Performance Characterization of Reactive Visual Systems

Subhagato Dutta, Abhishek Chugh, Robert Tamburo, Anthony Rowe, Srinivasa G. Narasimhan
Robotics Institute, Carnegie Mellon University

Abstract

We consider the class of projector-camera systems that adaptively image and illuminate a dynamic environment. Examples include adaptive front lighting in vehicles, dynamic stage performance lighting, adaptive dynamic range imaging and volumetric displays. A simulator is developed to explore the design space of such Reactive Visual Systems. Simulations are conducted to characterize system performance by analyzing the effects of end-to-end latency, jitter, and prediction algorithm complexity. Key operating points are identified where systems with simple prediction algorithms can outperform systems with more complex prediction algorithms. Based on the lessons learned from simulations, a low latency and low jitter, tight closed-loop reactive visual system is built. For the first time, we measure end-to-end latency, perform jitter analysis, investigate various prediction algorithms and their effect on system performance, compare our system's performance to previous work, and demonstrate dis-illumination of falling snow-like particles and photography of fast moving scenes.

1. Introduction

A Reactive Visual System consists of three primary components: an imaging component to sense a fast changing environment or scene, a processing component to analyze the captured imagery, and an actuation component to manipulate a light source (e.g., projector or display). This type of projector-camera system has numerous applications in computational photography (e.g., adaptive dynamic range imaging [10, 7]), lighting (e.g., adaptive automotive industrys [3, 14] or glare free presentation projectors [15]), theater/stage performance lighting, re-lighting a scene [11]), and displays (e.g., 3D displays on dynamic volumes [1] or moving mirrors [6], view and lighting reactive displays [9], anywhere-projection [5]). In all of these examples, one or more cameras and light sources are reactively and iteratively controlled to adapt to a changing environment.

The analysis of reactive visual systems is challenging for many reasons. The input and output signals consist of dense 2D or 3D arrays of pixels and are high dimensional. Algorithms to understand scenes from images are complex and inherently ambiguous, making it difficult to predict their performance in real-world situations with the dynamics from a physical environment. Like other tight closed-loop control systems [3, 14, 17], these factors are amplified by timing jitter resulting in few works that have systematically analyzed their behavior.

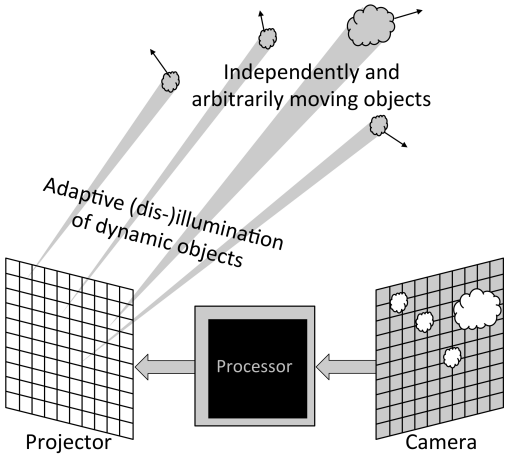

Figure 1. A binary reactive visual system illuminates or dis-illuminates objects detected from images captured by a camera.

In this work, we analyze reactive visual systems and characterize their real-time performance in terms of latency for given algorithms that have an inherent tradeoff between execution time and accuracy. Covering the factors affecting all reactive visual systems is too broad in scope, so we will limit our discussion to *Binary Reactive Visual Systems*. Here, the captured images (say, 8-bits per pixel) are binarized (to 1-bit per pixel) and processed to control a binary spatially varying illumination source, such as a DLP projector [2, 4]. We will assume that the scene (physical environment) consists of multiple objects that are moving independently at differing speeds and directions. The goal of a Binary Reactive Visual System would be to sense (image) the scene, detect objects and predict their future locations, and illuminate or dis-illuminate the objects (Fig. 1). Many of the applications mentioned above belong to this class of reactive visual systems.

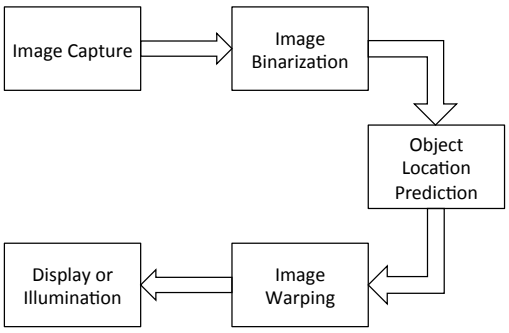

Figure 2. Block diagram of main architectural components of a Binary Reactive Visual System.

We present the design and evaluation of a low-latency, high-speed Binary Reactive Visual System under various application workloads. We characterize the system's Quality-of-Service (QoS) in terms of *error* and *light throughput* as a function of the system's response time. QoS in time-sensitive systems has been well studied by the real-time community [8, 12, 13], but this work focuses specifically on Binary Reactive Visual Systems. As a first step, we have developed a simulator that allows us to explore the design space of such a system, including various strategies for predicting object movement. We identify key operating points where decreasing the system latency with a less complex algorithm can outperform higher complexity algorithms that take more time to run to achieve the same goal. We see a similar performance trade-off where jitter (not just delay) in the control loop decreases performance.

