Guideline

Document name
Dome and interactive content

Project Acronym:	IMMERSIFY
Grant Agreement number:	762079
Project Title:	Audiovisual Technologies for Next Generation Immersive Media

Revision:	1.0
Authors:	Thomas Clifford, Mariadele Arcuri Rossoni, Christian Södergren, Erik Sundén
Reviewer:	Roland Haring, AEF
Delivery date:	M29
Dissemination level (Public / Confidential)	Public

Abstract

This guideline aims to explain the process behind creating interactive content for a multi-user dome or planetarium environment using real-time game engines.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement 762079.

Table of Contents

1 Introduction

In general, interactive productions for the dome format are few and represent an untapped possibility for presenting high resolution, real-time material. The aim is to make the production of interactive multi-user experiences easier and more common.

In this guideline we focus on the use of Unity as a tool for production. It offers a flexible "off-the-counter" solution, meaning it should be fairly easy to find resources and staff to work with similar production styles.

The end result is an immersive, interactive multi-user experience that allows the audience to be part of the production. The audience may interact with the characters and interaction with the movie will influence the outcome of the story.

1.1 Dome

Size: ca 300sqm projection surface, diameter 15 m, 99 seats

Projection: 6 x DCI 4K Barco 6P RGB Laser projectors, stereoscopic 3D

Net resolution depends on actual system setup and calibration due to edge blending and geometric calibration. Resolution without edge blending is 8192x8192.

Audio: 7.1 Surround-Sound

2 Production

2.1 Real-Time

In the production of an interactive dome experience, there are several things to keep in mind.

When aiming for real-time graphics, minimizing loading times of large environments is essential. This can be achieved by creating small submodules or subscenes where parts of the landscape are grouped together under one parent. These modules can then be activated and deactivated through their parents at the beginning of each scene. This can significantly decrease loading times and most of all guarantee an acceptable frame rate.

An economical approach to applying 3D models can also be of use. By using only a set amount of models and textures for landscapes and environments, gains can be made through repetition rather than using unique models.

The aim should be to create a movie that runs smoothly in dome facilities. If a completely real-time software is to be achieved, the FPS should not drop below 30 FPS and preferably stays stable at around 60 FPS.

2.2 Interactive

Deciding how the audience should interact with the medium is key to the process. Depending on the available technology, mentometers (audience response systems) and other sensors can be used to heighten the experience. On the other side of the spectrum, a human-interaction can be just as useful when not to make the audience lose their attention to the screen when dealing with mobile phone apps or other technical solutions or when the audience consists of children, who may have difficulties with some technical solutions. Utilizing staff in the role of "game-master", they can guide the audience through the experience and manoeuvre the game according to the audience's response. It also opens up for more analogue inputs such as using physical signs for the audience to hold up for "yes" and "no" choices.

Response from the audience can range from actual dialogue between the guide and the audience (in smaller groups and smaller theaters), the use of aforementioned physical signs that can be held up by the audience, or digital input through mobiles and mentometers or buttons. Some interaction styles fit most dome setups and can be used as a general guide for getting started:

Voting live - The audience chooses between different story paths by utilizing physical signs (printed on paper or cardboard for example).

Action-like interaction - The audience influences the performance of the main character(s)– how fast they run, how fast they swim, steering their direction, timing of jumps and other actions. This works best for action filled scenes and works best when graphics are being rendered in real-time.

Sound-activated interaction - by clapping hands or making noise, the audience can affect certain outcomes.

By utilizing recognizable everyday actions such as clapping hands, your audience should be able to jump right into the game.

2.3 Multi-computer / multi-display support

When possible it is great from a software flexibility stand-point if you are able to run the dome application from one single machine, as the software itself then does not need support for running on multiple computers (cluster) simultaneously. But often, to utilize the full capabilities of the display system one single system does not have the capacity for running the desired application with enough performance. Such support is less common within game engines, but even more established ones (such as Unreal Engine through nDisplay) have recently started to implement their own support within the engine for running an application on a multi-display and multi-computer setup.

2.4 Single machine setups

But from a flexibility stand-point, solutions which have limitations in regard to resolution and our framerate are still considered within domes and immersive environments, such that content creators can utilize more preferred tools available. That's why it is technically possible to date to capture at least a 4Kx4K fisheye directly from a single PC and utilize another system for displaying that on the dome. This makes it possible for creators to produce real-time content applications which have no cluster-support but still have camera-setups such as fisheye lenses available. In fact, the extra effort between a dome production compared to a normal flat screen production is then quite small. The popular Unity game engine platform does not support any cluster functionality in general, but there is actually free and competent work made available for fisheye cameras that can be just drag-and-dropped into a scene without much effort[1].

