SCReen Adjusted Panoramic Effect - SCRAPE

Carl Flynn
CASALA Research Centre
Dundalk Institute of Technology
Dundalk, Co. Louth
carl.flynn@casala.ie

David Monaghan
CLARITY: Centre for Sensor Web Technologies
Dublin City University
Dublin 9
david.monaghan@dcu.ie

Noel E. O'Connor
CLARITY: Centre for Sensor Web Technologies
Dublin City University
Dublin 9
noel.oconnor@dcu.ie

SUBMITTED to ACM MULTIMEDIA 2013 OPEN SOURCE SOFTWARE COMPETITION

ABSTRACT

A Cave Automatic Virtual Environment (CAVE) is an enclosed virtual reality room that uses multiple projectors to display images across its surfaces. It uses one or more computers to synchronise and combine the images and allows users to control virtual worlds using a host of interaction devices. Traditionally, a CAVE is used by a single user at any one time and by utilising some form of motion sensing, the user's head position can be tracked to allow for first virtual perception. The images are then displayed in stereographic 3D in order to complete the virtual reality effect.

Professional CAVE installations are expensive and can cost upwards of several hundred thousand euros. This tends to act as a significant barrier to their propagation, however, as the reduction in cost of high specification computers, projectors and graphics cards continues apace, it has sparked a renewed interest in CAVE environments and given rise to the realistic possibility of setting up low cost, amateur CAVEs. Unfortunately, one of the greatest disadvantages of CAVE systems is the lack of inexpensive, easy to use, specialised software. In this paper we present an open source and easy to use CAVE software toolkit called SCReen Adjusted Panoramic Effect or SCRAPE for short. We believe that SCRAPE is the first major piece in a longer-term vision that aims to bring easy to setup, easy to use, portable CAVE systems to all types of non-expert users.

Categories and Subject Descriptors

D.2 [**Software**]: Software Engineering; H.5.1 [**Multimedia Information Systems**]: [Artificial, augmented, and virtual realities]

General Terms

Design, Experimentation, Human Factors, Measurement

MM'13, October 21–25, 2013, Barcelona, Spain.

http://dx.doi.org/10.1145/2502081.2502230.

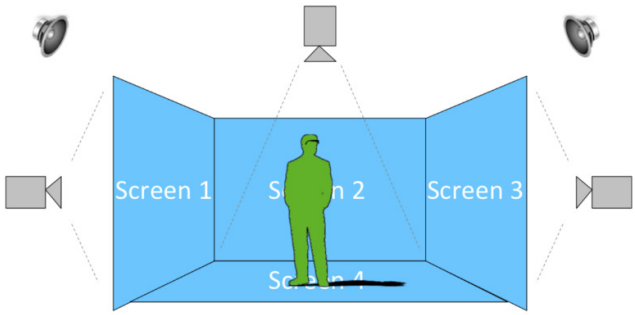

Figure 1: The standard layout for a CAVE

Keywords

CAVE; SCRAPE; Processing; Virtual Reality; 3D

1. INTRODUCTION

The standard setup for a CAVE [7, 6, 8] environment has not changed much since its inception and can be seen in figure 1. The archetypal CAVE is made up of four screens but anywhere between two and six screens is also possible. They typically consist of expensive hardware and software solutions and require extensive investment of time and expertise in order to develop useful interactive virtual environments. After assessing the many different CAVE software platforms available, it became clear that there was an opportunity to develop a simple application which would take the accessibility of one of the most interesting open source visual programming tools available today and adapt it to work in a CAVE.

The primary objective of the work described in this paper was to develop an application that could be easily configured to work in CAVEs of differing sizes, dimensions and screen numbers using a multitude of different inputs. We also put an added emphasis on the need for the software to be open source and freely available. The chosen software development platform is an open sourced tool called PDE (Processing Development Environment) or Processing for short [5]. Processing is released under the GNU GPL (General Public License) and its libraries are released under the GNU LGPL (Lesser General Public License). The technical project to adapt Processing to CAVE environments is called SCReen Adjusted Panoramic Effect or SCRAPE and is intended for

use by anyone who works with a CAVE or similar virtual reality environment (see figure 2). SCRAPE is particularly suited to abstract data visualisations [9] but can be used for visualising a wide range of CAVE-based projects.