The lessons learned from design exploration were used to guide the development of a low-latency Binary Reactive Visual System. The platform includes a high-speed camera and frame capture interface, and a time sensitive Linux-based processing pipeline connected to a custom hardware front-end that controls a DMD projector module. Our system has a much faster reaction time and less jitter than other work with reactive visual systems [3, 14]. For the first time, we measure and analyze end-to-end latency starting from camera exposure to scene illumination, perform jitter analysis, investigate various prediction algorithms and study their effects on system performance, compare the performance of our system with other systems, and demonstrate two adaptive lighting applications: dis-illumination of falling snow-like particles and photography of fast moving scenes.

2. Binary Reactive Visual Systems

The configuration of a Binary Reactive Visual System is shown in Fig. 2. The camera captures an image of a scene containing a set of objects with unknown locations and arbitrary motion. The processor then identifies the objects and records their current locations. This process is termed *binarization*, where, image pixels corresponding to the object locations are 'on' and are 'off' otherwise. The binary image, and perhaps a history of previous binary images, is then used to predict the object locations at the end of the system's response time. The predicted image is then warped to the coordinate frame of the projector, which in turn illuminates or dis-illuminates (depending on the application) the detected objects by turning pixels on or off, respectively. Note that while we focus on binary systems, pulse-width modulation coding can be used to project arbitrary grayscale values.

In order to compute the mapping between light rays from the objects observed by the camera and the corresponding light rays exiting the projector towards the objects, the three-dimensional positions of the objects must be computed. Such a mapping is commonly achieved by using multiple cameras in a stereoscopic configuration. Instead, for the purposes of this work, the camera and projector are optically co-located using a beamsplitter, which avoids parallax between the camera image and the projector image [14]. Co-location removes the need to compute the distances of objects in three dimensions and allows for all computations to be performed in the image space.

The latency (response time) of a Binary Reactive Visual System is defined as the time elapsed from the start of exposure by the camera and the completed illumination by the projector. In addition to measuring latency, we also measure the jitter or uncertainty in the end-to-end latency. In this work, systems (simulated and hardware prototype) with varying parameters are evaluated based on the following performance measures as a function of the system's latency and jitter. *Error* quantifies the incorrect illumination or dis-illumination of the scene and *light throughput* quantifies the total amount of light that illuminates the scene.

We apply this evaluation to two types of applications. The objective of the first application is to dynamically illuminate objects while the objective of the second application is to dis-illuminate objects. For the illumination application, we use rigid objects moving in a linear or projectile motion (ping pong balls moving and colliding). System error for this application is computed as the percentage of pixels that do not illuminate the objects and illuminate the background. For the dis-illumination application, we consider a large number of small objects moving chaotically, such as snowflakes during a storm. This application evaluates how well the system can act to avoid illuminating snowflakes to reduce their visibility to observers or a camera. In this case, error is computed as the percentage of pixels that incorrectly illuminate the objects. The error caused when incorrectly dis-illuminating the scene (road environment) is calculated separately as light throughput to assess the trade-off between reducing snowflake visibility and illuminating the road for the observer.

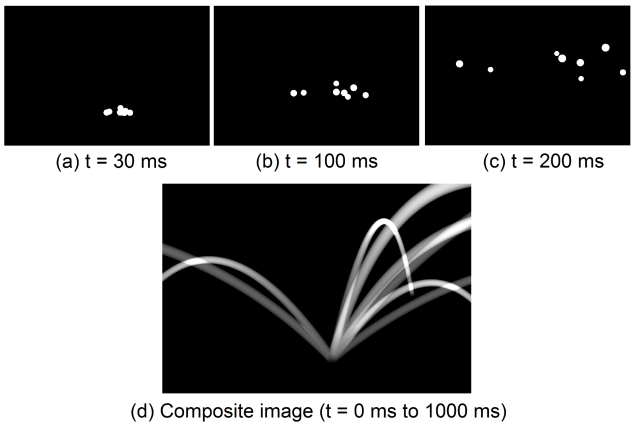

Figure 3. Images (a)-(c) are renderings of rigid objects (e.g., ping pong balls) quickly accelerating from rest (exploding) at different times. A long exposure (1 s) image of the event is shown in (d).

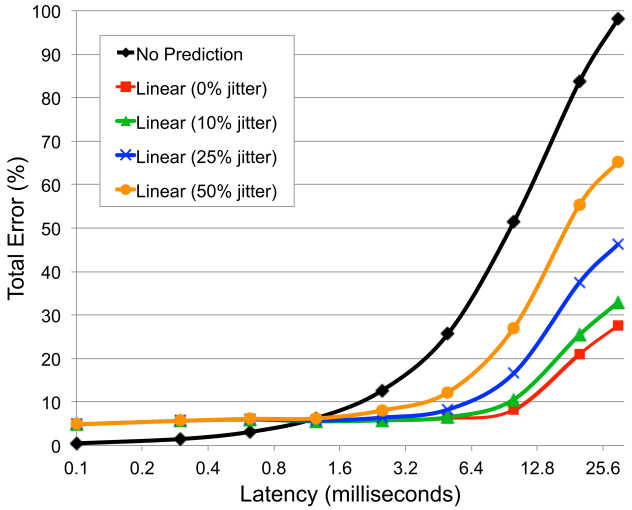

Figure 4. Total error while illuminating rigid objects as they explode for different system latencies. The black curve shows errors achieved with the no prediction algorithm. The other curves show error with linear prediction and latency uncertainty. Latency is plotted on a logarithmic scale.

