**Weixin Wu**
**Yujie Dong**
**Adam Hoover***
Department of Electrical and
Computer Engineering
Clemson University
Clemson, SC, 29634-0915

# Measuring Digital System Latency from Sensing to Actuation at Continuous 1-ms Resolution

## Abstract

This paper describes a new method for measuring the end-to-end latency between sensing and actuation in a digital computing system. Compared to previous works, which generally measured the latency at 10–33-ms intervals or at discrete events separated by hundreds of ms, our new method measures the latency continuously at 1-ms resolution. This allows for the observation of variations in latency over sub 1-s periods, instead of relying upon averages of measurements. We have applied our method to two systems, the first using a camera for sensing and an LCD monitor for actuation, and the second using an orientation sensor for sensing and a motor for actuation. Our results show two interesting findings. First, a cyclical variation in latency can be seen based upon the relative rates of the sensor and actuator clocks and buffer times; for the components we tested, the variation was in the range of 15–50 Hz with a magnitude of 10–20 ms. Second, orientation sensor error can look like a variation in latency; for the sensor we tested, the variation was in the range of 0.5–1.0 Hz with a magnitude of 20–100 ms. Both of these findings have implications for robotics and virtual reality systems. In particular, it is possible that the variation in apparent latency caused by orientation sensor error may have some relation to simulator sickness.

## 1 Introduction

This paper considers the problem of measuring the latency in a digital system from sensing to actuation. We are motivated by sensors and actuators that operate using their own clocks, such as digital cameras, orientation sensors, displays, and motors. Figure 1 shows a typical configuration. The system latency, also called end-to-end latency, is defined as the time it takes for a real-world event to be sensed, processed, and actuated (e.g., displayed). Latency is commonly in the range of tens to hundreds of ms, and thus while difficult to measure, is in the range that affects control problems and human users. In virtual reality systems, latency has been shown to confound pointing and object motion tasks (Teather, Pavlovych, Stuerzlinger, & MacKenzie, 2009), catching tasks (Lippi, Avizzano, Mottet, & Ruffaldi, 2010), and ball bouncing tasks (Morice, Siegler, & Bardy, 2008). In robotics, latency has an impact on teleoperation (Ware & Balakrishnan, 1994) and vision-based

Presence, *Vol. 22, No. 1, Winter 2013,* 20–35
doi:10.1162/PRES_a_00131


*Correspondence to ahoover@clemson.edu.

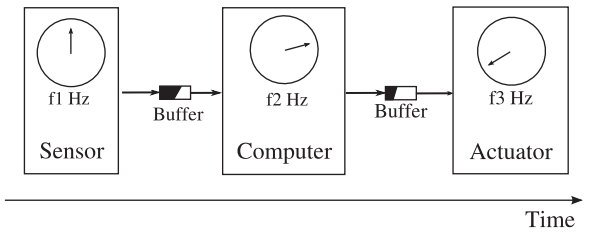


**Figure 1.** *System latency is nonconstant due to components using independent clocks and the variable delays in buffers connecting components.*

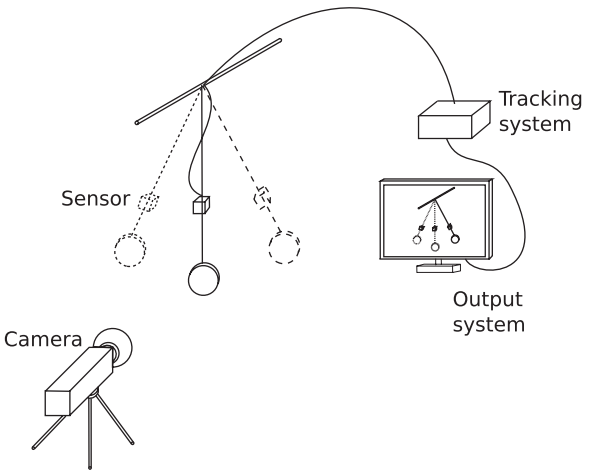


**Figure 2.** *Continuous approach to measuring latency.*

control (Liu, Hoover, & Walker, 2004). Its effect has also been studied in immersive video conferencing (Roberts, Duckworth, Moore, Wolff, & O'Hare, 2009).

