for prototyping commands for generating large amounts of foliage or buildings (which will be included with the system). The actual parameterization involved with L-Systems will be abstracted away from the user and instead the values will come from the user's motion data. Finally, a method will need to be created to switch control between the Leap and mouse and keyboard. All of this must be tested thoroughly for best user interface. All of the foreseeable code will be written in C++. The demo environments themselves will require some time to author so as to provide a meaningful representation of what options open up to a user using this system.

3.2 Target Platforms

This project will utilize the Leap Motion controller hardware and SDK, the Unity game engine and IDE, and C++ for development.

unity

3.3 Evaluation Criteria

Users will be able to prototype their game environments using motion controls that allow for more intuitive and rapid changes. The ultimate goal is to provide a control scheme that is faster and more natural to use than mouse and keyboard for certain types of editing, particularly large scale edits where the big picture matters more than the finer details. Mouse and keyboard will still be superior for detailed editing. Evaluation can be done based on how users react to using the Leap control scheme versus mouse and keyboard in terms of ease of use, efficiency, intuitiveness, and general enjoyment. If the user feels like they have high level control over their environment as if they were playing around in a sandbox, then we can consider the project a success.

4. RESEARCH TIMELINE

See figure 1 for Gant Chart timeline

Project Milestone Report (Alpha Version)

- Completed all background reading and understand how the pipeline of Leap to Unity plugin will function
- Prototyped the basic Leap data output application, the user interface, and the Unity plugin

Project Milestone Report (Beta Version)

- Building out Unity scene manipulation to Leap motion controls functional with test cases identified and undergoing testing
- Switching between Leap motion control scheme and mouse and keyboard control scheme functional
- Begin work on L-System backbone for asset population

Project Final Deliverables

- A Unity plugin for a motion control system for editing environments using the Leap
- Demo videos on sample usage and documentation videos
- Documentation on how to use the motion control system

Project Future Tasks

- A user study test to evaluate the effectiveness of the motion control system versus mouse and keyboard for different types of environment editing
- Expand on the motion control system to control more aspects of Unity on top of scene editing
- Either abstract or port the system for use in other popular game engines such as Unreal

5. Method

6. RESULTS

7. CONCLUSIONS and FUTURE WORK

APPENDIX

A. Optional Appendix

References

[EDW13] EDWARDS C. Real-Time Scene Design. SIGGRAPH 2013. http://area.autodesk.com/siggraph2012/cyODJsNTqxr_nUQikbgjoIOqpis0NBJr

[LEA] Leap Motion.
https://www.leapmotion.com/company

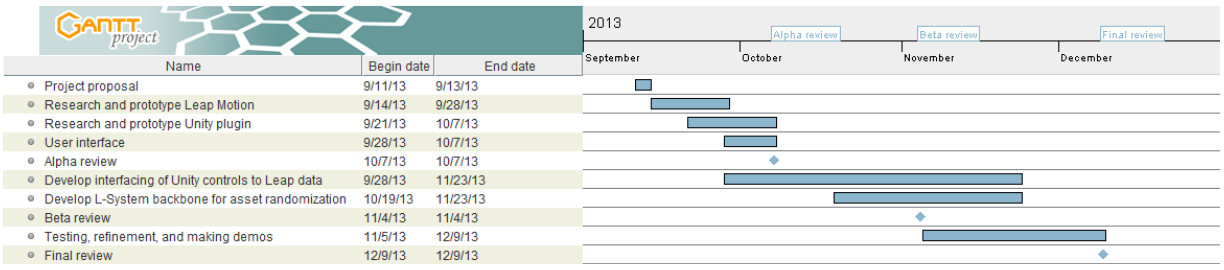
[NIH*11] NEWCOMBE R., IZADI S., HILLIGES O., MOLYNEAUX D., KIM D., DAVISON A., KOHLI P., SHOTTON J., HODGES S., FITZGIBBON A.: Ki-nectFusion: Real-Time Dense Surface Mapping and Tracking. IEEE 2011.

[ORL12] ORLAND K.: Leap Motion Promises Kinect-Beating 3D Tracking. ArsTechnica 2012. http://arstechnica.com/gaming/2012/05/leap-motion-promises-kinect-beating-3d-tracking/

[PAMU01] PARISH Y., MULLER P.: Procedural Modeling of Cities. ETH Zurich 2001.

[PRU03] PRUSINKIEWICZ P.: L-systems and Beyond. SIGGRAPH 2003. http://algorithmicbotany.org/papers/sigcourse.2003.html

[UND10] UNDERKOFFLER J., Pointing to the future of UI. TED TALK 2010. http://www.ted.com/talks/john_underkoffler_drive_3d_data_with_a_gesture.html



Name	Begin date	End date
Project proposal	9/11/13	9/13/13
Research and prototype Leap Motion	9/14/13	9/28/13
Research and prototype Unity plugin	9/21/13	10/7/13
User interface	9/28/13	10/7/13
Alpha review	10/7/13	10/7/13
Develop interfacing of Unity controls to Leap data	9/28/13	11/23/13
Develop L-System backbone for asset randomization	10/19/13	11/23/13
Beta review	11/4/13	11/4/13
Testing, refinement, and making demos	11/5/13	12/9/13
Final review	12/9/13	12/9/13

Figure 1: *Gant Chart timeline*