3. Design Exploration

Our simulator models multiple types of objects and their motions in the scene to characterize the performance of generalized reactive visual systems with different latencies performing different algorithms. The scene is rendered using OpenGL and the rendered image is an input to the simulated reactive visual system, which analyzes the image and generates a corresponding illumination pattern. For the remainder of this section, we explore, for the first time, the performance of reactive visual systems for an illumination and a dis-illumination application.

3.1. Dynamic Lighting

Dynamic illumination of moving objects could be used, for example, to spotlight dancers on a stage or to study the trajectory of fast moving objects in a dynamic scene (an example of dynamic lighting can be observed in Figure 12). The actual system illuminates the scene with infrared light sources and observes the environment with a near-infrared, monochrome camera ensuring that the system's output (visible light) is not captured by the camera. The input image is thresholded to produce a binary image, which can then be displayed for immediate system response. We refer to this as the *no prediction algorithm*. Alternatively, prediction strategies can be employed to compensate for object motion - most likely adding to the system's latency though.

A simple algorithm for motion compensation assumes objects have an equal probability of moving in any direction. This algorithm is implemented by performing dilation with a kernel of fixed size on the binary image. A more intelligent approach would predict the future location of individual objects and produce the illumination pattern based on this information. In our implementation, blobs are detected from binary images and stored for the two most recent images. A metric based on blob size and position is used to find correspondences between newly detected blobs and previously detected blobs. The image velocity and position of objects in the two images are used to linearly extrapolate the position of objects at the time of illumination to produce an illumination pattern.

Through simulation, system performance was evaluated for these different prediction strategies. For performance evaluation, the fast motion of multiple rigid objects (perhaps, ping pong balls) being struck with a larger object (perhaps, a tennis ball) was simulated (Fig. 3). After receiving an image of the event, the simulated reactive visual system generates a binary illumination pattern. Based on the system's latency, an image of the scene is generated the instant that it is dynamically illuminated.

For quantitative evaluation, the image of the scene at the time of illumination and the illumination pattern are compared. There are two types of error; *Positive Error* and *Negative Error*. Positive Error is caused when a foreground object is not correctly illuminated and can be calculated as $\text{Positive Error} = \frac{\text{Object pixels not illuminated}}{\text{Total object pixels}}$. Negative Error is caused when the background is incorrectly illuminated and is defined as $\text{Negative Error} = \frac{\text{Background pixels illuminated}}{\text{Total object pixels}}$. The Performance of the system for this application is therefore measurable by $\text{Total Error} = \frac{\text{Positive Error + Negative Error}}{2}$.

As shown in Fig. 4, the no prediction algorithm has an error approaching 100% for latencies of typical video frame rate (30 Hz). Error can be reduced dramatically to around 10% by decreasing the system latency to 2 ms or less. At this latency, the no prediction algorithm is comparable to

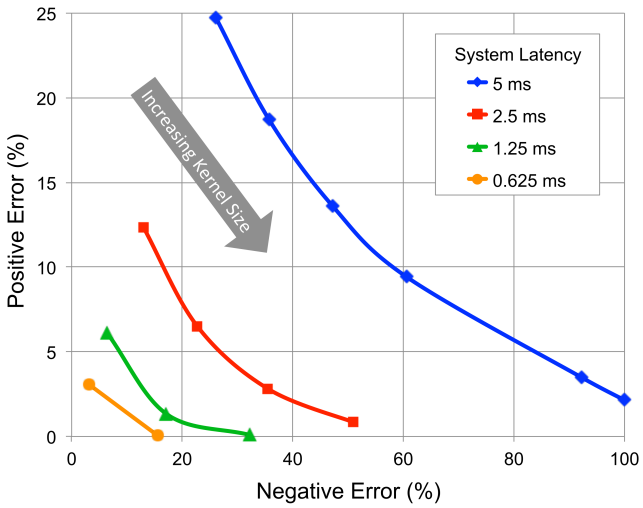

Figure 5. Increasing dilation kernel size improves proper illumination of the objects (positive error decreases) while increasing incorrect illumination of the background (negative error increases).

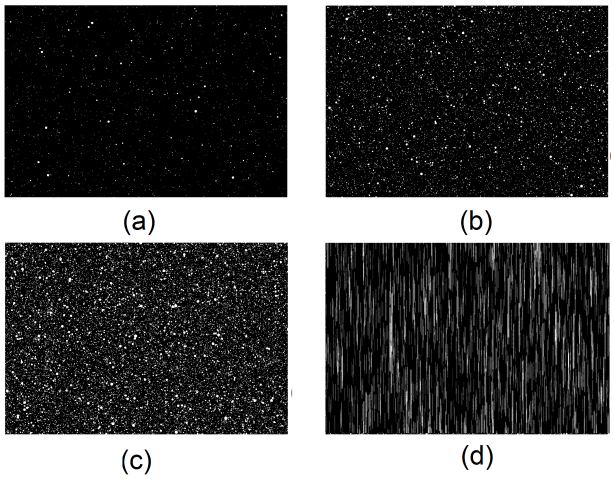

Figure 6. Images rendered by the simulator for snowflakes falling at (a) 0.5 mm/hr (b) 3mm/hr (c) 10 mm/hr. (d) is a long exposure (60 ms) capture of snow falling at 0.5 mm/hr.

the linear prediction algorithm, making the no prediction algorithm a viable strategy for fast reacting systems. For slower systems, the linear prediction algorithm results in much better performance. The downside of the linear prediction algorithm are sources of error that arise from incorrect correspondences, especially when the objects are near each other or overlap. Even at low latencies, this type of error is difficult to remove. At high image capture rates (e.g., 1 kHz exposure time), it is better to avoid linear prediction since typical objects will not move much in such short time periods. The contrary may be true for objects that move extremely fast. Since the prediction algorithm uses system latency to predict the position of objects at the time of illumination, it is imperative to study the effect of jitter. In the simulator, jitter is modeled as Gaussian noise in the latency. In Fig. 4, we also compare systems with different amounts of jitter (standard deviation in Gaussian distribution) and, although, jitter is an important consideration at higher latencies, its effect is minor at lower latencies.