It is possible to measure latency internally using the computer in the system, by time-stamping when a sensor input is received, and by time-stamping when an actuation output is commanded. However, these time stamps do not include the time that data may spend in buffers, nor do they include the time that may be spent by the sensor acquiring the data or by the actuator outputting the data. Therefore, it is preferable to use external instrumentation to measure the latency by observing the entire system. Two general approaches have been taken to this problem, one that uses a camera to continuously observe the system, and one that uses event-driven instrumentation such as photodiodes to more precisely measure discrete events.

Figure 2 illustrates a typical experimental setup for the camera-based continuous approach. A sensor (usually a component of a 3-DOF or 6-DOF tracking system) is placed on a pendulum or other moving apparatus. A computer receives the tracking data from the sensor and displays it on a monitor. An external camera views both the live motion and the displayed motion, comparing them to determine the latency. Bryson and Fisher (1990) pioneered this approach by comparing human hand movement of a tracked device against the displayed motion; latency was calculated as the number of camera frames between when hand motion started and when the displayed motion started. He, Liu, Pape, Dawe, and Sandin (2000) used a similar approach with a grid visible behind the tracked object so that multiple points could be used for measurements. Liang, Shaw, and Green (1991) were the first to suggest using a pendulum to move the sensor so that the actual motion was known; latency was calculated as the time between when the camera frames showed the pendulum at its lowest point versus when the tracked data showed the pendulum at its lowest point. Ware and Balakrishan (1994) followed the same approach but used a motor pulling an object back and forth linearly so that the tracked object velocity was constant. Steed (2008) also used a pendulum but fit sinusoidal curves to both the live and displayed data, calculating the relative phase shift between the curves, so that a more precise estimate of latency could be made. In one experiment, Morice et al. (2008) used a racket waved in an oscillatory motion by a human; latency was measured by finding the time difference between frames containing the maxima of the motion in the live and displayed data. Swindells, Dill, and Booth (2000) used a turntable; latency was measured using the angular difference between the live and displayed data. Instead of using a camera to observe the system, Adelstein, Johnston, and Ellis (1996) moved the tracked object using a robot arm; latency was measured by comparing the angle of the motor encoder of the arm against the angle of the tracking sensor. All of these methods are capable of measuring latency continuously, but the reported experiments were limited by the sampling rates of the

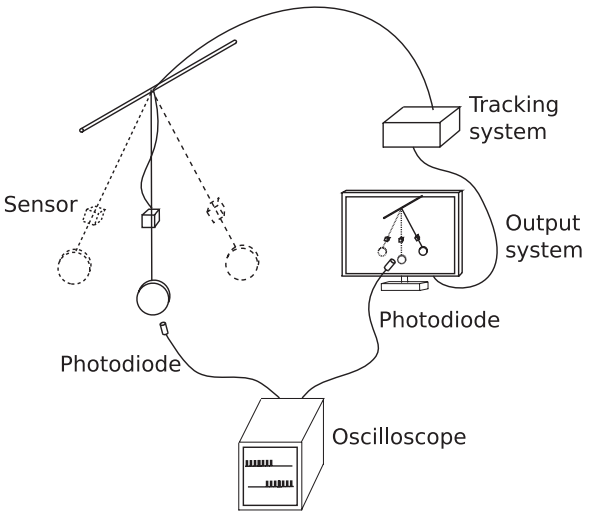


**Figure 3.** *Discrete event approach to measuring latency.*

cameras or instrumentation (25–50 Hz). Because the measured latency is in the range of 30–150 ms, multiple measurements were averaged or data were interpolated in between measurements.

Figure 3 illustrates a typical experimental setup for the discrete event-based approach to measuring latency. In this approach, a photodiode is placed at a fixed position so that when the tracked object passes that point, a signal is registered on an oscilloscope. A second photodiode is placed at the corresponding fixed position for the displayed output. This approach was pioneered by Mine (1993), who used several variations of the idea (with different instrumentation) to estimate latency in different parts of the systems of interest. The method has been used by other researchers with similar results (Akatsuka & Bekey, 2006; Morice et al., 2008; Olano, Cohen, Mine, & Bishop, 1995; Teather et al., 2009). While this approach allows for more precise measurements of latency (because the instrumentation is not limited to the sampling rate of a camera), measurements can only be made at the discrete times when the tracked object passes the reference point. This approach does not account for variations in latency that may happen at different positions of the sensor and actuator; for example, actuation in a display monitor takes place at different times across the screen as the image is redrawn. All of the experiments reported using this approach calculated average latencies, and did not describe latency variation over time.