In this paper we describe the main motivations behind the development of SCRAPE. We present a technical description of the SCRAPE software platform, outline its configurations, reference key libraries and provide a brief summary of a standard SCRAPE setup.

2. MOTIVATION

The initial seed for SCRAPE was planted in 2010 when the primary author, based between the Centre for Affective Solutions for Ambient Living Awareness (CASALA) [1] and the Centre for Sensor Web Technologies (CLARITY) [2], started working with a CAVE installation. CASALA had recently purchased a CAVE for a six figure sum from well-known CAVE suppliers which provided everything from the physical CAVE structure to the 3D projectors, ceiling mirror, screens, workstations, IR trackers, IR emitters, glasses and interaction controller. They also installed the initial CAVE 3D and tracking software as well as an assortment of additional demo software installations and examples.

Using the supplied software one could take 3D models and quickly view and interact with them in the CAVE. It supported 6 degrees of freedom around 3D objects as well as head tracking to help with the sense of immersion but in reality it was little more than an extravagant stereo 3D model viewer. On discussing these limitations with the supplier, they suggested using an alternative software that would provide us with the levels of interactivity that we were looking for. However, this would have incurred a significant additional cost and the software required considerable expertise in low level 3D programming. Initial trials indicated that a lot of development time would be required to do even relatively simple 3D interactive worlds. Several other similar CAVE software packages were also investigated, however most had similar limitations in terms of accessibility or were simply too costly.

The high cost and inaccessibility of most CAVE software applications left us with a bad impression and initiated our thinking about the idea of creating a low cost CAVE and developing open source software that could cope with the particular requirements of a CAVE. At the same time we wanted to allow ordinary developers to code something that was both interactive and relatively easy to use.

The Processing open source programming language and data visualisation tool platform, created by Casey Reas and Benjamin Fry [5], holds as its tenets the fundamental concepts of being freely available and easy to use by non-experts. While Processing has some limitations, it is an excellent tool for developing 2D and 3D interactive visuals that can help in understanding and interpreting large quantities of data.

Over the nearly four years of using the CAVE we received a steady stream of enquiries from people from all over the world interested in setting up their own CAVE systems. In the majority of cases cost was an important factor and most were surprised when they discovered the price tag associated with both the hardware and the software. There was, however, a definite interest from people in setting up CAVEs for a wide range of different uses, such as data visualisation, product prototyping, training, exhibitions etc. Stemming from our familiarity with CAVE installations, the price tag

Figure 2: SCRAPE being used in a CAVE

associated with them did not seem to accurately reflect the technology and constructions involved. We were convinced that a four screen CAVE could be built for a few thousand euros rather than hundreds of thousands and even less for three screen and V-screen (i.e. two screens) builds. Jeffrey Jacobson, director of PublicVR, developed a set of modifications for the Unreal Tournament (UT) 2004 Game Engine many years ago which enabled the UT engine to display in a CAVE [4] . He also published an online article on how to build a low cost PC based CAVE with UT. Keeping examples such as this in mind, we were confident that it would be possible to build a reasonably comparable CAVE to our professional one for around €3000 but most importantly we wanted to provide a free software tool that even a relatively non-technical person could use.

Included amongst Processing's long list of library contributions lies one particularly interesting library called 'Most Pixels Ever' (MPE) [3]. MPE was created by Daniel Shiffman, an assistant arts professor at NYU and his library enables Processing sketches to run synchronously across multiple machines and across multiple screens in order to give the impression of one large screen. It works particularly well with 2D sketches displayed on a flat surface and was crucial in inspiring the initial idea for the development of SCRAPE.

