IMAGE 2010 Conference

SHADOW MAPPING IN HIGH PERFORMANCE VISUAL SIMULATIONS

Cody Starr
Presagis
Texas, USA

Anthony Hinton, Neil McLaughlin and Steve Butrimas
Presagis
Texas, USA

ABSTRACT

In 2009, Presagis added shadow mapping support to two products including its visualization toolkit Vega Prime and turnkey image generator Lyra. A brief background on shadow mapping and its caveats are presented here, followed by descriptions of various techniques employed to address common problems with shadow mapping. Strategies are discussed on how to balance high frame rates with key training objectives. Finally, the roles shadows play with Night Vision Goggles (NVG) and Infra Red (IR) wavebands are discussed. Working illustrations from Vega Prime are provided.

INTRODUCTION

Shadows play a critical role in how one perceives the world. Shadows help to define the shape and dimension of objects in the environment as well as the spatial relationships between them. As such, correctly depicting shadows in a visual simulation is of significant training value. Historically, rendering time-of-day accurate shadows within a 60 Hertz visual application was difficult. However, with recent hardware advances and rendering techniques it is now possible to render shadows while maintaining interactive and deterministic frame rates.

In addition to the general problem of delivering quality shadows in real time, there are specific challenges to shadowing versus other types of interactive applications. For example, large visibility ranges that are standard in helicopter and fixed wing trainers make full scene shadowing difficult to support. In most military trainers today one must also consider shadows in non-visual wavebands such as NVG or IR. Managing the draw load imposed by shadow mapping to maintain 60 Hertz requires typical performance versus quality tradeoffs and adequate control mechanisms to allow trainers to utilize shadows in areas that maximize training value.

The number of shadowing solutions is vast and each has their pros and cons. This paper will review only the relevant, high level concepts of real-time shadowing before moving on to discuss the shadow mapping approach. Please see the references for further background reading.

BACKGROUND

In the real world, objects cast shadows onto both other objects (*inter-object shadowing*) and themselves (*self-shadowing*). Shadows move based on light source positions relative to objects in the scene (*occluders*) and bend across faceted or curved surfaces. Primitive approaches such as planar shadow techniques only offer inter-object shadowing and shadows are clamped to a single planar surface. Meanwhile, horizon mapping [1]and bump mapping [2]only provide self-shadowing. Other shadowing techniques suffer from being static in nature where shadows are "baked in" with the scene content. Light mapping and polygonal modeling of shadows are two classic examples. For those few techniques that manage to provide it all, complex implementations, restrictions, or corner cases get in the way.

Among existing solutions, the most prolific shadow rendering method employed in games and training applications is *shadow mapping*. To determine whether or not something is "in" a shadow, the algorithm observes whether the light source (as seen from the view of the camera) is obstructed. In computer graphics this test is usually done on a per-pixel basis where only pixels that can "see" the light source are illuminated. The shadow mapping implementation uses a multi-pass rendering algorithm. In the first pass the scene is rendered from the point of view of a light source and depth information is captured as a texture (i.e., depth map or shadow map). In the second pass the scene is rendered from the observer's

Presented at the **IMAGE 2010** Conference
Scottsdale, Arizona – July 2010

vantage point and the depth map is projected into the scene. In a pixel shader the current fragment's depth is compared against the shadow depth. If the current depth is greater than the shadow depth, the fragment is in shadow; otherwise, the fragment is not shadowed.

Shadow mapping's popularity stems from it providing all the major characteristics of real shadows – shadows correctly project across surfaces, self and inter-object shadows are present, and the shadows change based on light position. Furthermore, the core algorithm is simple to implement, fast for the quality achieved, and imposes no modeling restrictions on the content.

SHADOW MAPPING ISSUES

Despite its widespread adoption shadow mapping isn't without its perils. Primarily, it suffers from a myriad of aliasing issues that must be minimized so image quality is maintained. There is a vast amount of literature that covers these problems and furthermore proposes variations to the base shadow mapping algorithm to address the issues. Below is a quick summary of the types of aliasing that occurs.

Perspective aliasing: This sampling problem is due to the image space nature of shadow mapping in which objects in the distance get the same number of pixels as nearby objects. This manifests itself in jagged or "blocky" shadow edges, especially for those objects that are close to the camera which need more shadow resolution. [3] [4] [5]

Figure 1: An example of perspective aliasing. While the shadow map resolution of 512x512 might be sufficient for the vehicle in the background, the shadow cast by the tank in the foreground lacks detail. Cascaded Shadow Maps, discussed later, can be used to improve this issue.

