Friston, S. (2020). Low-Latency Rendering With Dataflow Architectures. *IEEE Computer Graphics and Applications*, *40*(3), 94–104. https://doi.org/10.1109/MCG.2020.298018

Low Latency Rendering with Dataflow Architectures

Sebastian Friston
University College London

Department: Dissertation Impact
Editor: Jim Foley, foley@cc.gatech.edu

ABSTRACT

Recent years have seen a resurgence of VR (Virtual Reality), sparked by the repurposing of low-cost COTS components. VR aims to generate stimuli that appear to come from a source other than the interface through which they are delivered. The synthetic stimuli replace real-world stimuli, and transport the user to another, perhaps imaginary, 'place'. To do this, we must overcome many challenges, often related to matching the synthetic stimuli to the expectations and behaviour of the real world. One way in which the stimuli can fail is its latency – the time between a user's action and the computer's response. We constructed a novel VR renderer, that optimised latency above all else. Our prototype allowed us to explore how latency affects human computer interaction. We had to completely reconsider the interaction between time, space and synchronisation on displays and in the traditional graphics pipeline. Using a specialised architecture – dataflow computing – we combined consumer, industrial and prototype components to create an integrated 1:1 room-scale VR system with a latency of under 3 ms. While this was prototype hardware, the considerations in achieving this performance inform the design of future VR pipelines, and our human factors studies have provided new and sometimes surprising contributions to the body of knowledge on latency in HCI.

1 INTRODUCTION

The concept of Virtual Reality has developed alongside those of the first graphical displays, displays that promised to provide a window into the "mathematical wonderland" of the computer [1].

The archetypical VR system is one that completely subsumes human vision. Pre-dominance of the visual system means that graphics displays are still front and centre. VR is not just seeing however, but believing.

VR differs from other HCI systems in that users integrate the stimuli in a way that they believe, or at least behave as if, it is from a 'real' place – and usually that that place is different from the real world in which they inhabit.

Providing stimuli that can be integrated this way allows us to create a virtual world of our design, that is very real in the eyes of the user. This world can be used for entertainment, training, communication, remote working, psychotherapy - or even just more effective manipulation of the computer's mathematical wonderland.

VR systems use many of the same technologies as other HCI systems, but to present stimuli in a way in which it appears to come from a real, physical place. This means matching the stimuli to the user's sensory expectations. An example of this is how in VR displays provide stereoscopic views from the perspective of the eyeball, so the geometry of the light matches what would be reflected from a real 3D environment around the head. This is quite different from the traditional first-person view, in which users see the world through the window of the monitor.

The synthetic stimuli must match in more ways than just geometrically however, it must also match temporally.

In HCI, *latency* typically refers to the end-to-end delay between a user's action and the perceived response to that action. Latency is unavoidable, and comes from every stage in a discrete computer system: including sensor sampling intervals, processing and buffering [2].

While for many traditional tasks a small delay (on the order of milliseconds) will not be noticeable, latency begins to affect continuous interaction. When a user attempts online control - such as steering or aiming in a game, or even just looking around in an HMD (Head Mounted Display) – they form a loop with the computer. Delays bound the frequency, as they would any other feedback loop. Effectively, the 'bandwidth' of the loop is limited.

This concept is formalised by Fitts' Law – a fundamental observation in HCI that relates models of human motion to information throughput [3].

In addition to reduced performance, latency in VR can induce additional negative effects, including simulator sickness and breaks-in-presence. In the latter, the illusion that the synthetic stimuli is coming from another place breaks down and the utility of the virtual world is undermined [4].

Humans do not expect latency in VR because the real world in which they have developed has no latency, yet computers have unavoidable latencies between every stage.

Studies have been conducted to find the limits at which latency begins to have these effects, but for many years were limited in how low they could go by the technology forming the apparatus. Studies often had to rely on analog simulacrums of virtual reality systems [5].

Modern apparatus are achieving lower real latencies, and the thresholds of perception are becoming clearer. Latency affects different modalities in different ways however. The visual system is one of the most sensitive, but many senses are important for VR. As latency is a function of action, which part of the body is used to interact will also have an effect. Latency is an established and enduring topic in HCI. Even with new apparatus the effects of latency at very low levels is not yet completely understood.

In this thesis we addressed the problem of latency from two directions. The first was to understand the effects and limits of latency in HCI. This was supported by the second, which was to construct an ultra-low latency VR system.

The traditional graphics pipeline is designed for throughput not latency. To achieve our goals, we had to re-consider the trade-off between simulation and sampling, the ordering and synchronisation of the pipeline, and the relationship between time and space in common display technologies.

