This research is a condensed abstract from a larger piece of writing, please contact michael@michaelmckellar.co.uk for a link to the complete document for proper referencing.

Performance Enhancement in Multi-Output Playback Systems

Michael A McKellar

michael@michaelmckellar.co.uk

Contents

1 Technical specific experience

Based on the experience and data gathered in observation, development and interviews regarding the immersive dome space, the influence of technical factors on the user experience became a key focus. PANS outlined the constructs of user experience within an immersive dome but is not currently able to gauge what effect changes to design or implementation will have on users.
This spurred an iterative design, development and implementation cycle for a number of key technical enhancements that focused on analysing and improving the base capability of the immersive dome space to reduce possible reductions in experience. As our research showed breaks in presence, such as those identified in studies by Usoh *et al.*, 1999; Slater and Steed, 2000; Chung and Gardner, 2012, all weighed more on user opinion on experience than having seamless execution of any of the constructs alone.

Schnall, Hedge and Weaver, including above, regard resolution, framerate and FOV (field of view) (2012) as the main technical elements to consider within an immersive fulldome space. They and other authors (Lantz, 2006 for examples) agree that technical limitations and poor technical implementation within fulldome environments are a key element that drives down user experience compared to other medias. Our own interviewees were also drawn to the ideas of technical limitations or breaks in presence within their experience of the Portal as detractors to the environment, Table 1 shows various examples from across the participants.

Quote	Participant
I guess anytime that it technically goes wrong it should be seamless, the experience should be instant	7
That [poor visual quality due to resolution] ruins it because as you can see this stunning scene on an oculus, and even better on a PC or a TV screen and then you go to the dome where you think that should be better.	9
the lighting has a huge effect. [At a conference] the big problem was we didn't account for the lighting, naturally lighting [the projection was lost]	4

Table 1: Examples of interviewee referencing technical limitations

This directed focus on refining the methodological concepts. Using the academically agreed technical elements of a fulldome experience we were able to define several experiments, refinements and implementations to test and improve the Portal. As the physical design was fixed in both manufacturing and preferred for the portability/mobility provided we regard FOV (field of view) as a constant throughout. Research from Deering (n.d) into the limits of human vision allow us to regard the 180°hx155°v FOV of the Portal as able to fully encompass a viewer's vision on a forward-facing surround. Exploring expanses of greater, completely spherical screens is outside the scope of this research.

However, the IDE inherited at the start of this research was both sluggish in maximum FPS and poor in terms of deliverable quality. The catering of viewable content was a very manual process and overall control over the experience was out of the scope for any 'non-expert' users. Limiting the overall experience of a user within the IDE and in theory lowering nearly all aspects of the PANS framework due to the required attendance and catering of all experiences by a dome technical.

Therefore, this research had very clear scope to focus on three main areas of the underlying technology and design of the Soluis IDE:

Technical Infrastructure – Improving and refining the existing system architecture and incorporating modern computing pipelines

Playback optimisation – Refining system wide understanding of ideal content settings and styles for improved user experience

Content Delivery - Understanding the user experience of getting, playing and uploading content to the IDE. The underlying management systems and interfacing methods

The remainder of this chapter will aim to explore and refine the Soluis IDE to improve the overall designable and technical experience offered within the space. The overall goal to improve the top down potential of the PANS framework. The better the theoretical potential experience, the better the overall experience.

During the developments within the Soluis immersive Portal, each area was tackled differently depending on the availability of other technologies or common research in the area. Some via technical development, hardware changes, software refinements as well as planned long-term development goals with solutions to improve the identified experience impactors.

Research was conducted via a natural A/B test system on the prior and post systems. Understanding whether the changes were having the desired effect on the users perceived experience and the initial findings and effects are discussed in the relevant headings.

1.1 Existing infrastructure

This research joined the Soluis dome environment in its second year of iterative development. While the physical structure was approaching the now standard aluminium geodesic frame Figure 1 there had been little or no direct technical input into designing a system to correctly operate it.

Figure 1: Naked geodesic frame

The new geodesic frame added complexity of higher potential resolution, via increase screen surface space and more direct projector inputs. Designed to have four ultra-short throw projectors with an initial resolution of WQXGA (1920px x 1200px) across the 2m radius the Portal was

theoretically able to produce 3000px x 3000px fulldome content at over 60 frames per second (fps) (see for more details on fulldome content: 'Emergence of Fulldome', 2005).

However, due to a multifaceted dual server system the Portal was originally only capable of 2500px x 2500px content at variable framerate, producing a maximum of 25 fps.

Figure 2 shows a high-level breakdown of the dual server configuration. One system, a high-end gaming machine, that was responsible for rendering and producing the visual content via games engine. The second, a higher end professional rendering machine that dealt with the complex geometric calibration to map fulldome content onto the curve projection surface.