Miller and Bishop (2002) describe a method to calculate latency continuously using 1D CCD arrays operated at 150 Hz. However, they average their calculations from these measurements in such a way that latency is only calculated at 10 Hz. DiLuca (2010) describes a method using photodiodes moved sinusoidally in front of a sensed and displayed gradient intensity. The variations in intensity are correlated to calculate the average latency. In their experiments they used a stereo input of a laptop computer, presumably operating at a 44-KHz frequency (this detail was not provided in the paper). However, the measurements were high-pass filtered and then correlated to find an average. Although their method potentially could be used to study continuous variations in latency, they did not pursue this idea.

Figure 4 summarizes the problem with all previous works. All the camera-based methods took measurements at regular intervals but computed an average latency as the output. All the discrete-event methods took more precise individual measurements at irregular intervals, but still computed an average latency as the output. The implicit assumption of all these works is that latency can be described by a random distribution (e.g., normal or uniform). We propose to take continuous measurements of latency in order to see whether nonrandom patterns are observable. For example, Figure 4(c) shows continuous measurements of the same underlying signal as Figure 4(a–b), where a sinusoidal variation in latency can be observed. Previous works have discussed the idea that system latency is not a constant (Adelstein et al., 1996; DiLuca, 2010). However, this paper is the first to show how to continuously measure the latency at a rate sufficient to see how it changes over a period less than 1 s.

More precise measurements and a better understanding of latency have potential applications in robotics and virtual reality. For example, a robotics gripping application typically builds the gripper large enough to compensate for the distribution of potential latency (Liu et al., 2004). This assumes that the latency follows a random distribution. In contrast, if the latency could

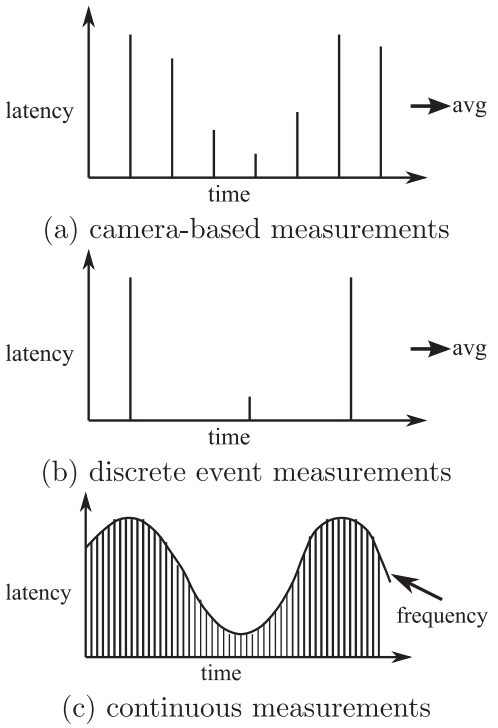


**Figure 4.** *All previous works compute an average latency using (a) regular but sparse measurements or (b) precise event driven but still sparse measurements. In contrast, we propose to measure latency continuously (c) so that frequencies in latency variations can be studied.*

be modeled as a sinusoidally varying function, control algorithms could be designed to compensate for and thereby reduce the size of the gripper. In virtual reality, simulator sickness is a phenomenon where users of head-mounted displays experience nausea or sick feelings while using these systems. Latency has long been studied as a possible cause, but previous works have only studied the effect on sickness of the average latency (Moss et al., 2011).

## 2 Methods

Our approach is similar to other continuous methods discussed in the introduction. Figure 5 illustrates our methodology. The system being measured is configured in such a way that the actuator outputs the same property (e.g., position, angle, etc.) sensed by

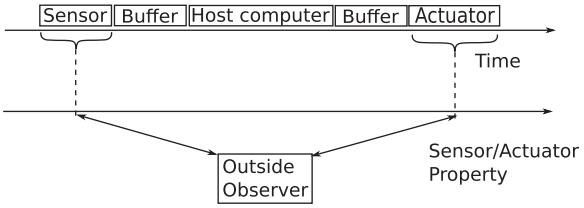


**Figure 5.** *Latency is measured indirectly via the property (e.g., position, orientation) being sensed and actuated.*

the sensor. The outside observer (we use this term to differentiate it from any camera used as a sensor in a system being measured) is a high-speed camera capable of observing the property. Latency is measured by calculating the number of high-speed camera frames between when the sensed property matches the actuated property. We performed experiments on two systems using this approach. We first describe our outside observer, then describe each system in detail.