Beyond this though, VR system components don't work in isolation. To build our system we combined consumer, industrial, and prototype hardware. Iterating through a set of prototypes, we created a 1:1 room-scale VR system with a latency of less than 3 ms.

Our apparatus demonstrates various techniques and considerations for achieving low latency VR systems, and proves it is possible to make systems with such low latency.

Our human factors experiments into low latency question previously held assumptions, and possibly explain artefacts seen but unexplored in prior studies.

This article describes how modelling the relationships between space and time in discrete components presents new opportunities for computer graphics, how we leveraged these to build a state-of-the-art VR system, and what the future of rendering for VR could look like.

2 Low Latency Virtual Reality

No discrete-time computer system can have zero latency, but coupled with predictive compensation, attempts have been made to reduce it to imperceptible levels.

The designs in this thesis are inspired by a lineage of highly specialised rendering hardware, including image warpers [6] and lightfield renders [7].

Such systems reframe the problem of image generation.

Instead of the two-stage process of generating a single image at a fixed point in time, then swapping to the display as an atomic operation, alternate architectures recognise the limits of discrete systems and treat images as an evolving structure. Images are refined with increasing frequency as they near the eye [8]. Frameless rendering originally promised to improve computational efficiency, but moving away from frames is the first step in improving VR displays in a number of dimensions [9].

3 Beyond Frames

When considering a display it is common to think in terms of refresh rate, and it's common, even now, to compare commercial HMDs in this fashion.

Considering a display as having a fixed, absolute update period presents us with a lower bound on latency: no matter how fast the graphics device, a given pixel can never be updated faster than the refresh period.

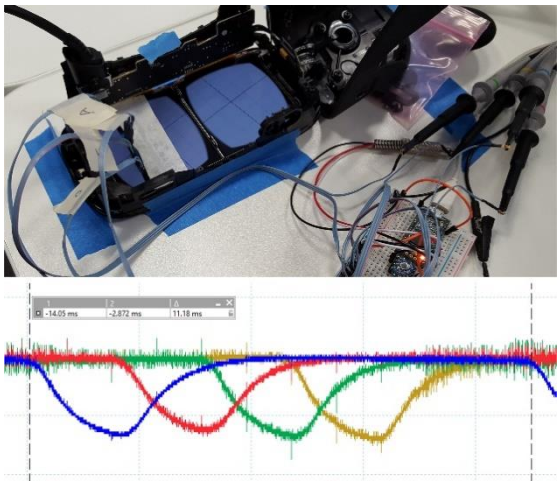

Figure 1 – Profile of an Oculus CV1 display captured by placing four photodiodes across the screen. The horizontal axis is the distance across the display and the vertical axis is the luminance of the area under the diode.

Frame rate however is an abstraction. At the millisecond and pixel level, displays and display technologies have varied behaviour.

The scanout pattern of a display defines a relationship between real-world time and location on a display at which data is or is not visible. We can use this relationship to overcome the limits due to the update period.

We used an Oculus DK2, which has a rolling-scan display. This display illuminates pixels in a narrow vertical band that moves across the display. This is shown in Figure 1, captured with a high-speed camera. The apparent width of the band will depend on the exposure time. The true profile can be captured with, for example, a set of photodiodes and an oscilloscope, as shown in Figure 2 for an Oculus CV1.

Displays do this in order to reduce persistence, and therefore motion blur. The scanout pattern however means we can model when and how long a pixel will be visible to the user. We can then generate a frame with non-linear time, knowing that when the user sees a given pixel it will appear in the correct location, even if the display is moving.

Research subsequent to the thesis has developed techniques that allow this to be done on traditional GPUs using prediction [10], but the original implementation used a deterministic computing platform to do it entirely in real-time.

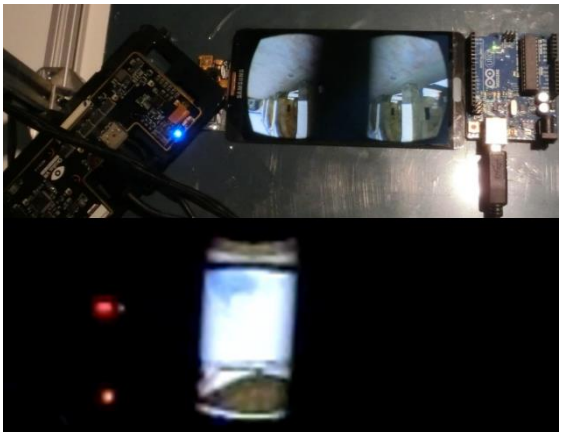

Figure 2 - Disassembled HMD (upper) and a quick exposure close-up (lower) of the display showing a band of illuminated pixels during scanout.