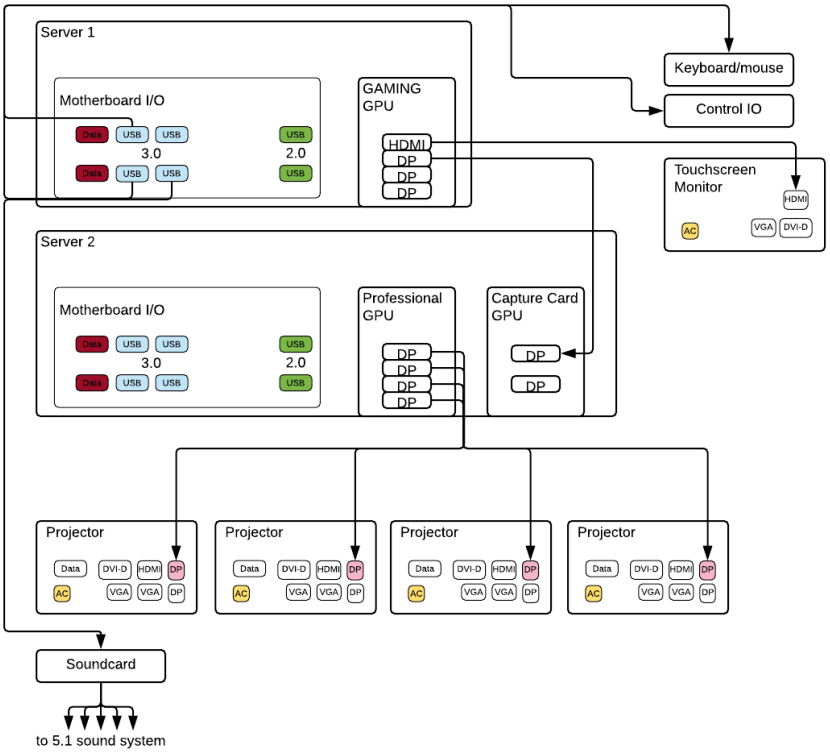

Figure 2: Overview of original Soluis Portal server configuration

Due to Soluis's use of unique high-quality games engine content within the Portal, these steps were seemingly necessary.

Internally, each server system used a variety of computational and graphical manipulation to achieve output onto the dome surface. The visual development environment 'Touchdesigner'

("Derivative TouchDesigner," 2018) was a host platform (inside the Windows OS) throughout both servers due to its widely adaptive capabilities in visual processing. Figure 3 is an overview of all computations required in the presentation of content, Touchdesigner being responsible for 2/3^{rds} of each specific content pipeline shown. Its breadth in ability allowed it to handle the both the conversion of 2d square images into flattened 3d imagery, as well as the complex mathematics of projecting the images onto a curved surface in real space. While outside the scope of this research, it is useful to know that the dome projection mapping occurs when processing an image through a specially made 3d mesh (or geometry) object per projector to correctly correlate each pixel into real space. This can be imagined as a digital 3d model of the dome with 4 rectangular cloths (the images) pressed onto the surface (See Bourke, 2016).

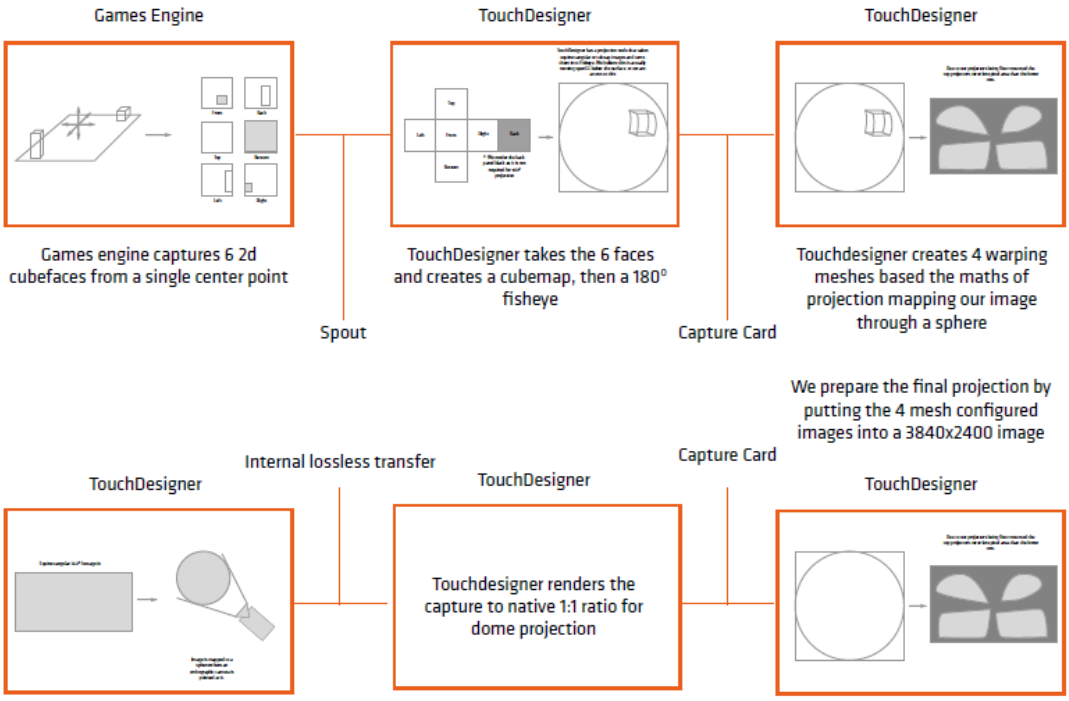

Figure 3: Content creation pipeline for dual server system

The final piece of understanding required of the original Soluis Portal is the technical process of getting a complete image frame from creation to projection. This is where nearly all of the bottlenecks in offered experience occurred. Figure 4 labels the perceived bottlenecks within the original systems. Specifically; those in the outdated content capture system within the games

engines; the creation of the fulldome spherical image; the bridge between servers; the projection mapping of each pixel onto the dome surface and the splitting of the image to each projector.