Projection aliasing: This under-sampling problem also causes aliasing and is due to insufficient resolution on polygons that are parallel to the light direction. [4] [5]

Temporal aliasing: Also known as "shadow shimmering", is a form of projection aliasing that is seen as visible flicker or shadow edges "swimming" as the camera or object moves. [6]

Incorrect self-shadowing or *"shadow acne":* Another sampling problem is due to differences in sampling rates between the shadow map and eye-space view. Depth precision problems (*depth quantization)* along with integer to floating point round-off errors appear as Moiré-like patterns in the shadow. [5]

Figure 2: A bad case of shadow acne.

IMPLEMENTATION POST-MORTEM

While revamping the dynamic shadow solution, the Vega Prime team was faced with many challenges. Given the vast amount of literature related to shadow generation, simply choosing which approaches to employ is a large task unto itself. To slightly thin the field of possible approaches, it was decided to constrain the initial solution to casting shadows from a single light source, positioned infinitely far way such that all light rays come from a single direction. This constraint is acceptable when simulating shadows cast due to occlusion from the sun and moon. Next, the team generated a survey of techniques, all of which originated in the game industry or academia and several popular approaches were considered. [7] [8] [9] [10] [11] [12] [13] [14] [6] [15] Prototypes were then written which implemented each algorithm. From there a divide and conquer approach was taken where developers modified the prototype by implementing one of the variations and using it to enhance the basic shadow

mapping algorithm. Using agile development practices, frequent iterations occurred each ending with an evaluation by product stake holders and developers. The review process assessed visual quality, performance impacts, compatibility with other techniques, and integration complexity for each of the solutions. This process continued for several months, eventually drawing the line between additional research time and the business pressures of release schedules and serving customers.

Once a subset of solutions was picked, integration into the Vega Prime toolkit ensued. This led to challenges unique to visual simulation applications. Support for coordinate systems, multichannel rendering, integration with existing shader techniques, special effects, and sensors are examples. Also, since Vega Prime is a toolkit, considerations were made to simplify the interface that was exposed to the end user. Both a GUI front-end and C++ API were provided and will be discussed later.

Of the techniques explored, the team selected the following for inclusion in Vega Prime 3.0.

Cascaded Shadow Maps: Abbreviated CSMs, the technique addresses perspective aliasing and shadow acne issues by dividing the view frustum into sub-frustums, each with their own shadow map. By using multiple shadow maps to cover an area, more control is given over where shadow map resolution is leveraged.[2] To greatly reduce temporal aliasing, Vega Prime employs the modifications to CSMs found in [6].

Figure 3: An improvement compared to Figure 1. In this image the cascaded shadow map solution captures shadowing information using four shadow maps versus just one. The result yields a substantial increase in detail in the tank shadow.

Percentage Closer Filtering: PCF is a filtering method based on bilinear interpolation the depth compare. It is a hardware accelerated way to filter shadow maps and results in improvements to projection aliasing and softening of perspective aliasing. [13]

Random Blur: A preprocessing effect applied to the shadow map(s) to soften aliased edges. [11] [16] Multiple samples are taken using a disk of random points. These samples are then averaged to create a percentage in shadow.

Figure 4: Blurring the shadow map helps to reduce aliasing artifacts as seen when comparing the bottom image (blur radius of 2 pixels, with 8 samples or "taps") to the top image (no blurring).

The rationale behind the selection of this particular combination of techniques was based on a myriad of factors. First, the three techniques can be used in conjunction with one another to improve overall shadow quality. PCF in particular was an easy selection because it is explicitly support in hardware. Also, of the techniques investigated, our selections were found to provide sufficient quality while still providing mechanisms to throttle quality for rendering performance. Finally, it was

not reasonable to allow the team to prototype every possible technique so the focus was given on those approaches being adopted by today's high-end game titles. [11] [17]

PERFORMANCE MANAGEMENT

In high performance visual applications, maintaining a good frame rate takes precedence over details added for realism. Since shadow mapping is a multi-pass rendering approach, it adds significant overhead that must be accounted for and managed to maintain 60 Hz. Fast jet and helicopter applications make matters worse due to their far visibility requirements that range from 30-150 kilometers or more. Last there are the high resolution and anti-aliasing requirements which, on many image generators, start at HD and move up approaching eye-limiting resolution. Despite fantastic advances in graphics hardware over the last decade, these factors combine to tax even today's hardware.