The effect of using dilation with kernels of varying size to compensate for object motion is shown in Fig. 5. By increasing the size of the kernel, objects are better illuminated (decreasing positive error), but with the tradeoff that more of the background is incorrectly illuminated (increasing negative error). Compared to the no prediction and linear prediction algorithms, dilation results in more total error even with the smallest kernel size. Our simulations show that, for dynamic lighting of relatively fast moving objects, there is no advantage to using a prediction algorithm for low latency systems such as the one implemented in Section 4, which has a latency of 1 ms. Whereas, higher latency systems greatly benefit with linear prediction.

3.2. Improved Visibility in Snow

Retro-reflection from falling snowflakes distracts drivers from observing the road and makes driving during a snowstorm dangerous and stressful. Using a binary reactive visual system, snowflakes can be dis-illuminated for a very short period to improve visibility. We simulate snow falling at 3 mm/hr [3] and study the performance of different systems (Fig. 6). Note that the objects (snowflakes) in this application are small in size but large in number. The objective with this application is opposite of the dynamic lighting application, i.e., objects are dis-illuminated instead of illuminated. The two performance metrics to characterize system behavior for this application are $\text{Error} = \frac{\text{Snow pixels illuminated}}{\text{Total snow pixels}}$ and $\text{Light Throughput} = 1 - \frac{\text{Pixels dis-illuminated}}{\text{Total pixels}}$.

The average light throughput with each algorithm was 95.7% for no prediction, 84.9% for linear prediction, and 87.6% for dilation with a 1×5 kernel. As shown in Fig. 8, the system performs well with the no prediction algorithm at lower latencies. Performance degrades for systems with 0.2 ms latency as compared to both the dilation and linear prediction algorithms. For systems with a latency below 2 ms, dilation with a vertical kernel of size 1×5 performs much better than linear prediction while maintaining higher light throughput. Linear prediction performs best for systems with a latency greater than 2 ms.

Although, it is clear that dilation reduces error, the effect of dilation on light throughput is investigated. Simulations were performed to measure light throughput for achieving 0% error. Results show that systems with latency below 1.6 ms have more than 90% light throughput, but systems with longer latency have less than 85% light throughput. Additionally, the amount of dilation is also dependent on

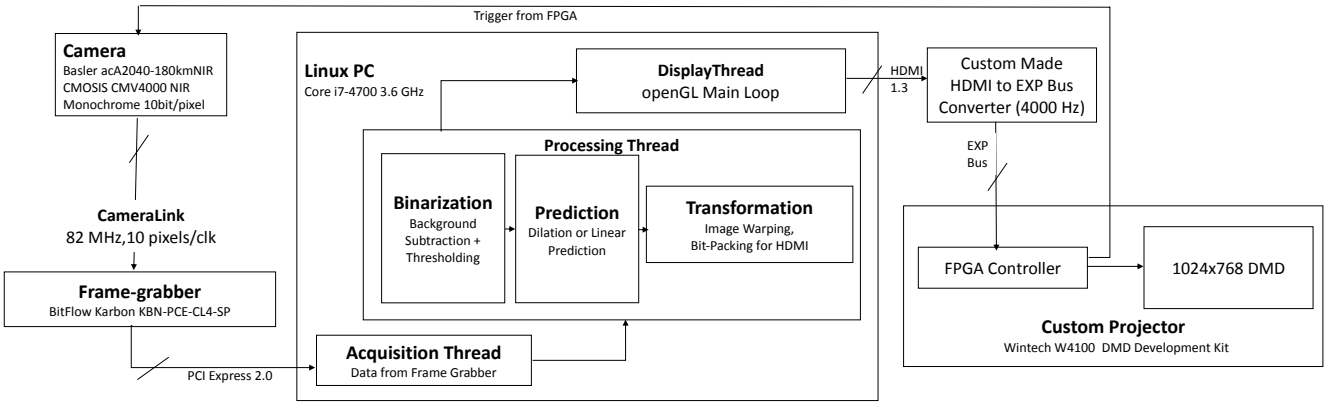

Figure 7. Diagram of our hardware implementation of a binary reactive visual system. Systems for other applications such as high dynamic range imaging and reconstruction will have similar implementations, but the same main components of sensing, processing, and projection.

the speed of the snowflakes. For example, if we wish to dis-illuminate snowflakes in a strong wind or on a fast moving vehicle, dilation would become inappropriate with even low latency systems.

We performed simulations to understand the effect of snowfall rate for our implemented binary reactive visual system described in Section 4. Latencies measured from our system were used to compare system error with different algorithms. Changes in snowfall rate do not change the latency of the systems with the no prediction or dilation algorithm. Thus, their error rates do not change either with the no prediction algorithm resulting in 70% error and the dilation algorithm resulting in 2% error. On the other hand, the linear prediction algorithm depends on the number of objects in the image, and therefore, causes an increase in computation time resulting in more error for higher snowfall rates. Error is approximately 15% for a light snowfall (0.3 mm/hr) and increases to 35% for blizzard-like snowfall rates (10 mm/hr).