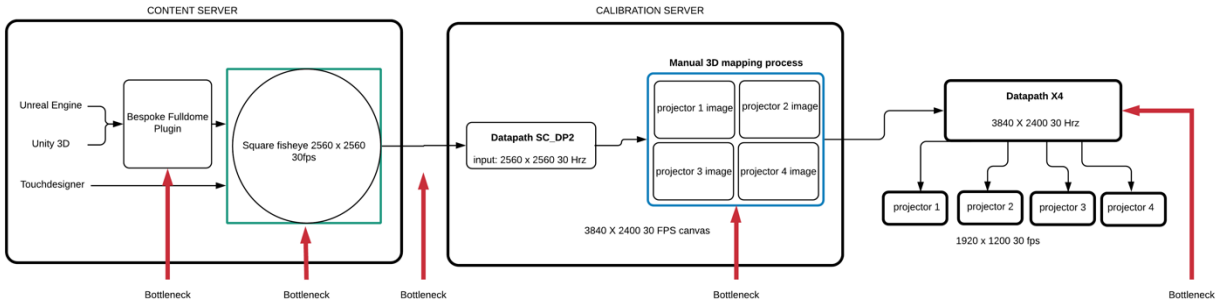

Figure 4: Technical content pipelines for dual server system

While this section directly references most technical limitations, items such as content control, selection, overall user experience and choice within the Soluis Dome did not yet exist. Controls were rudimental implemented via the use of a games console controller and large touchscreen for user input (Figure 5). All content was loaded and accessible in single chunks and if the user wished to swap content, a restart of the software was required (this removed the projected image while loading).

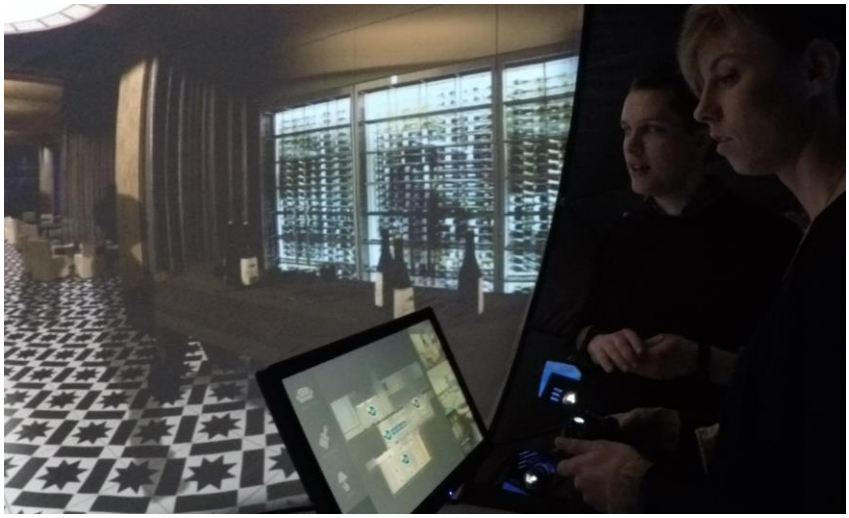

Figure 5: Initial control mechanisms

The following sections will address the existing infrastructure, their refinements, tested and perceived impact on the user experience, PANS rating and overall improvements to IDE design in the future.

1.2 Technical streamline

Technically improving the Soluis dome was one of the most directly measurable areas of development, due to nearly all changes being non-user facing. As Figure 2 and Figure 4 explore above there was a large amount of over complication and redundant hardware within the Soluis pipeline. Compared to the causes of lowered engagement (resolution and framerate) and the subsequent breaks in presence and reduced PANS rating, being able to improve the overall technical quality of various steps by a measurable amount would in theory improve a user's end experience. It is presumed that all technical streamlining will allow for higher framerate playback and increase the overall pixels that can be created per frame.

For the scope of this research there were three main areas of technical streamlining:

- Reducing the amount of complexity and steps required for content projection
- Improving base graphics processing technology
- Altering the geometry warping calculation methods

1.2.1 Single Server Optimisation

Initially, reducing the amount of conversions, steps or transfers from load/creation through to projection will reduce a large amount of system latency and more importantly free up system resources. As discussed by Deering (no date) 60hz (or 60 frames per second (fps)) is a presumed standard for refresh rate where the human eye doesn't perceive a loss in experience due to the limit. Due to the complex conversions and capturing of data numerous times between the multiple servers it would take up to 40 milliseconds to render a single frame. To achieve 60fps that number would have to be less than 17 milliseconds. Setting the goal framerate to a maximum of 60 fps was also a logical set from a theoretically possible perspective. Figure 6 shows the exponential curve of render time per frame vs goal fps. For example, aiming for 70 fps would require another 20% reduction in render time over that required for 60 fps.

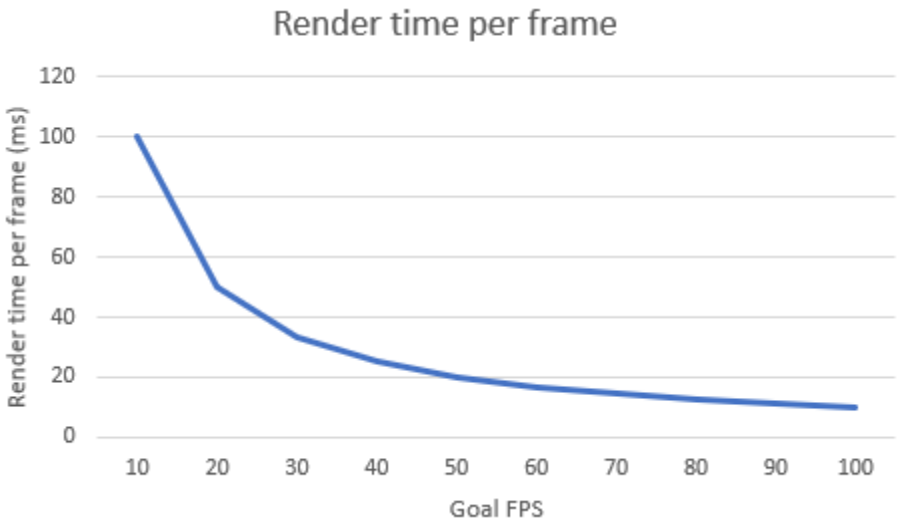

Figure 6: Render time per frame graph (ms)

While the exact times are specific to the task being carried out (running real-time games engine content takes much more resource than video playback), Table 2 shows an example frame render timeline for a real-time scene within the original setup. This data was collated over several random sample tests within the dome.