For SCRAPE we have adapted MPE in conjunction with many other useful libraries in order to visualise seamless 3D sketches on screens placed at entirely different angles and positions to each other. SCRAPE is a setup that is flexible and can accommodate different screen numbers and sizes, different controllers and both stereo and non-stereo images. It was developed specifically to work well on both low budget homemade CAVE installations and six figure research CAVEs. Video demonstrations of SCRAPE in action can be viewed via a dedicated YouTube channel:
`http://www.youtube.com/user/cjflyn`

3. TECHNICAL DESCRIPTION

3.1 Overview of Software Components

SCRAPE works in a similar manner to all Processing sketches in that it uses the two key functions of Processing, setup() and draw(), in order to initialise and animate a 3D sketch. setup() handles all the initialisations for SCRAPE

Figure 3: Modified Processing Icon for the SCRAPE platform

and is only run once while draw() runs continuously and is used to animate the sketch. SCRAPE also contains multiple Processing sub classes which are displayed as tabs within the Processing IDE. Each sub class handles a specific aspect of the SCRAPE application such as the camera movements, the use of different controllers and the use of a stereo and a non-stereo sketch. The SCRAPE logo can be seen in figure 3.

The following is an explanation of the functionality of each class:

SCRAPE.pde This is the core of SCRAPE and defines its main configurations:

1. Imports key libraries
2. Configures main scene variables
3. Reads the mpe.ini configuration file settings and applies relevant functions
4. Sets up the Most Pixels Ever library which controls screen synchronisation and message broadcasting
5. Configures and applies the Obsessive Camera Direction library based on configurations set in the mpe.ini file

joypad.pde
This uses the proCONTROLL library to enable SCRAPE sketches to be easily configured to work with a multitude of different USB controllers. Each action of the controller is then broadcast out to all the other SCRAPE clients which then interpret that action via the 'navigation' code.

keyboard.pde
Some basic code that enables a SCRAPE scene to be controlled out of the box with the arrow keys and spacebar but can be expanded and modified as required. Similar to the 'joypad' code, each action of the keyboard is then broadcast out to all the other SCRAPE clients which then interpret that action via the 'navigation' code.

navigation.pde
This code listens for messages broadcast by other clients and translates those messages into actions within the SCRAPE scene. Actions are applied differently to different SCRAPE clients depending on which screen is set in the mpe.ini file for that particular client. Movements for the camera of a 'FRONT' screen will obviously be different to movements of a camera for a 'FLOOR' screen which point in different directions and sit on different axes.

nunchuckoo.pde
Nunchuckoo is a project in of itself but is incorporated into SCRAPE. The Nintendo Wii Nunchuck has proved to be an excellent controller for SCRAPE and this code allows you to easily connect and configure your Nunchuck and Arduino to SCRAPE. As with the 'joypad' and 'keyboard' code files, each action of the keyboard is then broadcast out to all the other SCRAPE clients which then interpret that action via the navigation code. More information on the Nunchuckoo project can be found at:
`https://github.com/c-flynn/Nunchuckoo`

scene.pde
Simply contains the non-stereographic scene code. This is the code that one would normally expect to find in a Processing draw() function

stereoScene.pde
This contains the same code one would expect to see in the 'scene' tab but also references the stereo library and splits the scene rendering per eye in order to be able to view SCRAPE in full stereo 3D in conjunction with active shutter glasses.

3.2 Configuration

One of the key goals in developing SCRAPE was to provide a system that could adapt dynamically to the requirements of different users with different CAVE specifications. In order to do this, SCRAPE uses a configuration file called mpe.ini. The mpe.ini file within SCRAPE is an extension of the original Most Pixels Ever configuration file and makes use of the previous MPE configurations but also adds a few CAVE specific elements such as screen selections, controller options and stereo activation. Using this file, SCRAPE is able to identify how many screens are being used, the position of each screen, the size of each screen etc. SCRAPE then takes this information and ensures that each camera is positioned according to the correct size, position and aspect ratio of each screen and that the images displayed on the screen appear and move seamlessly. This is a core function of what SCRAPE does and it is primarily handled on initialisation of SCRAPE in the SCRAPE.pde file within the setup() function.