### 2.1 Outside Observer

For an outside observer, we used a Fastec Trouble Shooter 1000 high-speed camera. It can capture video at 480 × 640 resolution at up to 1,000 Hz for 4.4 s. We have found that at this speed, the scene being imaged must be very brightly illuminated, because the exposure interval is so small. Steed (2008) reports trying to use a 500-Hz high-speed camera and having the same problem. To compensate for this, we use external spotlights mounted around the systems to increase the ambient illumination. Because the spotlights operate at 60 Hz synchronous to the power source, they cause an oscillation in intensity in the high-speed camera frames. To address this problem, histogram equalization and adaptive thresholding (discussed later) are used during the processing of the images.

### 2.2 System 1

Our first system uses a camera for sensing and a computer monitor for actuation. The camera is a Sony XC-75 (www.subtechnique.com/sony/PDFs/xc-7573e.pdf), an interlaced camera operating at 30 Hz.

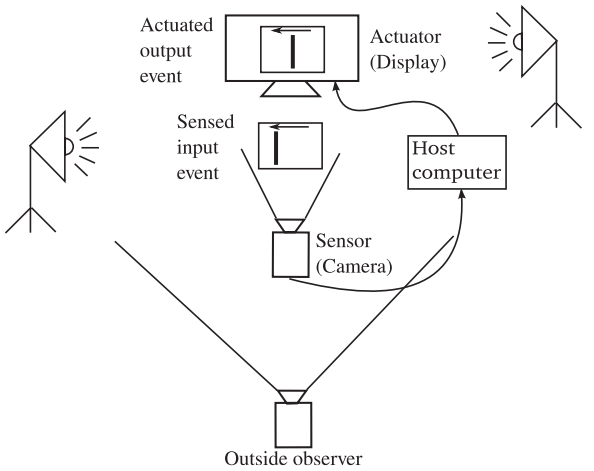


**Figure 6.** *System 1: camera to monitor.*

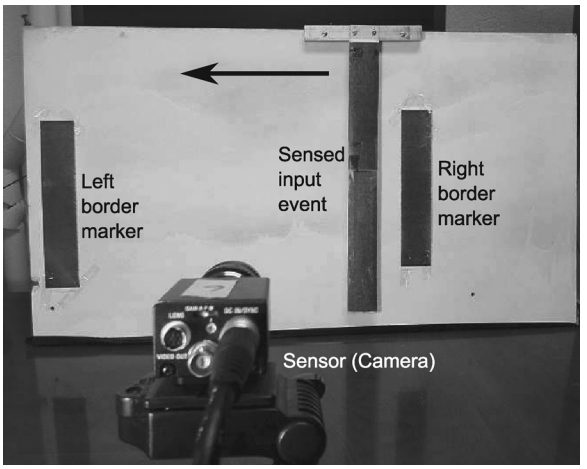


**Figure 7.** *Camera-to-monitor experiment apparatus.*

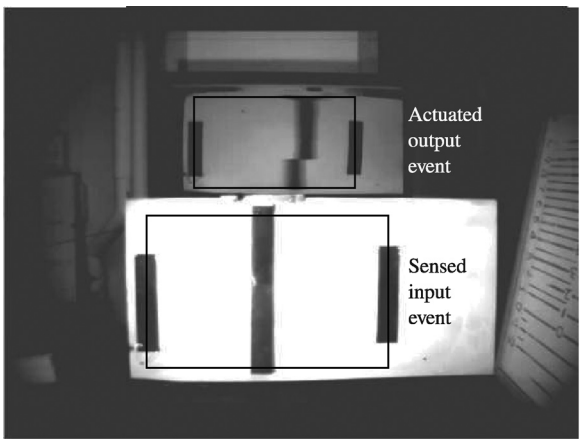


**Figure 8.** *Camera-to-monitor system as seen by the outside observer.*

The computer has an Intel Core Duo 2.8-GHz processor, 4-GB main memory, and a 500-GB hard drive. The frame grabber is a Matrox Meteor-II Multi Channel (http://www.matrox.com/imaging/en/products/frame_grabbers/). The graphics card is an nVidia GeForce 9500 GT (http://www.nvidia.com/object/product_geforce_9500gt_us.html/). The operating system is Windows XP Professional SP2. The monitor is an Acer AL2216W operating at 60 Hz.