The performance versus quality ratio should be weighed against the training objectives of the application. Ultimately, the application writer must decide where shadows add the most training value and tailor the configuration settings to achieve maximum quality at 60 frames per second.

In Vega Prime the customer is given ample interface to manage both quality and performance of shadows in their application. The interface consists of both a low level, implementation specific interface for advanced users and an abstracted layer providing predefined settings that are presented on a moving scale of Quality versus Performance.

The primary driver of performance in shadow mapping is pixel fill, which in turn is dictated by shadow map resolution, the number of shadow maps, and the amount of content rendered in the shadow map(s). Within Vega Prime, the application writer has control over several variables that combine to determine both shadow map coverage and resolution. As mentioned earlier, control over these variables are exposed in the toolkit as API and within the pre-runtime GUI front-end to Vega Prime, known as LynX Prime.

Shadow Distance: specifies the range to render shadows. A shadow map with large coverage relative to its resolution results in perspective aliasing or a "blocky" shadow appearance.

Shadow Map Resolution: controls the size of a depth map texture generated during the initial render pass. In a given coverage area, higher resolution shadow maps result in better pixel quality.

Number of Shadow Maps: In a cascaded shadow mapping scheme this corresponds to the number of sub-frustums created to cover the view frustum out to the shadow distance. Each sub-frustum gets its own shadow map which mandates its own render pass. The main advantage to using additional "cascades" over simply increasing the shadow map resolution is that coverage of each shadow map can be controlled in a non-linear fashion giving the opportunity to allocate more resolution near the eye point where it is needed.

Selective Shadow Casters and Receivers: At a dataset level, the user can specify in LynX Prime or through the runtime API which geometry is considered in shadow mapping computations. While similar "tagging" of content could have been accomplished in a modeling tool, it was decided that providing this flexibility in the runtime would allow quicker iterations of performance tuning. Geometry capable of casting shadows onto other geometry must be included in the depth render pass while shadow receivers must incur the fill cost of the shadow map texture look up and comparison with its depth value.

It is easy to see how all of these variables combine with screen resolution to introduce a significant rendering cost. While on the surface it may appear that shadow quality and rendering performance are mutually exclusive, there are fortunately a myriad of filtering techniques other than increasing shadow map resolution. These include Gaussian blur, Percentage Closer Filtering, and anti-aliasing techniques. Users can control the filtering through the following interface:

Number of "Taps": the number of samples used for performing Percentage Closer Filtering. The more samples used, the softer the shadow edges.

Blur radius: used to control the amount of blurring applied with larger numbers resulting in a blurrier shadow.

Cascade Blend Factor: used to determine the locations of cascade "splits" in the view frustum. The location of sub-frustum splits ultimately determines where shadow map

resolution exists. Vega Prime can use either a logarithmic or linear function to spit the view frustum into *n* shadow maps. However, using the Cascade Blend Factor allows for a weighted average of both which yielded the best visual results. Compared to increasing the number of shadow maps or shadow map resolution, achieving image quality via filtering and intelligent allocation of existing pixel resolution is more efficient for rendering purposes.

SUPPORTING NON-VISUAL WAVEBANDS

The majority of today's military visual applications support more than the typical visible range of 390-750 nanometers by simulating various sensors. While these allow one to see information not detectable to the human eye, most sensors produce images that lack high frequency detail and contain noise artifacts. As such, one could argue that shadows become even more important in sensor wavebands since they add much needed depth and detail to the image. Rendering shadows within non-visual wavebands requires consideration of how both reflected and emitted energy are affected by object light source occlusion.

For those sensors sensitive to wavebands near the visible spectrum, reflected energy dominates. For these cases the shadow problem can be treated the same way as in the out-the-window case. Night vision devices, which are sensitive to energy in the 600-900 nanometer range, fall into this category. Vega Prime achieved shadowing in a night vision environment by simply adding the shadow pass into the rendering pipeline and ensuring the shadow computations did not conflict with all the other image based effects required like photocathode and ion noise, gain and level, halos, etc.