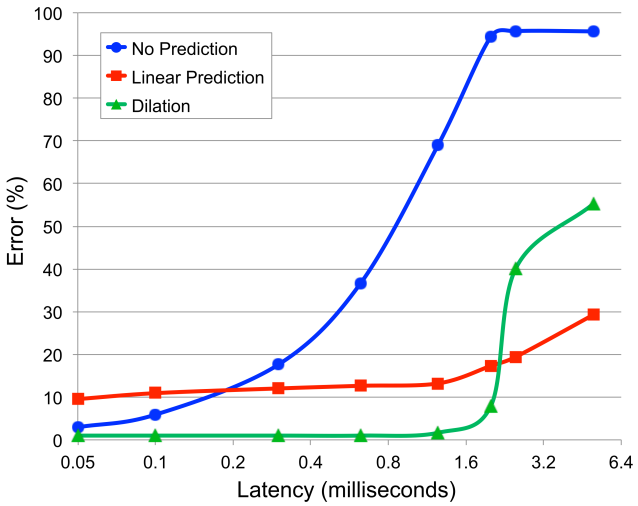

Figure 8. The effect of latency on system performance with various algorithms. On average, the no prediction algorithm achieves 95.7% light throughput, while linear prediction and dilation (1×5 kernel) achieve 84.9% and 87.6% light throughput, respectively. Latency is plotted on a logarithmic scale.

4. System Implementation

A Binary Reactive Visual System was built using a camera for sensing, a computer for processing, and a custom-built high-speed projector for illumination. A diagram of our hardware implementation is shown in Fig. 7. Our system is similar to that of [14] except with significant improvements to reduce system latency by at least 40% with minimized jitter.

4.1. Imaging

A computer with an Intel Core i7 3.6 GHz CPU interfaces between the camera and projector and processes the images. A monochrome, near-infrared camera (Basler) with a global shutter is used to capture images. The camera is capable of low latency and high data throughput because of its pipelined pixel architecture and extended CameraLink interface. The camera and projector are co-located following the procedure described in [14]. Camera exposure was synchronized to the projector to quantize system jitter in a deterministic manner. Synchronization also permits measuring system latency as explained in Section 5.1. A PCI express 2.0 frame grabber (Bitflow) is used to transfer data, without buffering, into DDR3 memory.

Software Architecture: The system's functionality is performed with multi-threaded software. The *acquisition thread* retrieves images from memory and copies them into an image buffer shared with a *processing thread*, where the image is processed (e.g., binarization, prediction, transformation, etc). After an illumination pattern is computed, it is stored in an image buffer shared with a *display thread*,

which uses OpenGL to transfer the illumination pattern to the projector via the custom board described in Section 4.2. Linux Kernel 3.8 with RT Patch (3.8.14.15.rt+) was used to ensure deterministic scheduling performance and hyper-threading was disabled to eliminate cache-coherence issues resulting in minimized jitter.

Image Processing: The processing component consists of image binarization, object location prediction, image warping, and bit-packing. Quantitative evaluation of system performance is discussed in Section 5.2.

Binarization: Objects are segmented as follows:

$$B(x,y) = \begin{cases} 1 & \text{if } |I(x,y) - I_{bg}(x,y)| > \alpha \\ 0 & \text{otherwise,} \end{cases}$$

where α is an intensity threshold and $I_{bg}(x,y)$ is a pre-computed background image. The 8-bit image is subtracted from the background image and then the resulting image is thresholded. Since both of these are per-pixel operations, they were combined and written in AVX2 optimized SIMD vectorized code to increase computational efficiency.

Prediction: As discussed in Section 3.1, two approaches for estimating future locations of objects were investigated.

Image Warping: Image distortion caused by the camera lens and the transformation between the camera and projector planes are stored in a look-up table. The look-up table maps pixels from the projector's coordinate system to the camera's coordinate system to guarantee that every pixel has a correspondence. Access of the look-up table was written in AVX2 to increase the speed of memory operations. The result of the transformation is a binary image containing the illumination pattern sized to the projector resolution.

4.2. Projection

A high-speed projector (4,000 Hz) was built to achieve high-speed illumination. The projector consists of a DMD development board (WinTech W4100) [16] and a 4,800 Lumen light source. One of the bottlenecks identified in [14] was the time to transfer the illumination pattern to the projector over USB 2.0 ($0.6-0.7$ ms). To avoid this bottleneck, we designed a custom module to interface the PC using HDMI directly to the EXP bus interface on the DMD board. The HDMI board uses an integrated circuit (TI TFP401a) to parallelize the HDMI signals to a 28-bit video signal with a pixel clock rate of 177 MHz.

The binary illumination pattern is transferred over HDMI as a 24-bit image. Since the illumination pattern is binary, it is packed into a low-resolution, 24-bit image. The bit-packed image is transferred to the projector at a regular interval of 250 μs. The FPGA of the DMD board was programmed to expand the bit-packed image to the full resolution of the projector (1024×768). The board can transfer a full resolution image in 250 μs allowing a display rate of 4,000 Hz, which is 3 times faster than [14].