Render Task	Time (ms)
Render games engine content (6 cube faces)	10-15
Generate 180° fulldome content	1-3
Capture data between servers	10-15
Geometric calculation	2-3
Warp image through geometry (4 projectors)	5-7
Manual image blending (4 projectors)	5-7
Rending windows	<1
	~34-51 ms

Table 2: Dual server example render timeline

While none of these operations are independently too large, together they far exceed the 16.666ms limit for 60fps playback. Removing the requirement for capturing data between the servers only could theoretically improve performance by 60% ("Video Capture Cards | Datapath Video Capture Cards," 2018). Using the release of latest generation graphics and computing processing power

(“NVIDIA Announces Quadro Pascal Family: Quadro P6000 & P5000,” 2016) a new single server solution was devised to attempt to elevate the issues outlined above. This would completely remove the dependency on the capture card connection (“Video Capture Cards | Datapath Video Capture Cards,” 2018) and allow a single Touchdesigner instance to self-regulate threads of spout (“Spout,” 2018). Spout is a modern graphics texture sharing method to allow multiple applications, process access and altering power over a single texture location within memory. This change allowed the single system to continually alter a single image, rather than create and alter a copy, with the previous stage being discarded until the rendering stage where it is flushed from GPU memory. Spout operates by making part of the GPU’s shared memory available in other applications. Allowing for direct sharing of what one application is seeing into another. While there are no direct studies on the impact of spout instances it is presumed as the most efficient method of sharing images on a GPU.

This also alleviates an arising in variable refresh of content in ‘real-time’ rendering setting, where Touchdesigner is ready to pull its data but it has not been passed by other software systems. meaning that content could not be guaranteed ready by either TD instance – causing jumps, jitters and tears.

Touchdesigner (TD) operates in a ‘cook’ per frame method, each time a frame is to be generated the TD network will calculate all its nodes in a pull (it goes from output up chain until termination). While this instanced method of TD operation is the normal for performance work, it comes into an issue where each instance requires content from the other (the GPU) and spout to complete all their relevant tasks (or cooks).

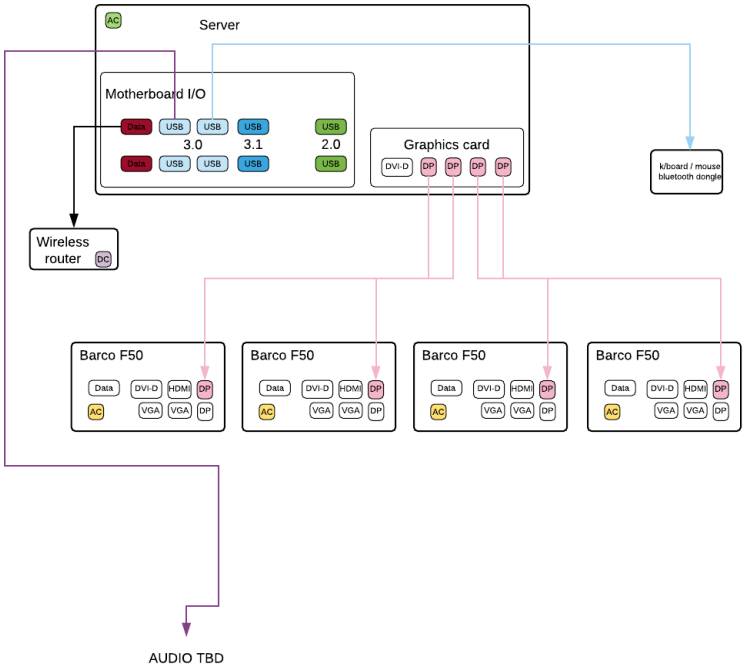

Figure 7: Single Server overview

Figure 7 and Figure 8 show the technical overview diagrams for both server architecture and technical content pipelines; both of which are visibly less complex than their predecessors in terms of parts and conversion. While this research will not describe the individual components of the server to conserve intellectual property exploring how the implementation of a single server was possible via new graphics technology is vital to understanding the overall improvements to the Soluis Portal and the potential effects on user experience.

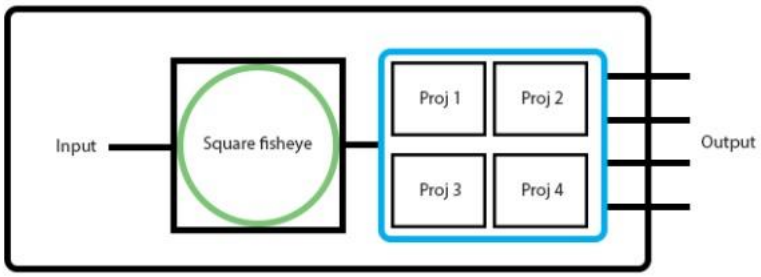

Figure 8: Single server content pipeline

As outlined above, one of the main optimisations moving to a single server allowed was the introduction of advanced texture sharing within the graphics card. While the improvements and processes of openGL texture sharing are well documented ("OpenGL Multi-Context 'Fact' Sheet | Perfect Internal Disorder," 2018; Wynn, 2018) in the case of the Soluis dome it meant that the theoretical rendering speed of the system was now dictated by the GPU processing power.

1.2.2 GPU Overhaul

In the period between 2015-2017 a new generation of graphic processing chip on market allowed for a near 60% improvement in conventional graphics performance (Smith, 2016). An obvious addition for the Soluis dome, replacing the previously top of the line professional GPU with the current top of the line GPU rendered a pipeline improvement of around 40%. Table 3 shows an overview of the render pipeline and timings for the single server and new (pascal) gpu setup.