2.5 Immersion

Going from light to dark environments is a quick and effective way to add depth to an immersive dome production. Real time lighting and shadows are nowadays very easily implemented in any game engine.

In a dome, making the whole projection screen black except for a well lit area is a great way to create an immersive feeling. It is usually very well received by the audience too. It is also a common video game trope that many will recognize, and the resulting spelunking will add an extra exploration layer to the story.

As in a good magic trick, content creators should keep in mind where the main focus of the audience is. The so-called sweet spot of the dome is simply the centre of the dome where the audience lock their gaze when they are sitting with their head in a relaxed position.

Move the protagonist from the sweet spot to a more remote part of the screen to lead the audience to look elsewhere than this spot. Once you have shown that things are going to happen anywhere on the screen, you will need much less effort to attract their interest to the full range of the dome.

Another simple method is to place colorful moving objects in the perimeter, such as vibrant flowers and foliage, interesting details or moving animals and fish. Likewise, you might want to have the protagonists physically point at an area of the screen that is off centre and mention a detail, and the audience will start focusing their gaze there. This is very useful if you want to create a surprise event where the audience suddenly notices a clearly visible fiend or guest character that has been lurking in a less focused on area of the screen. The resulting effect can be either comical or frightening, keeping the audience on their toes without the need for a constant cavalcade of unlikely but spectacular events.

Thus cutting down on production time and giving a more balanced and engaging rhythm to the story – the audience starts to feel more immersed in the story and feel like a part of it, as opposed to being a mere spectator.

[1] https://assetstore.unity.com/packages/vfx/shaders/fullscreen-camera-effects/dome-tools-62664

3 Practices

3.1 Camera setup

Always make in- game camera dummies and have the dummies run the show, with well separated scripts to aim the camera and move it around. You will want to attach several cameras to the same dummy, but only have one activated at a time.

The different cameras could be a preview cam, a dome projection cam, a regular 16:19 cam with deactivated scripts for frame grabbing. This solution works out very well and ends up being a flexible solution through and through.

You need a dummy to adjust these two cams singularly as they are not completely interchangeable at the same angle and distance. Notice also how the toucan's distance from the camera is affected. You need to take that into account when framing your two different cameras, and that's where having a dummy really comes in handy.

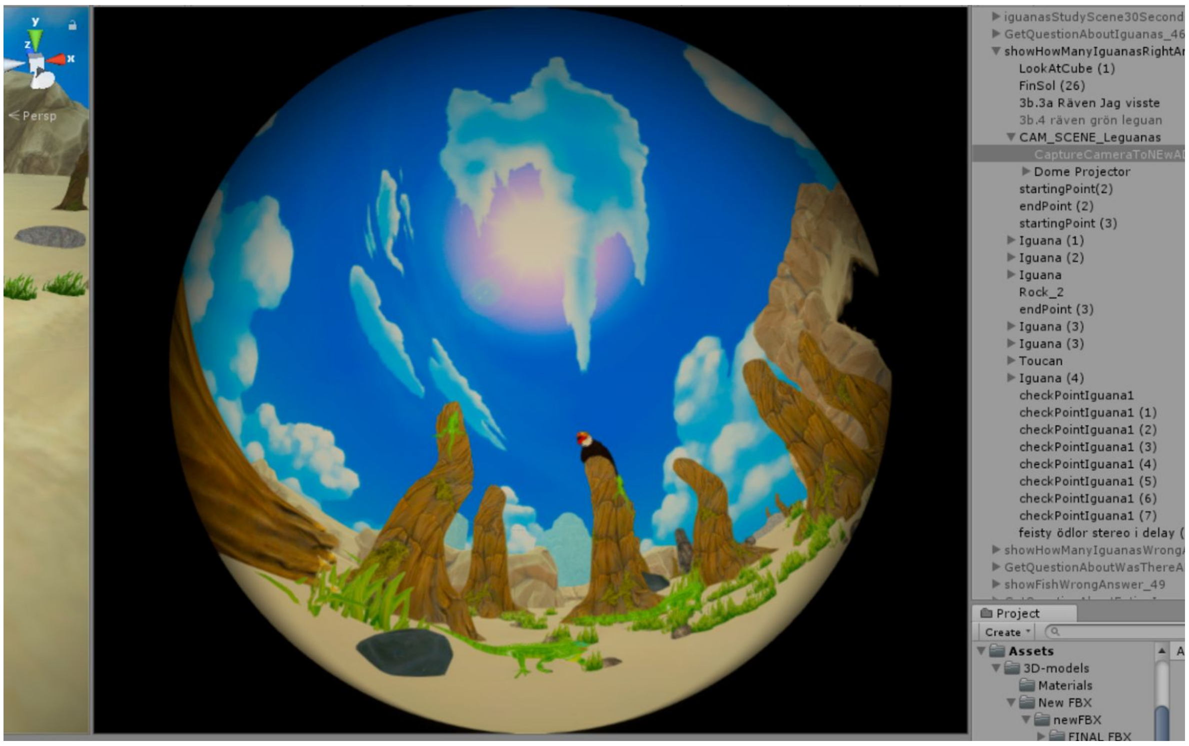

Figure 1. Dome projection camera.