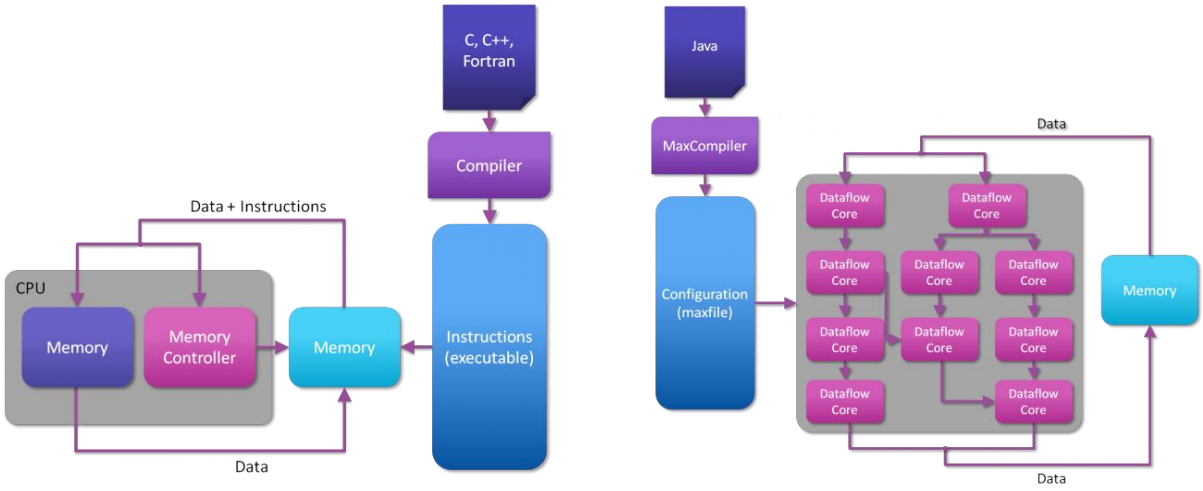

Figure 3 - Comparison of Control Flow (left) vs. Dataflow (right) Computing

4 Dataflow Computing: Just In Time Pixels

Dataflow computing is a parallel computing architecture in which operations are laid out in space, rather than time (Figure 3).

Logically, this form of computing can be seen in node-based programming languages: for example Unreal's Blueprints, Lego Mindstorms' LabVIEW-based NXT-G, or any of the number of node-based shader tools. In Dataflow computers however, this representation is also realised at the silicon level.

Dataflow graphs are described in a high level language then compiled to a spatial computing platform, such as an FPGA (Field Programmable Gate Array) - a gate-level reconfigurable chip. Our platform was provided by our collaborator, *Maxeler Technologies Ltd*, who build both spatial computing hardware and software stacks [11].

Dataflow computing has a history in graphics [12] but recently is only found in specialised applications such as Finite Difference Wave Propagation Modelling [13].

The reason for this is the unique characteristics of the architecture. As Dataflow computers must assign physical logic to each operation, branching is highly inefficient - like older shader units. But for suitable algorithms, they can achieve true parallelism – up to the number of operations in the algorithm itself. This means a dataflow computer can provide a result on every clock tick. The latency or length of the algorithm is unchanged, but the throughput is massively higher than any CPU or even traditional GPU. Further, the throughput is not just high, but deterministic.

This means we can clock the dataflow computer at the same rate as a display, and reliably have a new pixel on each tick, regardless of the length of the algorithm.

In this thesis, the algorithm was short – so short that it was well below the scanout interval. Coupled with our rolling-scanout display, we could generate pixels only a few microseconds before they would be sent, and displayed, allowing us to modify the content of a frame, while it was scanning out.

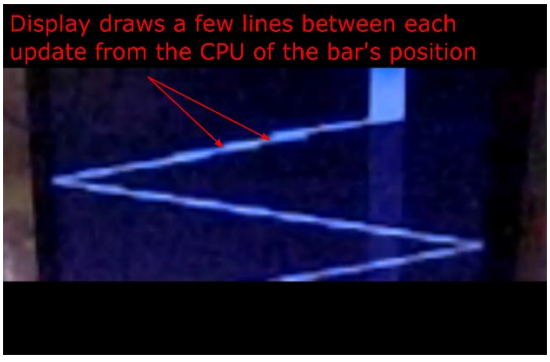

Figure 4 - Frameless image generator showing a white bar moving back and forth across a desktop display.