Figure 5: Shadows in NVG look like those in the visible spectrum. Notice both the self-shadowing of the helicopter engine on the fuselage and the cast shadow on the ground.

For those sensors that detect medium or long wave infrared energy one must consider not only attenuated reflected energy but thermal emissions as well. Shadows in a thermal sensor are indicative of temperature differences along a surface and not just occlusion. For example, the ground underneath a stationary vehicle is cooler compared to ground directly exposed to sunlight.

To represent thermal shadows, Vega Prime leveraged an I24 solution that provides per-texel thermal emissions for any time of day for any altitude, orientation, or material mix in real time.[17] In addition to attenuating reflected light, the captured occlusion information was used to appropriately attenuate surface temperatures, per-texel, based on the material properties of the surface, environmental conditions, and simulated waveband.

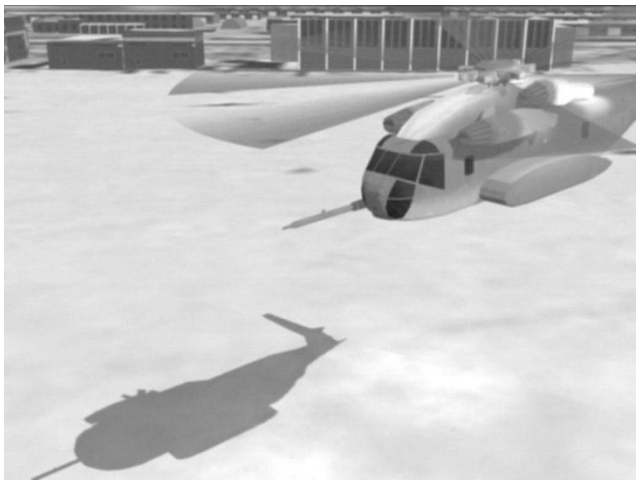

Figure 6: Shadows in infrared. Surface temperatures of objects in the scene are attenuated based on occlusion from sunlight.

Thermal wavebands have an additional shadowing condition that proves more challenging. Transient thermal shadows are possible where a surface occluded for some time becomes no longer occluded yet still exhibits a shadow, since the surface temperature is still cool compared to its surroundings. Over time the shadow will fade as the surface heats up and reaches equilibrium with the environment. Simulating this phenomenon is more difficult because it requires the application to remember the occlusion history for every pixel over an extended period of time. Furthermore, since different materials heat

and cool at varying rates, the time a thermal shadow persists will vary per material or per-texel. While I24 supports per material heating and cooling variation, transient thermal shadows are not in our initial implementation and are slated for future work.

CONCLUSIONS AND FURTHER WORK

In the near term, improving shadow mapping quality while maximizing performance will continue to produce a plethora of new techniques and literature. Application writers will still have to juggle trade-offs in performance, quality and realism. Fortunately, the bar will continue to rise as improved hardware support and new algorithms focused on faster multi-pass rendering will facilitate better looking shadows covering more of the rendered scene.

Future work requires further evaluation of new techniques that have surfaced since the release of Vega Prime 3.0. Better techniques for dynamic, full-scene shadows over the large visibility distances typical in visual simulation are needed. Also, methods of shadow generation for omni-directional light sources need improvement as today's solutions require many shadow maps for a single source in order to cover all directions. Also, deferred rendering approaches that are growing increasingly more popular in games allow for tens or hundreds of light sources to exist in the scene. [18]Obviously this presents the problem of generating shadows for such a large number of lights.

Ultimately, shadow generation will benefit from the upcoming real-time computer graphics paradigm shift between raster-based rendering and direct ray tracing.

As new hardware supports real-time ray tracing at 60 Hertz, techniques like shadow mapping will likely be simplified, since many of the precision issues that are a function of its raster based nature will be relaxed.

AUTHOR BIOGRAPHIES

Cody Starr serves as the subject matter expert on visualization at Presagis with 10 years in the visual simulation industry. He began his career as a visual systems software developer with MultiGen-Paradigm in 2000 and has focused on turn-key image generation, shader technology, night vision simulation, real-time rendering techniques and optimization. Also during his tenure, Cody served as technical lead on many projects including Presagis' visualization flagship product, Vega Prime. Since Presagis' inception in 2007 until recently, Cody was a Director of Product Development, managing the engineering teams responsible for Presagis' visualization product portfolio – Vega Prime, Sensor Prime, Lyra, and Thea. Cody holds an M.S. in Visualization Science and a B.S. in Computer Science, both from *Texas A&M University* in College Station, Texas.