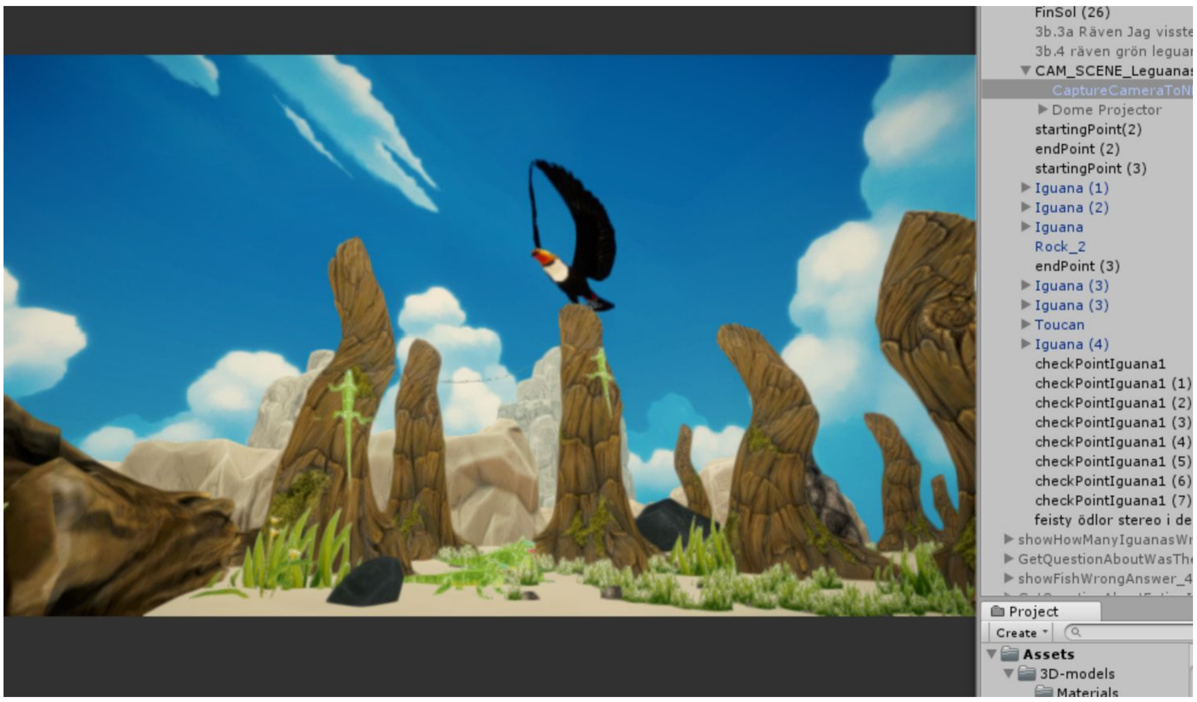

Figure 2. 16:9 camera.

3.2 Using camera paths imported from 3D programs

For more dramatic scenes, you may want to try a different method. Some scenes may be custom made in Maya and may involve a group of characters interacting with each other to a set dialogue. Here you can animate the camera in Maya and then import the animated camera with the rest of the fbx. You can then parent your own fish-eye camera to the camera object. It will inherit the custom made path for that scene. This in the case where character animations are strictly tied to camera work and vice versa.

3.3 Code your software for content creators

Important for development work is to have a good understanding of what type of tweakables that are to be delivered. A developer’s goal should be to create versatile tools for the team.

Understanding deliverables in areas where a developer might not be so well versed is especially important, such as camera paths and clipping, fading, as all these things are to be custom scripted by the developer. The tools and pipeline created at the beginning of the process can do a great deal to cut down your production time and increase the quality of the end result.

Having thorough dialogue early in the process and continuously throughout the project with content creating teammates is pivotal to avoid re-writing of code and upping the quality of the end result. Going through possible features together with the team before starting to implement them seriously, should be common practice in interactive productions. Bottom line – develop with one goal in mind: It must be easy for content creators to test new solutions with as little effort on their part as possible.