Figure 4 shows a prototype drawing a white bar moving across the screen. The bar's position is set by the CPU at discrete intervals. The aliasing from line-to-line expresses the

CPU loop frequency in lines – every step indicates a new update.

The ghosting in Figure 4 also illustrates how display persistence can result in motion blur when the same technologies are used to present images in VR.

4.1 Latency and Image Fidelity

Both latency and persistence introduce error into the perceived image. Latency results in spatial differences between what the user should be seeing and what they actually see, while persistence means they see out-of-date content for longer.

In one study, we used a high-speed camera to compare images generated by our frameless renderer to those from the traditional pipeline. Both renderers drew the view from inside a skybox. We captured head motion data from a real user in a HMD to drive the camera and generate the ground truth. For this, we ray-traced a whole-screen image for each tracker sample (Figure 5). This emulated what an ideal display, with zero latency & persistence, and running at 1000 fps, would show at the sample's time.

Using objective Image Quality Measures (IMQ), we showed the frameless renders had significantly higher fidelity than traditional GPU equivalents under motion, when compared to the ground-truth.

Figure 5 - Ray-casted ground truth. All stimuli was synthesised and rendered as if it would be displayed to a real user in an HMD, down to the per-colour distortions to compensate for chromatic aberration in the lenses. This image would have been compared to a real frame such as that shown in Figure 2.

Table 1 shows linear regression coefficients when we model IMQ results as a function of user motion. We fit three orientation terms and velocity, but here show only velocity as the interesting term, as it represents 'sensitivity' to motion.

In an ideal world, the system would always show the correct perspective everywhere at any time, regardless of motion, so the coefficients would be zero.

Based on the actual values and our knowledge of the systems, we make four observations:

1. Higher latencies should be more sensitive to velocity. This is seen in every case except for 1 ms Frameless RMSE. We suspect based on the small effect size this was due to noise, given that we were capturing with a real camera with many uncompensated factors.
2. The more sophisticated and sensitive to structure the IQM, the more we'd expect to see differences between rendering methods. We see this as SCOR and VIF measures show larger differences between Traditional and Frameless than RMSE.
3. As exposure time increases so does the proportion of 'old' pixels, and so we'd expect higher error. This is what we see in the larger coefficients for the 13 ms conditions compared to 1 ms. We also see that they are smaller for Frameless compared to Traditional, since the Frameless renderer would have been updating across the frame as well.
4. This effect should be more pronounced as exposure time decreases. This is because the Traditional renderer continues to scanout out of date content while the framless renderer constantly updates. This is what we see, with the SCOR and VIF being smaller for Frameless than Traditional – so far as to be statistically insignificant for VIF.

One outlier we couldn't explain is the positive correlation for the SCOR metric, since the correlation should be insignificant at best.

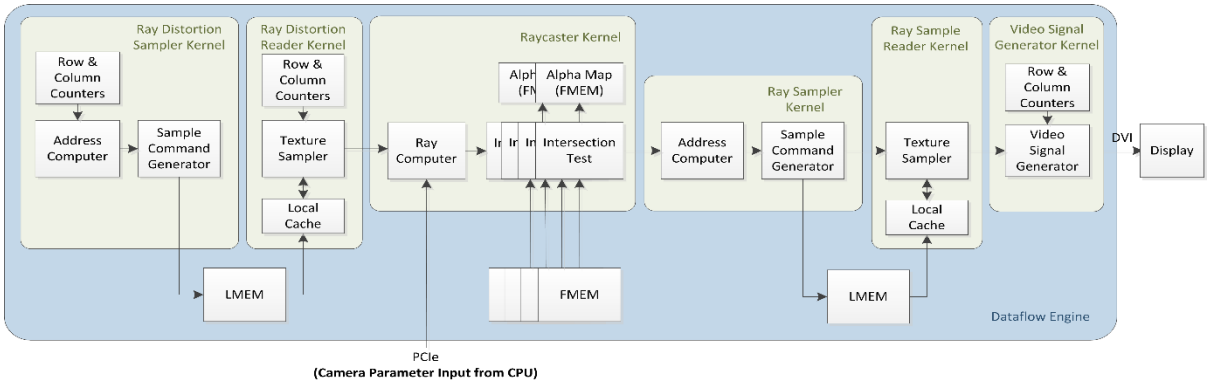

Figure 6 - Architecture of Low Latency Ray-caster

Coefficient Estimates for three IMQ multiple linear regression models ($p < 0.05$)				
Exposure Time	1 ms		13 ms	
Predictor	Frameless	Traditional	Frameless	Traditional
Root Mean Square Error Measure				
Velocity	1.010	1.089	1.008	1.531
R^2	0.434	0.449	0.296	0.383
Spearman's Rank Correlation Coefficient Fidelity Measure				
Velocity	0.039	-0.144	-0.490	-0.802
R^2	0.192	0.215	0.436	0.481
Visual Information Fidelity Measure				
Velocity		-0.033	-0.715	-0.410
R^2	0.155	0.142	0.236	0.108