System	Our system		[14]	[3]
Resolution	960×340	960×170	1000×340	244×120
Exposure	$100\mu s$	$100\mu s$	$100\mu s$	$5000\mu s$
Image Transfer	$925\mu s$	$520\mu s$	$925\mu s$	$4200\mu s$
Acquisition	$20\mu s$	$10\mu s$	$\geq 300\mu s^{\ddagger}$	$4100\mu s^{\ddagger}$
Binarization	$20\mu s$	$12\mu s$		
Warping	$160\mu s$	$80\mu s$		
Bit-Packing†	$20\mu s$	$10\mu s$		
Display	$250\mu s$	$250\mu s$	$760\mu s$	$4200\mu s$
Total	$1495\mu s$	$992\mu s$	$\geq 2085\mu s$	$17500\mu s$

† Bit-packing not performed in [14] and [3]
‡ Processing time increases as the number of detected objects increases

Table 1. Latency of our system with two different image resolutions compared to other reported systems. The latency of our system is independent of the number of objects detected and can be used to emulate other systems by means of a software delay.

5. System Response Time

5.1. Measuring End-to-End Latency

To measure the reaction time of the system, the FPGA of the DMD board was programmed to output a trigger pulse when a new illumination pattern was received. Latency was measured with a logic analyzer (Saleae Logic 8) as the time between successive trigger pulses. Because the HDMI clock transfers data asynchronously at 4,000 Hz, jitter is quantized at 250 μs intervals. For example, if there is a delay in the system, (e.g., because of image analysis), the image will have to be displayed after a delay of 250 μs.

In addition to measuring round-trip latency of our system, we also measured the time to execute various processes. The latency of the system depends on various factors including image size (transfer and processing), algorithm complexity, and system uncertainty. Latency of our system for two resolutions (only binarization, warping, and bit-packing performed) are detailed in Table 1. The latency of similar systems [3, 14] are shown in the same table.

5.2. Effect of Prediction Algorithms on Latency

To compare system performance with different prediction algorithms, images generated from the simulator (Section 3) were used to guarantee data variability and repeatability for different trials. Latency was measured using the same method in Section 5.1. Data were collected for a minute for each trial and summary statistics were computed.

The image resolution for which the camera field of view covers the projector's field of view is 960×680. To decrease the system's reaction time, camera images were vertically decimated by a factor of both 2 and 4 yielding image resolutions of 960×340 and 960×170, respectively. Kernel sizes are reported as their effective size at full image resolution. For example, a kernel of radius 9 in the full resolution image has a radius of 5 for decimation by a factor of 2 and a radius of 3 for decimation by a factor of 4.

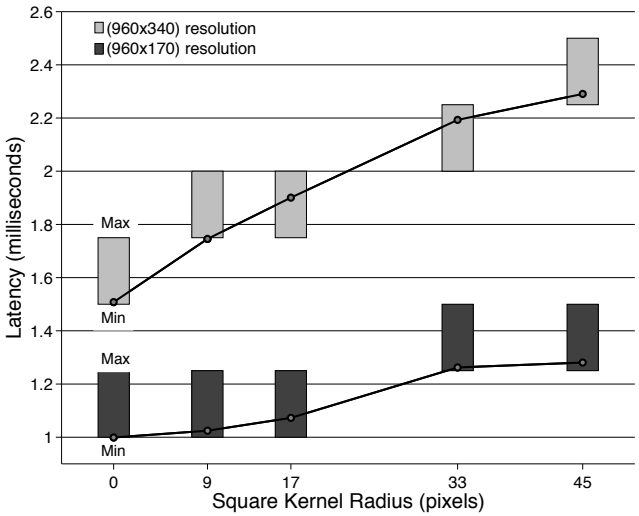

Figure 9. Increasing the size of the dilation kernel increases the latency of the system. Kernel radius is reported as the effective size at a full image resolution of 960×680. Circles indicate average latency, which are quantized by the 250 μs jitter of our system.

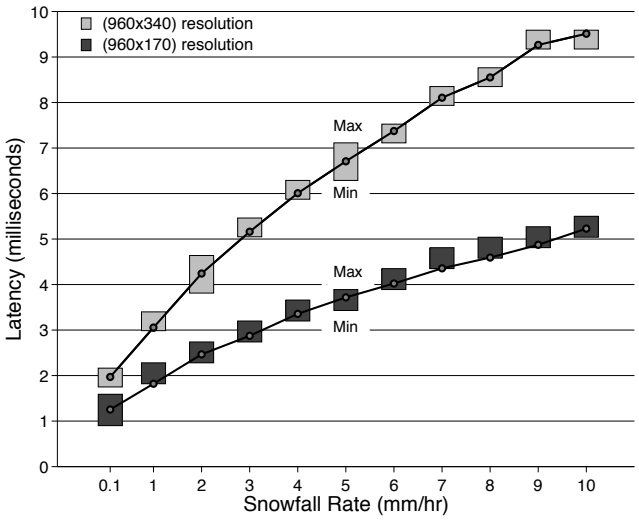

Figure 10. The latency of a system with linear prediction steadily increases with snowfall rate because our algorithm requires more computations to process additional detected snowflakes. Recall, that our system has 250 μs jitter, which is demonstrated by the average latency as indicated by circles.

For objects moving in an arbitrary direction (like the exploding ping pong balls), dilation with an isotropic kernel is sufficient. However, for particles moving in a generally known direction (falling snow), it is a better strategy to dilate in the known direction. As shown in Fig. 9, increasing the kernel size significantly increases latency. Latency is reported as an average, minimum, and maximum value. Latency measured for different snowfall rates ranging from a light flurry to a blizzard are shown in Fig. 10. Latency increases because the algorithm depends on the number of objects detected, which increases with the snowfall rate.