Render Task	Time (ms)
Render games engine content (6 cube faces)	8-12
Generate 180° fulldome content	1-3
Capture data between servers	N/A
Geometric calculation	<1
Warp image through geometry (4 projectors)	2-4
Manual image blending (4 projectors)	1-3
Rending windows	<1
	~14-24 ms

Table 3: Single Server, Pascal GPU

1.2.3 Automatic Projector Warping and Blending

The final technical optimisation focused on by this research was the changing of the processing of the 3D geometry warping, or 'mapping', required to correctly project content within a domed surface. The Soluis dome used an outdated manual 3d warping tool that altered 3d-meshes based on pixel position. Primarily this was controlled by conventional computer mouse, caused a 2d axis in a 3d space context issue. Human error and inaccuracy causing numerous discrepancies in mapping consistency. The conventional method of multiple projector mapping blending is to fade in equal amounts and opacity. Figure 9 shows an overview of what the edges of each projected

image should be. Standard procedure is to use an overlap of between 20%-30% of the overall image to blend seamlessly. Dome projection mapping suffers from multi-projector, multi-angle blending requirements that without impractical amounts of resource and time are impossible for a human to map and blend efficiently. Figure 10 shows an actual capture of the previously standard human mapping and blending between the four projectors within the Soluis IDE. The areas of miss-matching blending where colour and opacity amounts are not equal. This directly translates to poorer experience within the Soluis dome, a very obvious technical error that stands between the users and the content in which they are to immerse. While un-measurable to a degree map-able by current PANS, it was presumed that any improvement to the quality of the projected image and the blend between images would increase the ratings received in follow up evaluations.

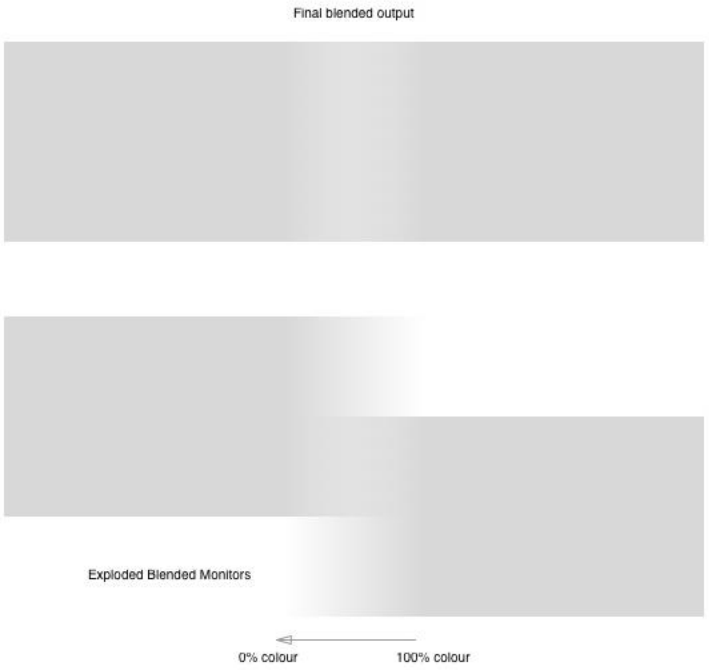

Figure 9: Conventional projector blending overview

Figure 10: Capture of four projector overlap with blending in Soluis dome

As well as the image quality possible via the existing mapping method, it was a multi-stage process per frame that had the most room for improvement without a hardware change. Table 3 references the whole process as between 4-8ms per frame (up to half the maximum time per frame for 60hz rendering). The main reason for this was the fact that the geometric calculations for the dome surface was not baked into the pipeline. The software option within Touchdesigner that controlled the 3d geometry was active every frame, essentially calculating complete 3d warping every time.Touchdesigner supports the embedded of specially formatted configurations files, all but removing the processing time.

However, modern automatic projection tools are well documented (see Lee *et al.*, 2004; Fiala, 2005; Audet and Okutomi, 2009) using a variety of computer vision tools to track projected patterns and calculate screen space.

Using specialist fisheye camera hardware, and bespoke automatic calibration software from Vioso ("VIOSO software for auto alignment multi projection, edgeblending, media server, warping and softegde," 2018) we were able to implement near perfect calibration of the Soluis IDE in under 15 minutes. Vioso is one of the most advances automatic software's on the market,

able to detect difference in surface texture and tint it attempts to correct for all within its two-step calibration of colour and geometry blending. Using a single 220º fisheye camera allowed for us to map the entire 155ºx180º surface of the Portal with a single camera, in roughly 4 minutes per projector. Figure 11 shows a side by side comparison of both the previous manual calibration best efforts against the new automatic calibration blending. Able to calculate in both 3d space and four directional overlapping it is capable to create a near seamless screen to the user.

Figure 11: side-by-side comparison of manual (left) and automatic (right) mapping & blending

Vioso generates a file that is readable by Touchdesigner in such a way as to remove almost all calculation and processing time ("Vioso - TouchDesigner 088 Wiki," 2018) while preserving the perfect blending created in the software. Implementing this step took the Soluis dome to a stage where it was finally below the vital 17ms number for frame time.

Table 4 shows the final render task timings based on analysis of the now running Soluis IDE including the single server optimisation, graphical system overhaul and the implementation of embedded automatic mapping calculations.