Table 1 - Correlation Results

An apparent outlier is the 13 ms VIF metric, but this is actually a reasonable reasonable result: the Frameless system will skew frames as they change during a scanout. With a high exposure, VIF - the most sophisticated IQM - may consider this distortion more egregious than those due to differences in time. When the number of older pixels is reduced in the 1 ms condition, this effect disappears. Future studies may be improved by designing new metrics for frameless rendering.

5 An Ultra-Low Latency VR System

The nature of dataflow computing forces us to address the trade-off between simulation vs. sampling common to many fields, but represented in computer graphics by the *rendering-continuum* [14].

The most efficient dataflow graph is one with the fewest conditionals. Our end goal was to design a pure image-based renderer, such as a lightfield renderer. In a general-purpose VR system this would form the end-stage of a more elaborate rendering pipeline or cascade, discussed further in Section 8. Over a set of prototypes we iterated from sprite-based compositors towards this goal. By the project's end we had a hardware accelerated ray-caster, with the ray-intersection test set approximating the sampling directive in an image-based renderer. While we would have liked to iterate further, the ray-caster had nearly ideal temporal performance, and supported all the environments necessary for our experiments.

Our ray-caster architecture is shown in Figure 6. In a dataflow graph, a unit of data enters the graph and is transformed as it moves (left to right in Figure 6) through a sequence of operations. In our design, these units start as pixel locations and are transformed into pixel colours. Locations are generated in a sequence matching the scan of the display, with the resulting colours transmitted via DVI. The core of the ray-caster is a set of sequential closed-form ray-intersection tests. The tests are sequential because it was easier to code them that way. They could just as easily be parallel with an accumulator – on a spatial computing platform they run with true parallelism, and take the same amount of space, either way. The result of the tests defines the environment map location to sample for that pixel. The sample determines the colour of the pixel scanned to the display.

While responsiveness is important in computer graphics, it is rare to encounter a problem with hard real-time requirements. Our graph had to synchronise with a physical display; DVI does not tolerate jitter. The challenge was a design that overcame the non-idealities of the platform, such as the DDR controllers, that break the abstraction of determinism.

For example, the memory bandwidth is less than the display's, so we had to implement a caching system. Since the ray order is determined by the scanout pattern, we could use a ray coherency based cache to reduce memory accesses. The ray step size across a surface is dependent not only on camera field of view, but also distance, so the cache uses mip-maps chosen by the previous step-size to ensure an average hit ratio of *8:1*. Imperfections were smoothed with a FIFO buffer. Buffering introduces latency, but at the pixel level by only trivial amounts: on our display one line (1920 pixels) was equivalent to 6.8 µs.

Coherency-based caching is only the start however; the features of the spatial computing platform provide the opportunity to make unique optimisations.

For example, we can use determinism to break dependencies between sections of the graph by duplicating logic. This can be seen between the Ray Distortion Kernels. The cache logic is implemented on both sides, so the upstream logic knows what memory pages have to be sampled and when, and the downstream logic can rely on the upstream logic issuing a read request for them, without the two ever having to communicate. A similar design pattern is seen at the end of the graph for the generation of the DVI control signals.

The platform also allows for truly parallel memory accesses because separate memory controllers can be created for each physical DDR module.

The camera view is set by the CPU and used by the function that transforms pixel locations into ray parameters.

On GPUs, a post-processing stage distorts images by the inverse transform of the HMD's lenses before scanning to the display. As we implement a ray-caster, we can map between the physical and virtual viewport locations before the ray parameters are defined.

This avoids a synchronisation stage; but further, because the latency of a dataflow graph directly depends on the number of downstream operations, our lens distortion has no effective computational latency either.

The camera is updated by the CPU asynchronously via dual-port memory. The update frequency is limited by the fastest tracker – the DK2's Inertial Measurement Unit (IMU) at 1 kHz - well below the period of an individual frame.

The counters that feed the graph are free-running. The only communication with the CPU are the camera updates, and these are asynchronous. Unlike a traditional GPU that is synchronised to the CPU, our renderer is synchronised to the display. Our graph will continue to drive the display even after the CPU application exits - albeit with a static viewpoint.

An image generator however is just one component of a VR system.