Figure 6 shows a diagram of the experimental setup. The sensor is aimed at a specially constructed apparatus, labeled the sensed input event in Figure 7. The images captured by the sensor are digitized in the computer and forwarded to the actuator, an LCD display. The computer does not change the content of the sensed images, so that the output image matches the sensed input image, but after some latency. The outside observer sits behind the system with its field-of-view positioned so that it can see the sensed input event and the actuated output event simultaneously. By comparing these and matching when they show the same content, we can indirectly measure the latency.

Figure 7 shows a picture of the apparatus. It consists of a background piece of wood painted white, with a wooden bar painted black in front of it. The bar is fixed vertically so that it can only move back and forth horizontally. The purpose of the apparatus is to create a motion that is easily discernible in the high-speed captured images. This facilitates image processing of the frames captured by the outside observer, in order to help automate the measurement process. During an experiment, the black vertical bar of the apparatus is manually moved horizontally.

An example raw frame captured by the outside observer is shown in Figure 8. The sensed input event is visible in the lower section and the actuated output event is visible in the upper section. The latency can be seen by the different positions of the bar. The tear in the bar in the actuated output is due to the redrawing of the image in the LCD monitor. The redrawing happens top-to-bottom, so that at any given time there is a varying amount of the most recently sensed image shown on the

display. We purposefully use the average horizontal position of a vertical bar to measure this latency. As more of the latest image is drawn on the display, the average horizontal position of the vertical bar changes, providing a continuous estimate of the amount of actuated output that has been completed. In general, we found these methods to be robust to any potential errors in image processing.

Automated image processing is used to take measurements from the raw frames captured by the outside observer. The processing only happens within the windows highlighted in Figure 8 and are done independently in each window. The steps of the processing include histogram equalization, adaptive segmentation, and binarization. The histogram equalization brings the exposure to a human-visible level and reduces the variation of intensity between frames, which leads to cleaner object segmentation. In the adaptive segmentation process, a threshold based on the histogram is computed and used to segment the object of interest. The threshold is chosen such that 15% of the pixels in the window are below it, which is the expected amount of area taken up by the bar; in this way, the threshold value can vary from frame to frame (over time) to help compensate for lighting variations. In the binarization process, the grayscale image is converted to a binary image, where a pixel value of 0 indicates a background, and a value of 1 indicates the object. An example segmented frame is shown in Figure 9.

**2.2.1 Sensing and Actuation Property.** For System 1, we define the sensed and actuated property as the position of the black vertical bar as a percentage of its distance from the right border marker to the left border marker (see Figure 7). We used percentage rather than raw position to simplify calculations that determine when the actuated output event is in the same position as the sensed input event. The horizontal positions of the border markers were manually marked as shown by $L$ and $R$ in Figure 9. The top $T$ and bottom $B$ boundaries of the areas of interest were also manually marked. Note that these only needed to be marked once during experimental setup, because the boundaries did not move during experiments.

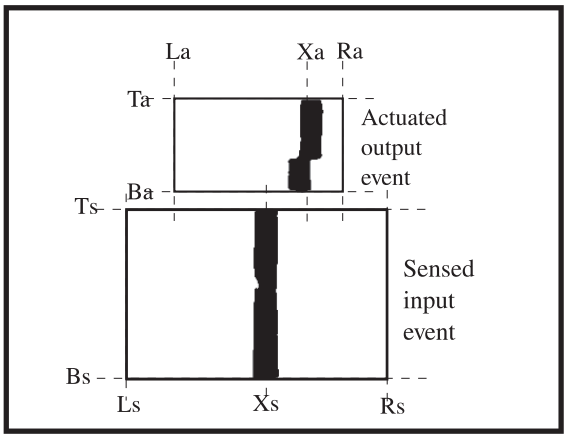


**Figure 9.** *Property (position) measured by outside observer.*

The position of the sensed input event is calculated as the object's first order moment in $x$ coordinates:

$$X_s = \frac{\sum_{y=B_s}^{T_s} \sum_{x=L_s}^{R_s} xI(x,y)}{\sum_{y=B_s}^{T_s} \sum_{x=L_s}^{R_s} I(x,y)}, \tag{1}$$

where $p$ is 1, $q$ is 0, $I(x,y)$ is the segmented binary image, and $X_s$ is the sensed input event's position. The sensed input property (position percentage) is then computed as:

$$P_s = \frac{\|X_s - L_s\|}{\|R_s - L_s\|}. \tag{2}$$

The position of the actuated output event is calculated similarly, substituting subscript a for subscript s in the variables shown in Figure 9 into Equations 1 and 2.