Render Task	Time (ms)
Render games engine content (6 cube faces)	8-12

Generate 180° fulldome content	1-3
Capture data between servers	N/A
Geometric calculation	N/A
Warp image through geometry (4 projectors)	N/A
Manual image blending (4 projectors)	N/A
Automatic Geometry and blending pass-through	1-3
Rending windows	<1
	~11-19ms

Table 4: Render overview after all optimisations

Perhaps more so important than the physical timing changes as a result of the technical streamlining was the non-direct improvements to overall potential experience. All of the improvements documented in this chapter serve to reduce the potential impactors on the PANS framework. By reducing such things as 'breaks in presence' in an environment that no longer lags, skips or stutters content playback; creating a more seamless window to the content that is attempting to immerse the user or increasing overall content playback speed to improve motion and the realistic feel of a scene. The following chapters will focus on more user direct impactors such as content access and playback and provide initial empirical study into the results based on A/B testing due to the user aspect of measure experience.

1.3 Content Playback Optimisation

The technical experience within an immersive dome environment is a distinctly non-user facing tool that powers nearly all other aspects of a user's experience. It can be presumed that by improving the technical experience, presence, and to some degree narrative's effect on a user, can be improved. This means that a technically superior computer system alone will not improve user experience, and this is the case for the Soluis IDE.

As stated in 1.1 there was little to no creative or designed aspects of the Portal beyond those required to operate and play content. This chapter will focus specifically on this researches changes, implementations and optimisations to allow for a more engaging user experience. Helping to further Agency, Narrative and improve the Social offer within the Portal.

In the bridge between technical and user focused development there were a number of improvements to the way that content was physically created with the goal of improving render times, the only figure not directly addressed in technical streamlining.

At the time of development there were three main types of content being deployed for the Portal; animated, video and 'real-time'. Taking either completed computer rendered sections of video, stitched 360º footage from specially made cameras or utilising real-time environments to showcase human scale spaces and places via graphics engine (Ch 'ng, 2007). Each brings different challenges to both the client facing and delivery experience. Large format animations require massive computing and decoding power to deal with the resolution (Salehi, Zhang, Kurose, & Towsley, 1998), 360º video requires detailed choreographing and viewer awareness to deliver maximum impact (Neng & Chambel, 2010) and real-time environments stress all major aspects of a technical system (Ch 'ng, 2007), while leaving users in unguided and 'without-rail' experiences for the most demanding system and user requirements.

Therefore, this research sought to implement content playback optimisations to help reduce the potential impact on a user's PANS response. Implementations took place across two main areas;

- Real-time content camera redesign
- Optimal resolution for content consumption

An extension of technical streamlining, redeveloping the full dome camera system within real-time games engine environments was evaluated via the ability to represent higher resolution environments in the Soluis dome at higher framerate.

Optimising the resolution for content consumption was a directly user facing experience tested and evaluated via participant blind A/B testing. It involved find the specific balance between screen refresh rate and pixel density within the projection surface. It also stood up to our hypothesis that overall experience is more important than pure quality.

1.3.1 Real-time Content Capture Redesign

As originally outlined in Figure 3, generating 360º within a games engine is not natively supported for the style of rendering and playback the Soluis Portal requires. To overcome this initial content implementations within the Unreal and Unity games engines use specially constructed camera

systems to capture the content. It is not within the scope of this research to fully explore the current methods for 360º rendering but an understanding is required. Using research from Paul Bourke (Bourke, 2009) the inherited Soluis system rendered six individual camera faces and passed them out of the games engine into Touchdesigner to deal with the pixel and vertex processing (Lindholm & Nickolls, 2009), creating either an equirectangular (360º image) or fulldome fisheye depending on requirement. Figure 12 shows an extract from the whole content pipeline detailing the process.

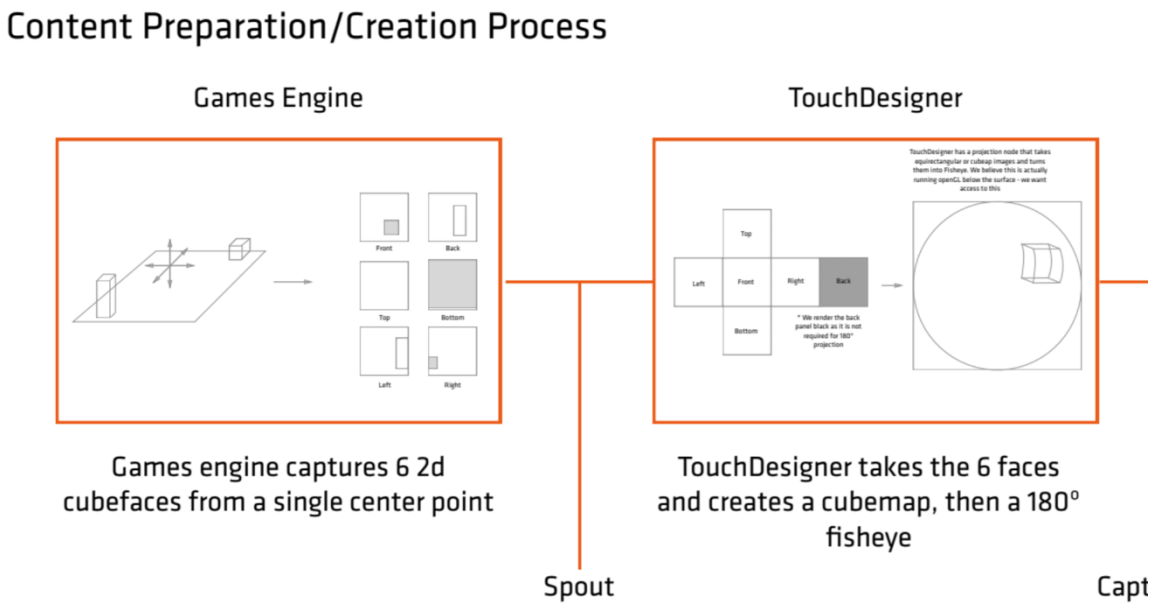

Figure 12: Detailed overview of real-time capture system

Understanding of this process is important for the impact of the changes made during this research, and the user experience improvements. The games engine outputs six individual 90º field of view 2d cameras (usually forward, back, left, right, up and down). These six images are then gathered into Touchdesigner and stitched together into the required output format.