Though asynchronous, the image generator occupies much the same place as a traditional GPU in the system architecture (Figure 7). Figure 8 shows the real hardware. The spatial computer is in the form of a PCIe co-processor card. The DK2 connects to it via a custom board that adapts the physical interface (electrical and form factor) between the headset and the card's edge connector.

For tracking, we use a PhaseSpace active-marker motion capture system running at 960 Hz. The PhaseSpace latency of ~3 ms is the largest of any component in our system. A linear complementary filter fuses the PhaseSpace and HMD IMU data to reduce spatial jitter.

A real-time thread running on a CentOS multi-core PC updated the camera and primitive parameters as fast as possible. As seen in

Figure 4, this was on the order of microseconds.

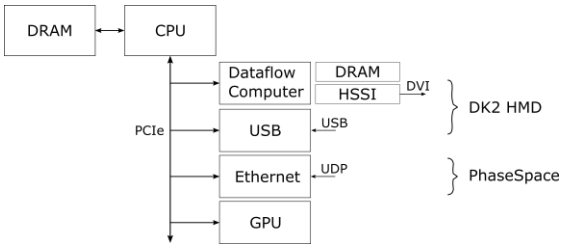

Figure 7 – VR System architecture. The dataflow computer takes the place of the GPU for the headset.

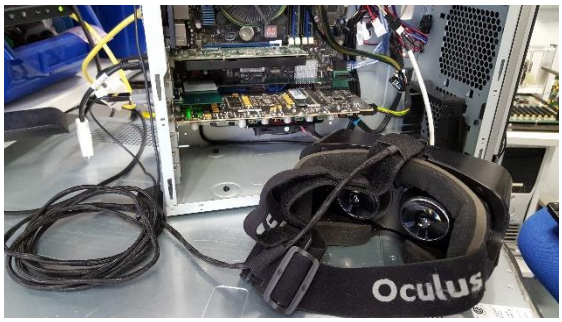

Figure 8 - Dataflow PCIe co-processor card connected to a DK2

The result was a working room-scale VR system, supporting 1:1 locomotion across 18m^2. An example environment is shown in Figure 9.

Figure 9 - Example of a depopulated virtual mimic of our lab used for a walking-short study

6 The Effects of Low Latencies

One of the first experiments performed using a dataflow renderer prototype was not in VR at all. The Fitts' Law model of target acquisition is a reliable measure in HCI, and we used this to investigate latency on a traditional desktop at levels below which it had been explored before.

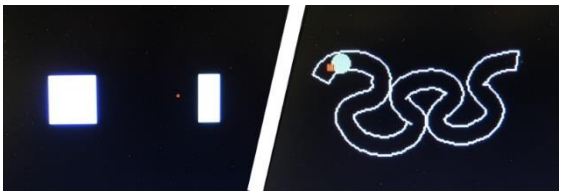

Figure 10 - Stimuli for the Pointing and Steering Tasks

The protocol for such an experiment is to hide the users hand so their only visual feedback is the cursor on the screen (Figure 10). Users then performed target acquisition tasks and path following tasks as the latency was varied.

Our system achieved a latency of 6 ms, with the display scanout accounting for the majority of this time. For this experiment we used a 120 Hz monitor with a global scan.

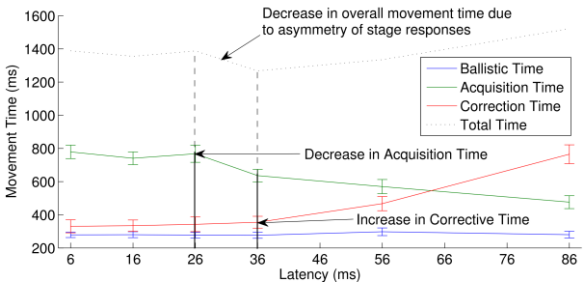

Figure 11 - Fitts' Law trial motion profile

As the latency decreased our results showed a highly repeatable and unexpected non-linearity in performance. The non-linearity is unexpected because it was previously hypothesised that the effect of latency was linear down to zero, as a consequence of it not following Weber's Law. Indeed, variants of Fitts' Law have modelled lag linearly [15]. The true profile is shown (as the black, dashed line) in Figure 11. User performance levels out around 26 ms but not before undergoing an anomalous peak in performance.

Considering total execution time, the cause is not obvious, but by breaking down the users' motion using the velocity profile, we can see a statistically significant pattern emerge. Briefly, that latency causes a velocity overshoot in a specific stage that can be compensated for under a range of conditions, coincidentally improving performance. The effects of latency do not stop here though, and despite the apparent peak, latency is not beneficial: in any other paradigm the same interference would more likely be an impediment.