Anthony Hinton is a Senior Software Developer at Presagis where he works in the visualization group. He has eleven years' experience in the software industry in the 3D graphics, cryptographic, and satellite communication fields. Anthony received a B.S. in computer science engineering from *University of Texas* at Arlington in Arlington, TX.

Neil McLaughlin is a Senior Software Developer who has focused on 3D graphics since 1994 when he added motion-controller support to top games such as *Dark Forces*, *Descent*, and *Mechwarrior 2*. Between 1997 and 2002 Neil developed and patented t*he Figment3D Rendering System*, a suite of parametric modeling tools. More recently, Neil has focused on quad-tree based meshing, shader languages, lip synchronization, collision detection, and shadow mapping. Neil received a Bachelor of (Computer) Science degree from the *University of (Massachusetts) Lowell* in 1993.

Steve Butrimas is a Senior Software Developer at Presagis where he has worked since 2003. Steve has eight years of experience in visual simulation, which started at NAVAIR Orlando where he was responsible for government acceptance of visual training systems. Steve is currently the Technical Lead for *Vega Prime Sensors* at Presagis. Steve holds a B.S. in Physics and a B.S. in Electrical Engineering from the *University of Central Florida* where he focused in Electromagnetism and 3D Computer Graphics.

REFERENCES

[1] Michael Wimmer and Daniel Scherzer, "Robust Shadow Mapping With Light-Space Perspective Shadow Maps," in *Shader X4 - Advanced Rendering Techniques*.: Charles River Media, 2006, pp. 313-330.

[2] Michal Valient, "Stable Rendering of Cascaded Shadow Maps," in *Shader X6 - Advanced Rendering Techniques*.: Charles River Media, 2008, pp. 213-238.

[3] Wolfgang Engel, "Cascaded Shadow Maps," in *Shader X5 - Advanced Rendering Techniques*.: Charles River Media, 2006,

pp. 197-206.

[4] Marc Stamminger and George Drettakis, "Perspective Shadow Maps," in *Proceedings of SIGGRAPH 2002*, 2002, pp. 557-562.

[5] Daniel Scherzer, "Robust Shadow Maps for Large Environments," in *Proceedings of the Central European Seminar on Computer Graphics 2005*, 2005.

[6] Michael Bunnell and Fabio Pellacini. (2004) developer.nivida.com. [Online]. http://http.developer.nvidia.com/GPUGems/gpugems_ch11.html

[7] Greg Humphreys Daniel Dunbar, "A spatial data structure for fast Poisson-disk sample generation," *ACM Transactions on Graphics (TOG), v.25 n.3*, July 2006.

[8] Peter-Pike J., and Michael F. Cohen Sloan, "Interactive Horizon Mapping," *11th Eurographics Workshop on Rendering*, pp. 281-286, 2000.

[9] James F. Blinn, "Simulation of Wrinkled Surfaces," *Computer Graphics, Vol. 12 (3)*, pp. 286-292, 1978.

[10] Daniel Scherzer, Stefan Jeschke, and Michael Wimmer,.

[11] William Donnelly and Andrew Lauritzen, "Variance Shadow Maps," *Proceedings of the 2006 Symposium on Interactive 3D graphics and Games*, pp. 161-165, 2006.

[12] Andrew Lauritzen, "Summed-Area Variance Shadow Maps," in *GPU Gems 3*. Boston, MA: Pearson Education, 2008, pp. 157-182.

[13] Kevin Myers, Randima Fernando, and Louis Bavoil, "Integrated Realistic Soft Shadows into your Game Engine," 2008, 2008.

[14] Tobias Martin and Tiow-Seng Tan, "Anti-aliasing and Conitnuity with Trapezoidal Shadow Maps," in *Eurographics*, Norrköping, Sweden, 2004, pp. 153-160.

[15] Fan Zhang, Hanqiu Sun, and Oskari Nyman, "Parallel-Split Shadow Maps for Large Scale Virtual Environments," 2007.

[16] Martin Mittring, Finding Next Gen - CryEngine 2, 2007.