3.3 Other Key Libraries

The key libraries used in order to enable SCRAPE to function correctly are:

Most Pixels Ever (MPE)
Controls the communication between the different sketch instances and synchronisation between screens
`https://github.com/shiffman/Most-Pixels-Ever/`

Obsessive Camera Direction (OCD)
Handles the multiple scene cameras required
`http://gdsstudios.com/processing/libraries/ocd/`

proCONTROLL
Allows users to connect a multitude of USB controllers in order to interact with SCRAPE
`http://creativecomputing.cc/p5libs/procontroll/`

stereo
Enables standard 3D sketches to be viewed in full stereoscopic 3D
`https://github.com/CreativeCodingLab/stereo/`

3.4 SCRAPE Setup

In this section we provide a brief summary of a standard SCRAPE setup. We assume that it is desired to install SCRAPE on a four wall CAVE which has one PC/workstation per screen and one master PC/workstation to control everything. Detailed instructions and open source code can be downloaded from:
`https://github.com/c-flynn/SCRAPE/`

1. Install the appropriate version of Processing onto each computer
2. Download SCRAPE from github onto each computer and copy the 'SCRAPE' folder into the Processing sketchbook folder
3. On the master computer copy the mpeServer.jar file from the SCRAPE folder and place in Processing's /lib directory. This file handles the communications between the different SCRAPE instances
4. Run the mpeServer.jar file on the master computer with the command: java -jar mpeServer.jar -framerate30 -screens5
5. Ensure all the other machines can talk to the master computer with telnet: telnet 192.168.1.x 9002
6. Modify the mpe.ini file for each instance of SCRAPE ensuring that the correct information and options are set for each; such as screen id (starting from 0 up), screen view (Front, left top etc.), screen size, stereo on/off etc.. The controller configurations should only be set and activated on the computer that a controller will be connected to
7. Run the sketch on each computer and once all 5 instances are running your SCRAPE sketch should be visible on all screens

4. CONCLUSIONS

We have presented an easy to use open source CAVE software creation toolkit called SCReen Adjusted Panoramic Effect or SCRAPE. With the ever increasing popularity of CAVE systems we believe that SCRAPE is one of the first major pieces in the overall CAVE jigsaw that will allow non-expert users, with modest budgets, to create and adapt stereoscopic virtual reality environments quickly and easily. The SCRAPE software platform can function in either professional or amateur based CAVE systems and is designed to adapt to multiple screen orientations and sizes using a multitude of inputs. SCRAPE is an ongoing project and continues to be developed as an open source platform for users of all levels to support the development of new and interesting 3D interactive CAVE virtual environments.

5. ACKNOWLEDGEMENTS

CASALA is funded under Enterprise Ireland's Applied Research Enhancement Program with support from EU structural funds. This work is supported by Science Foundation Ireland under grant 07/CE/I114. The research that lead to this paper was supported in part by the European Commission under the Contract FP7-ICT-287723 REVERIE.

6. REFERENCES

[1] Centre for Affective Solutions for Ambient Living Awareness. http://www.casala.ie/.
[2] CLARITY: Centre for Sensor Web Technologies. http://www.clarity-centre.org/.
[3] Daniel Shiffman's Most Pixels Ever. https://github.com/shiffman/Most-Pixels-Ever.
[4] Jeffrey Jacobson's Unreal Tournament. http://planetjeff.net/ut/CaveUT.html.
[5] Processing open source programming language. http://www.processing.org/.
[6] C. Cruz-Neira, D. J. Sandin, and T. A. DeFanti. Surround-screen projection-based virtual reality: the design and implementation of the CAVE. In *Proceedings of the 20th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '93, pages 135–142, New York, NY, USA, 1993. ACM.
[7] C. Cruz-Neira, D. J. Sandin, T. A. DeFanti, R. V. Kenyon, and J. C. Hart. The CAVE: audio visual experience automatic virtual environment. *Commun. ACM*, 35:64–72, June 1992.
[8] A. Febretti, A. Nishimoto, T. Thigpen, J. Talandis, L. Long, J. Pirtle, T. Peterka, A. Verlo, M. Brown, D. Plepys, D. Sandin, L. Renambot, A. Johnson, and J. Leigh. CAVE2: a hybrid reality environment for immersive simulation and information analysis. In *Proceedings of IS&T/SPIE Electronic Imaging, The Engineering Reality of Virtual Reality 2013*, 2013.
[9] C. Flynn, H. Lee, and N. E. O'Connor. Visualising and interacting with a cave using real-world sensor data. In *iHCI 2011 - Irish Human Computer Interaction Conference*, Cork, Ireland, 2011.