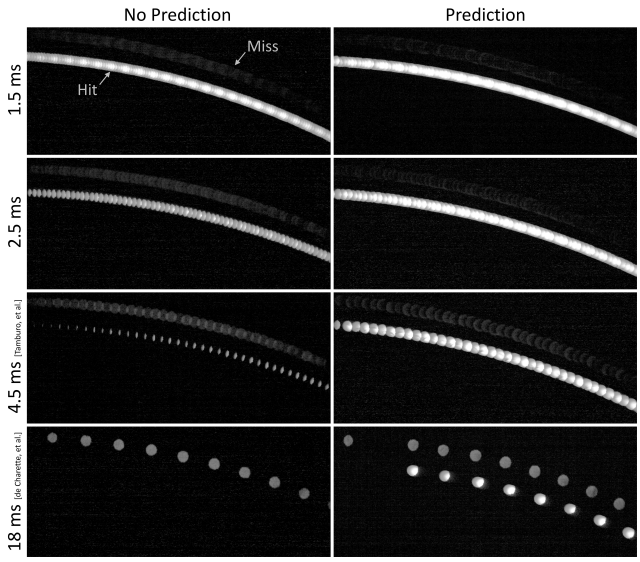

Figure 11. A ping pong ball with projectile motion was illuminated by the system while capturing a long exposure image. Bright streaks (hit) show correct illumination of the ball while dark silhouettes (miss) in the background show the error caused by a slow reaction of the system. Results from our system are in the top row. Results using latencies from [14] and [3] with our system are shown in the third and fourth rows, respectively.

6. Visual Evaluation

The effect of latency, prediction, and dilation on system accuracy was qualitatively evaluated by visual observation. First, system accuracy was evaluated for different latencies with and without performing prediction. In this evaluation, the system illuminated a rigid object (ping pong ball) with projectile motion traveling 15 mph across the system's field of view at a distance of 15 feet in a dark room. The balls were sensed in the dark by using infrared LEDs to illuminate the scene. Long exposure (1 second) images were taken with a DSLR camera to capture the balls being illuminated by the system, which appear as a streak in the image foreground. High brightness and smoothness of the streak indicates high accuracy of the system. A second streak in the background captures the system's error, i.e., the system's reaction time is too slow to illuminate the ball and instead illuminates the backdrop. Our system was compared to previous systems [3, 14] with and without prediction by adding their latency to our system with a software delay. From Fig. 11, it is clear that prediction is unnecessary for a system with a response time of 1.5 ms and is beneficial in systems with a slower response time.

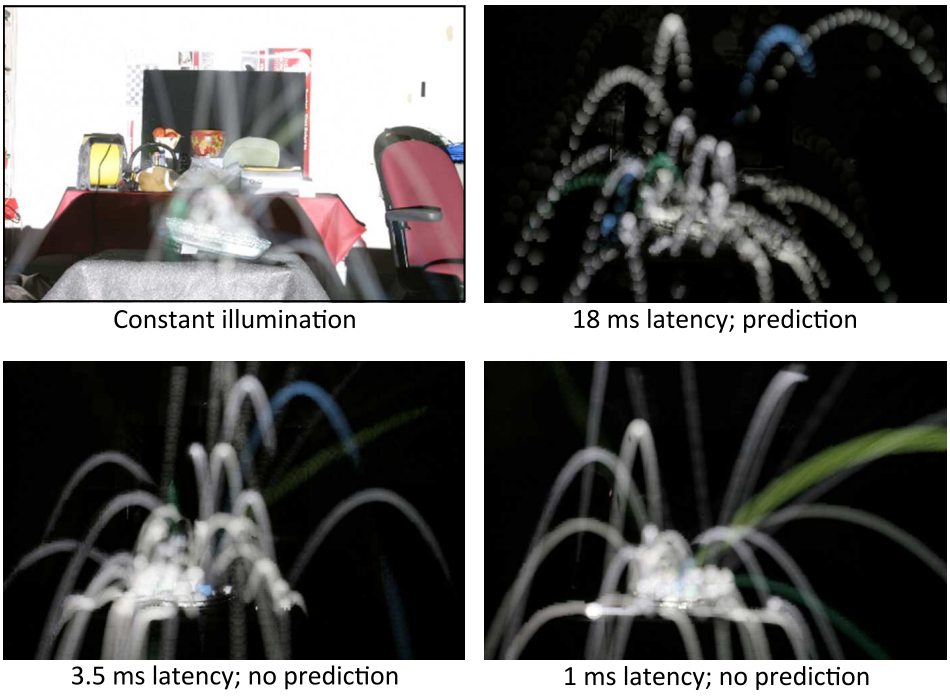

Figure 12. Visual evaluation of system performance for rigid objects with projectile motion. A tennis ball is thrown into a bowl of ping pong balls. The first image shows constant illumination. In subsequent images, ping pong balls are adaptively illuminated.

Figure 13. Visual evaluation of system performance for small objects moving chaotically. Artificial snowflakes are dropped. Still frames from a video captured at 30 fps are shown. In (a), snowflakes are constantly illuminated. In (b)-(d), snowflakes are adaptively dis-illuminated while increasing the size of the dilation kernel resulting in higher system accuracy and lower snowflake visibility.