Due to Touchdesigner and Unreal Engine not having a synced rendering system, and the manner in which spout texture sharing works (eg a constantly active link between an image) a situation would arise where a number of the camera faces had rendered the next frame, but the remainder had not. This left a ghosting or lagging effect on the Portal playback, an obvious technical error to users and a break in presence.

The obvious solution was to reduce the amount of information passed between each program to a single image, this way as long as Touchdesigner could process the data fast than 16ms per frame then each rendered frame on the Portal would be shown as delivered by Unreal Engine.
This would also allow for the leveraging of the more powerful GPU utilisation within the games engines. The development would later become known as the 'Portal Plugin' internally, over the time with this research it has taken two forms; The first a stepping stone using the methods outlined above but internally compressed into the fisheye format. The second using direct pixel and vertex shader implementation for the most efficient direct method of processing the data.

The first iteration took the six internally captured cameras and warped them onto a special sphere mesh hidden inside the Unreal Game Scene, a single orthographic camera then captured the result from within the origin of the sphere pointing outwards, towards the desired direction.
Figure 13 gives a profile shot of the inside of the sphere in concept and Unreal. This method of capture was neither an improvement or a sustainable rendering method due to a number of failings in the execution but served as a proof of concept required to create the complete shader implementation in iteration two. The system suffered as it was now having to internally capture six direction cameras, map them into a cubemap and onto a sphere, then recapture that sphere at incredibly high resolution – all whilst rendering the scene. With some optimising it may have been a feasible approach, however its overall improvement to system performance where never fully measured.
The second issue was the method of capture, using an orthographic camera from the internal sphere rendered something known as a hemispherical (or linear) fisheye image. For the Portal and its image to be correctly orientated and represented we needed to utilise something called an angular fisheye, a method for equality spreading the resolution of an image across the fisheyes width (Bourke, 2001). This did not happen in the internal camera build as it compressed the angle of view as it approached the edge of the half-sphere.

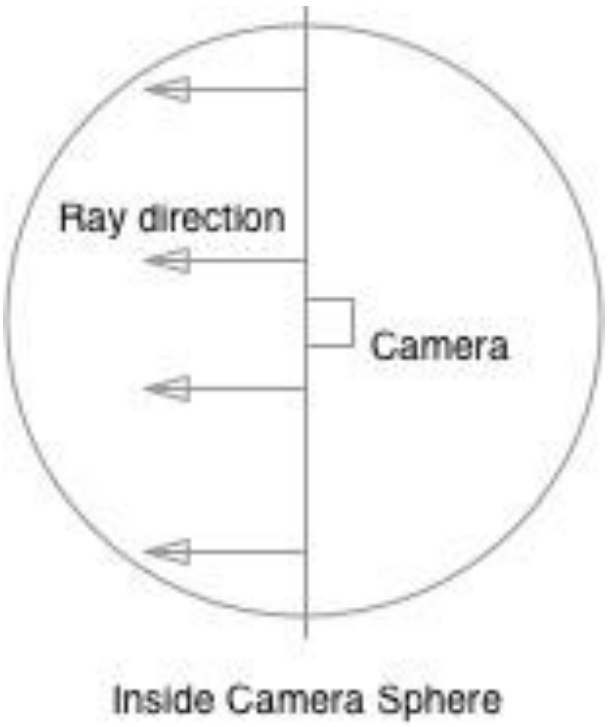

Figure 13: Inside initial plugin capture sphere

The second iteration of the Portal Plugin set to put the theory of the first into a more sustainable, efficient method. Using the same research from Paul Bourke (2001) we were able to implement a completely shader-based version of the calculations. Still using the same six camera operators within the play environment (accurately capturing beyond 90º FOV is an issue of current generation games engines) a cubemap was rendered, equirectangular made and then warped using code rather than physical representation within the game scene. Appendix **Error! Reference source not found.** has an early example of the complete pixel shader code as created for the Portal Plugin.

With the implementation complete, this research was able to move to a single point of connection between the games engine and rending pipeline in Touchdesigner. Meaning that no matter how variable the framerate between the two, the playback would not show tears, stutters or lag based on the cubemaps not rendering in sync. Measuring the technical impact of the change is more complex than listing the numbers as above in the technical streamlining section due to the complexities of multiple rendering system working in tandem. However, with the new camera system both Unreal Engine and Touchdesigner were able to fully maintain a lower than 17ms

rendering time across all actions per frame. The means that both Touchdesigner and Unreal could generate content at up to 60fps.

The real power of the Portal Plugin and its new rendering method came in the form of being able to push the boundaries of playback possibilities and the ability to evaluate the effect on quality on a user's experience.

1.4 Future position of technical development

While this research made many positive changes in both the technical infrastructure, technically delivered experience and reduction in potential impactors on experience. Technical developments have not been measured accurately, and their changes not quantified.

The impact and changes made via the above developments have made profound and lasting changes to the state of the Soluis immersive dome environment. Given the nature of the work, consideration of the outlined hypothesis should be deliberated during any implementation of future immersive, interactive spaces of this nature. However, these claims and perceived effects of changes need to be validated.

It should be proposed that future research should cover exactly this area and aim to analyse and quantify the difference in experience passed on to users via the improvements to technical facilitation alone.

The next chapter will expand and explore the perceived relationship between the current state of the technical development of the IDE in this research and the PANS framework outlined earlier.