Such results have implications for HCI researchers as the non-linear profile means we must study latency at higher resolutions, the lower it becomes.

7 Future of Latency in Virtual Reality

As the new wave of VR adoption progresses, so does the awareness of the importance of latency. Traditionally considered in HCI as a bandwidth issue, VR headset manufacturers, SDK designers and developers are recognising the severe effects of latency on the whole user experience, including on the vulnerability to simulator sickness and breaks-in-presence.

The recognition can be seen in technologies such as *timewarping* in the Oculus VR SDK, which refines frames using image warping. Using up to date tracking data that was not available when the frame was first generated, and synchronising with the start of the scanout to minimise the time between refinement and display, this feature acknowledges the importance of the relative real-world times between image generation and perception.

Currently, though, the perception is still that latency is a single per-frame value. In practice, the lowest latencies will be achieved by modelling display behaviour and considering it as an integral part of the pipeline.

In subsequent collaborative work, the principles described above were adapted to traditional GPUs [10]. GPUs remain frame-based renderers, but the implementation utilised ray-tracing in the fragment shader to generate images non-linearly in space and time (Figure 12). The linear rasterization stage no longer sampled the image but became an acceleration pass, constraining the ray-tracing problem to make it feasible. Prediction was used to change the view across the display, reducing latency due to scanout, while the lens distortion was applied in a pre-processing stage as above, reducing computational latency and avoiding the synchronisation.

While a number of low-latency systems have been proposed over the years, ours is still (c.a. 2020) the only one that supports a functional, room-scale virtual environment. We hope shortly to adapt our recent work on non-linear rasterization on GPUs to popular game engines, in order to drive larger and more sophisticated virtual worlds with equally low perceptual latency.

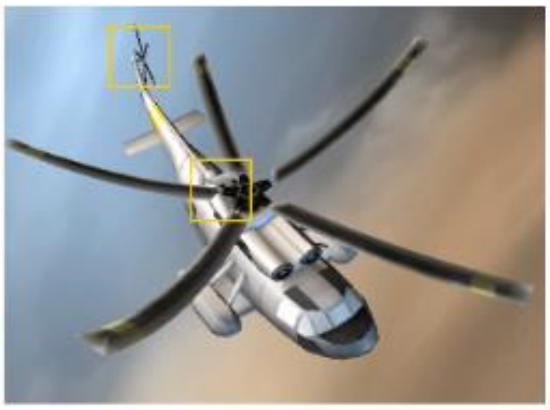

Figure 12 - Non-linearly rasterised image. Traditionally such images would be produced by warping a linear rasterization. This would introduce artefacts in highly detailed areas such as the highlighted insets above, but the fragment-shader ray-traced image is pixel perfect.

8 Future of Graphics Pipelines

Non-linear time is only one aspect of how future pipelines could benefit VR, and latency in VR.

In our ray-caster prototype we showed how latency could be reduced by pre-transforming the sampler where non-linear rasterization is available.

Non-linear rasterization is just one part of future pipeline capabilities. We expect that as

the pipeline evolves it will move away from the concept of frames. An alternative concept is that of our group's Ambient Fields [16], in which we do not render an image at a single point in time, but in a higher dimensional representation that covers an interval in space and time around the user's viewpoint. This representation can then be sampled asynchronously by smaller, tighter loops, perhaps highly coupled with the display. Such a pipeline could not only provide lower latency, but would be far more efficient than current implementations, that often throw away the information-rich fragment buffer each frame.

Other groups have imagined future pipelines as a series of cascaded transforms [8]. The father into the pipeline, the simpler the operations become, but correspondingly the higher frequency they can run, and the lower latency they have.

Of course, these ideas and others are not mutually exclusive. The ideas of cascading stages, ambient fields and non-linear rasterization will be integrated in a new project beginning this year to explore future pipelines. In this architecture what we present would be a near-end stage in a longer pipeline, sampling an environment map of higher dimension.

9 Future of Scene Representation

The image generation component is not the only thing that stands to be revised. As VR technologies become more mature, so do the use cases of *mixed reality*. Investment in displaying virtual worlds is matched by capturing the real one with higher fidelity. Stereoscopic, lightfield and voxel based volumetric capture are all being explored. These higher dimensional captures of the world are more abstract than traditional video and have increased scope for integrating with synthetic content.

A goal we did not have time to achieve was to make full use of the memory of our reconfigurable platform. With 10's of GB of low-cost DDR memory our single FPGA card could demonstrate the potential for sampled representations such as lightfields. To do so would only require substituting the ray-primitive intersection test for a more abstract sampling function.