We evaluated the accuracy of systems for multiple objects quickly accelerating from rest. A tennis ball was thrown into a bowl of ping pong balls while capturing a long exposure (1 second) image with a DSLR camera. Results are shown in Fig. 12. In the first image, the entire scene is illuminated washing out the event. As evident by the smoothness of motion trails, the second image shows that a slow system with linear prediction performs worse than faster systems without any prediction (third and fourth images).

The effect of increasing the size of the dilation kernel is shown in Fig. 13. Artificial snowflakes (styrofoam beads) were dropped while being dis-illuminated by the system. As the kernel size is increased, accuracy increases, and consequently, visibility of the snowflakes decreases. Larger kernel sizes decrease the visibility of snowflakes while increasing false positive error - evident by larger dark streaks visible on the road. The requirements of this trade-off space will vary between applications. Results of these visual evaluation experiments confirm those of the simulations.

7. Conclusions

We present a first attempt at analyzing systems that adaptively image and illuminate a dynamic environment using factors that guide the design of general reactive visual systems. The design space includes factors such as system response time and jitter and choice of algorithm and its complexity for adaptation. Our analysis provides simple rules for performance-driven implementation in multiple applications and answers design questions like: when does prediction help? what should be the latencies in a particular application? how are the latencies and prediction algorithm complexity coupled? what are the bottlenecks in the system? Our current hardware implementation is flexible enough to adapt to the performance requirements of two very different applications - dynamic lighting and dis-illumination of falling snowflake-like particles. While we have focused our efforts on binary reactive visual systems containing one camera and one projector, future work will include analysis of general imaging and illumination configurations necessary for several applications in this space.

8. Acknowledgments

This research was supported in parts by ONR grant N00014-14-1-0595, NSF Grant CNS-1446601, a US DOT RITA University Transportation Center TSET Grant, and a Ford Motor Company gift. The authors thank Feng Yang for implementing the linear prediction algorithm, Zisimos Economou for helping design the HDMI board, Tejal Kudav for assisting with the VHDL design for FPGA and Vinay Palakkode for assisting with code optimization.

References

[1] P. C. Barnum, S. G. Narasimhan, and T. Kanade. A multi-layered display with water drops. *ACM Trans. Graph.*, 29(4), July 2010.

[2] J. S. D. Dudley, W. M. Duncan. Emerging dmd applications. *Proc of SPIE MOEMS Display and Imaging Systems*, 4985, 2003.

[3] R. de Charette, R. Tamburo, P. Barnum, A. Rowe, T. Kanade, and S. Narasimhan. Fast reactive control for illumination through rain and snow. pages 1–10, 2012.

[4] M. R. Douglass. DMD reliability: A MEMS success story. *Proc of SPIE Reliability, Testing, and Characterization of MEMS/MOEMS II*, 4980, 2003.

[5] A. Grundhofer and O. Bimber. Real-time adaptive radiometric compensation. *Visualization and Computer Graphics, IEEE Transactions on*, 14(1), Jan 2008.

[6] A. Jones, I. McDowall, H. Yamada, M. Bolas, and P. Debevec. Rendering for an interactive 360°; light field display. *ACM Trans. Graph.*, 26(3), July 2007.

[7] P.-J. Lapray, B. Heyrman, and D. Ginhac. HDR-ARtiSt: an adaptive real-time smart camera for high dynamic range imaging. *Journal of Real-Time Image Processing*, January 2014.

[8] J. W. S. Liu, K.-J. Lin, R. Bettati, D. Hull, and A. Yu. Use of imprecise computation to enhance dependability of real-time systems. In *Foundations of Dependable Computing: Paradigms*, pages 157–182. Kluwer Academic Publishers, 1994.

[9] S. K. Nayar, P. N. Belhumeur, and T. E. Boult. Lighting sensitive display. *ACM Trans. Graph.*, 23(4), Oct. 2004.

[10] S. K. Nayar and V. Branzoi. Adaptive dynamic range imaging: Optical control of pixel exposures over space and time. In *Proceedings of the Ninth IEEE International Conference on Computer Vision - Volume 2*, ICCV '03, 2003.

[11] M. O'Toole and K. N. Kutulakos. Optical computing for fast light transport analysis. *ACM Trans. Graph.*, 29(6), 2010.

[12] R. Rajkumar, C. Lee, J. Lehoczky, and D. Siewiorek. A resource allocation model for QoS management. In *Real-Time Systems Symposium, 1997. Proceedings., The 18th IEEE*, pages 298–307, Dec 1997.

[13] D. Seto, J. Lehoczky, L. Sha, and K. G. Shin. On task schedulability in real-time control systems, 1996.

[14] R. Tamburo, E. Nurvitadhi, A. Chugh, M. Chen, A. Rowe, T. Kanade, and S. G. Narasimhan. Programmable automotive headlights. In *Computer Vision - ECCV 2014*, volume 8692 of *Lecture Notes in Computer Science*, 2014.

[15] D. S. Tan and R. Pausch. Pre-emptive shadows: eliminating the blinding light from projectors. In *CHI '02 Extended Abstracts on Human Factors in Computing Systems*, pages 682–683, 2002.

[16] WinTech. DLP discovery W4100 kit. 2014. `http://wintechdigital.com/Upfiles/W4100_Brochure.pdf`.

[17] F. Zheng, T. Whitted, A. Lastra, P. Lincoln, A. State, A. Maimone, and H. Fuchs. Minimizing latency for augmented reality displays: Frames considered harmful. *IEEE International Symposium on Mixed and Augmented Reality*, 2014.