2 Bibliography

Audet, S., & Okutomi, M. (2009). The 6th IEEE International Workshop on Projector-Camera Systems (Procams 2009). Retrieved from http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.852.6373&rep=rep1&type=pdf

Bourke, P. (2001). Computer Generated Angular Fisheye Projection. Retrieved May 29, 2018, from http://paulbourke.net/dome/fisheye/

Bourke, P. (2009). iDome: Immerisve Gaming with the Unity game engine.

Bourke, P. (2016). Sphere 2 fisheye. Retrieved May 22, 2018, from http://paulbourke.net/dome/2fish/

Ch 'ng, E. (2007). Using Games Engines for Archaeological Visualisation: Recreating Lost Worlds. Retrieved from http://complexity.io/Publications/RecLostWorlds-f_echng.pdf

Chung, J., & Gardner, H. J. (2012). Temporal Presence Variation in Immersive Computer Games. *International Journal of Human-Computer Interaction*, *28*(8), 511–529. https://doi.org/10.1080/10447318.2011.627298

Deering, M. F. (n.d.). The Limits of Human Vision. Retrieved from http://michaelfrankdeering.org/Projects/EyeModel/limits.pdf

Derivative TouchDesigner. (2018). Retrieved May 22, 2018, from https://www.derivative.ca/

Emergence of Fulldome. (2005). Retrieved from http://c.ymcdn.com/sites/www.ips-planetarium.org/resource/resmgr/pdf-articles/200509FulldomeTheaters-Davis.pdf

Fiala, M. (2005). Automatic Projector Calibration Using Self-Identifying Patterns. Retrieved from https://pdfs.semanticscholar.org/d711/45817e299304aa09ef6c1a9793780da50375.pdf

Lantz, E. (2006). Digital Domes and the Future of Large-Format Film. *LF Examiner*, *9*(8). Retrieved from http://extranet.spitzinc.com/reference/papers/LFExaminerArticleLantz.pdf

Lee, J. C., Dietz, P. H., Maynes-Aminzade, D., Raskar, R., & Hudson, S. E. (2004). Automatic Projector Calibration with Embedded Light Sensors. Retrieved from http://monzy.org/merl/projcal.pdf

Lindholm, E., & Nickolls, J. (2009). NVIDIA TESLA: A UNIFIED GRAPHICS AND COMPUTING ARCHITECTURE TO ENABLE FLEXIBLE, PROGRAMMABLE GRAPHICS AND HIGH-PERFORMANCE COMPUTING. Retrieved from https://fenix.tecnico.ulisboa.pt/downloadFile/3779576765088/IEEEMicro_TESLA.pdf

Neng, L. A. R., & Chambel, T. (2010). Get Around 360º Hypervideo. Retrieved from https://pdfs.semanticscholar.org/3fbf/43a8ab7de61b06d19b797db225af4ef9b93c.pdf

NVIDIA Announces Quadro Pascal Family: Quadro P6000 & P5000. (2016). Retrieved January 13, 2018, from https://www.anandtech.com/show/10516/nvidia-announces-quadro-pascal-family-quadro-p6000-p5000

OpenGL Multi-Context 'Fact' Sheet | Perfect Internal Disorder. (2018). Retrieved May 25, 2018, from https://blog.gvnott.com/some-usefull-facts-about-multipul-opengl-contexts/

Salehi, J. D., Zhang, Z.-L., Kurose, J., & Towsley, D. (1998). Supporting Stored Video: Reducing Rate Variability and End-to-End Resource Requirements through Optimal Smoothing. Retrieved from https://pdfs.semanticscholar.org/4855/9096e5f3d7ed12b46b1d54ee21e653cfcd9d.pdf

Schnall, S., Hedge, C., & Weaver, R. (2012). The Immersive Virtual Environment of the digital fulldome: Considerations of relevant psychological processes. *Journal of Human Computer Studies*, 1–15. https://doi.org/10.1016/j.ijhcs.2012.04.001

Slater, M., & Steed, A. (2000). A virtual presence counter, 9(5), 413-434. *PRESENCE*, 413–434. Retrieved from http://publicationslist.org/data/melslater/ref-49/3deec5219d535442b8.pdf

Smith, R. (2016). Rise of the Tomb Raider - The NVIDIA GeForce GTX 1080 & GTX 1070 Founders Editions Review: Kicking Off the FinFET Generation. Retrieved May 25, 2018, from https://www.anandtech.com/show/10325/the-nvidia-geforce-gtx-1080-and-1070-founders-edition-review/19

Spout. (2018). Retrieved May 23, 2018, from http://spout.zeal.co/

Usoh, M., Arthur, K., Whitton, M. C., Bastos, R., Steed, A., Slater, M., & Brooks, F. P. (1999). Walking > Walking-in-Place > Flying, in Virtual Environments. Retrieved from https://www.cise.ufl.edu/research/lok/teaching/ve-s07/papers/1999-SIGGRAPH-Usoh.pdf

Video Capture Cards | Datapath Video Capture Cards. (2018). Retrieved May 23, 2018, from https://www.datapath.co.uk/datapath-products/video-capture-cards

Vioso - TouchDesigner 088 Wiki. (2018). Retrieved May 27, 2018, from https://www.derivative.ca/wiki088/index.php?title=Vioso

VIOSO software for auto alignment multi projection, edgeblending, media server, warping and softegde. (2018). Retrieved May 27, 2018, from http://www.vioso.com/

Wynn, C. (2018). NVIDIA PROPRIETARY AND CONFIDENTIAL OpenGL Render-to-Texture

OpenGL Render-to-Texture. Retrieved from http://developer.download.nvidia.com/assets/gamedev/docs/opengl_rendertexture.pdf