10 Summary

The aim of Virtual Reality is to generate stimuli that the user can believe, at least at some level, is coming from another world, rather than the interface itself. For this to be successful, stimuli must be realistic in a number of dimensions not traditionally considered in computer graphics. One of those is temporal. Traditional graphics pipelines have been optimised for throughput rather than speed, but recently the importance of temporal realism is being recognised in more and more quarters.

This thesis was concerned with latency in virtual reality, and required re-considering many of the assumptions of the traditional pipeline and its abstractions.

The future of VR will require re-evaluating these abstractions, changing where and how we represent the world, and seeing models of the system's physical dynamics become an integral part of the pipeline.

This thesis only scratched the surface, but we hope its conclusions, considerations, questions and demonstrations can help to inform the future of VR rendering.

11 Acknowledgements

This work was funded by the EPSRC (EP/G037159/1) and Maxeler Technologies Ltd. I wish to thank all of the individuals I was fortunate enough to call colleagues throughput my doctorate. My especial thanks to Per Karlstrom and Georgi Gaydadjiev at Maxeler, and most of all to my supervisor at UCL, Professor Anthony Steed.

12 REFERENCES

[1] I. E. Sutherland, "The Ultimate Display," in *Proceedings of the Congress of the Internation Federation of Information Processing (IFIP)*, 1965, pp. 506–508.

[2] M. R. Mine, "Characterization of end-to-end delays in head-mounted display systems," 1993.

[3] R. W. Soukoreff and I. S. MacKenzie, "Towards a standard for pointing device evaluation, perspectives on 27 years of Fitts' law research in HCI," *Int. J. Hum. Comput. Stud.*, vol. 61, no. 6, pp. 751–789, 2004.

[4] M. Slater, "Presence and The Sixth Sense," *Presence Teleoperators Virtual Environ.*, vol. 11, no. 4, pp. 435–439, 2002.

[5] J. Jerald, M. Whitton, and F. P. Brooks, "Scene-Motion Thresholds During Head Yaw for Immersive Virtual Environments," *ACM Trans. Appl. Percept.*, vol. 9, no. 1, pp. 4:2-4:23, 2012.

[6] M. Regan and R. Pose, "Priority rendering with a virtual reality address recalculation pipeline," in *Proceedings of the 21st annual conference on Computer Graphics and Interactive Techniques - SIGGRAPH '94*, 1994, pp. 155–162.

[7] M. J. P. Regan, G. S. P. Miller, S. M. Rubin, and C. Kogelnik, "A real-time low-latency hardware light-field renderer," in *Proceedings of the 26th annual conference on Computer Graphics and Interactive Techniques - SIGGRAPH '99*, 1999, pp. 287–290.

[8] P. Lincoln *et al.*, "From Motion to Photons in 80 Microseconds: Towards Minimal Latency for Virtual and Augmented Reality," *IEEE Trans. Vis. Comput. Graph.*, vol. 22, no. 4, pp. 1367–1376, 2016.

[9] E. J. S. Zagier, "Defining and Refining Frameless Rendering," 1996.

[10] S. Friston, T. Ritschel, and A. Steed, "Perceptual rasterization for head-mounted display image synthesis," *ACM Trans. Graph.*, vol. 38, no. 4, pp. 1–14, 2019.

[11] Maxeler Technologies Ltd, "Programming MPC Systems." 2013.

[12] P. J. W. Ten Hagen, I. Herman, and J. R. G. De Vries, "A dataflow graphics workstation," *Comput. Graph.*, vol. 14, no. 1, pp. 83–93, 1990.

[13] O. Pell, J. Bower, R. Dimond, O. Mencer, S. Member, and M. J. Flynn, "Finite-Difference Wave Propagation Modeling on Special-Purpose Dataflow Machines," *IEEE Trans. Parallel Distrib. Syst.*, vol. 24, no. 5, pp. 906–915, 2013.

[14] M. Zwicker, M. Gross, and H. Pfister, "A Survey and Classification of Real Time Rendering Methods," 2000.

[15] I. S. MacKenzie and C. Ware, "Lag as a determinant of human performance in interactive systems," in *Proceedings of the SIGCHI conference on Human factors in computing systems*, 1993, pp. 488–493.

[16] A. Steed, V. Pawar, S. Friston, and M. A. Srinivasan, "Ambient fields: representing potential sensory information," in *2016 IEEE 9th Workshop on Software Engineering and Architectures for Realtime Interactive Systems (SEARIS)*, 2016, pp. 1–2.