**2.2.2 Mapping Property to Latency Measurements.** For each outside observer frame, we measure $P_s$ and $P_a$. These can be plotted over time (over consecutive outside observer frames) as shown in Figure 10. To measure the latency at a particular frame $P$, we find the frame $P'$ where the actuated output property $P'_a$ is equal to the sensed input property $P_s$. This latency can be computed independently for every outside observer frame.

**2.2.3 Spatial Calibration.** The accuracy of this method depends to some degree upon the spatial calibration of the components. Figure 11(a) illustrates

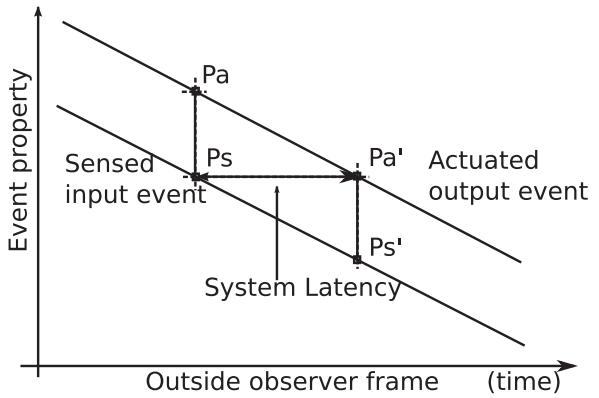


**Figure 10.** *Mapping property measurements to latency measurements.*

that the coordinate planes of the sensed input, actuated output, and outside observer should ideally be positioned in parallel. The projection of the sensed and actuated events onto the observer plane should ideally be orthogonal. In this manner, the percentage of the distance that $x_s$ has moved from $l_s$ to $r_s$ can be accurately compared against the percentage of $x_a$ from $l_a$ to $r_a$ (and other event motions can be compared similarly). The accuracy of this calibration can be observed in a recording that includes a period of the event property at rest, followed by motion, followed by another period of rest. Figure 11(b) illustrates the expected measurements of a system exhibiting constant latency if the measurement system is calibrated properly. The sensed and actuated measures should line up during the periods of no motion, verifying that the event property matches. During the period of motion, the sensed and actuated measures should be in parallel.

Poor spatial calibration of the components can cause inaccuracies in the measurement of latency. Figure 12 illustrates some possible errors. In Figure 12(a), note that the sensed and actuated measures do not line up when the event property is at rest. This would typically be caused by incorrect calculation of the left and right marker positions in the outside observer images. In Figure 12(b) it can be observed that the actuated measure is not parallel to the sensed measure during motion of the event property. This would typically be caused by nonparallel alignment of the components. Figure 12(c) illustrates the situation where the motion does not have constant velocity; in this example, it speeds up roughly halfway through the motion. Although this is not a calibration error, and does not affect the calculation of latency, it is illustrated so as to understand how nonconstant motion would appear in the measurements. Other possible sources of calibration error include excessive perspective projection and radial lens distortion in the outside observer.

For our experiments, we did our best to achieve near-ideal conditions by manually orienting the components to be as parallel as possible. The high-speed camera was placed at a distance of approximately 1 m and used a 6-mm lens. The sensed and actuated events were positioned to be observed toward the center of the high-speed camera's field-of-view. The values $x_s$ and $x_a$ were determined using centroids of automatically segmented regions, as opposed to single pixels or manually measured locations. The values of the left and right markers ($l_s, r_s, l_a, r_a$) were determined semiautomatically as the centroids of thresholded regions. The event motion, in this case the vertical bar moving across the apparatus, was accomplished by manually pulling the bar using a string. Periods of motion typically lasted 0.5 to 1.0 s and were done as near constant velocity as possible.

For some sensors and actuators, this method could be difficult to calibrate. For example, a head-mounted display is small compared to the monitor display we used in System 1. It would need to be positioned much closer to the outside observer, or optics could be used to magnify its image.

#### 2.2.4 Modeling the Camera-to-Monitor System Latency.

In this section, we briefly discuss a timing model of the expected latency in System 1. We used this model to generate simulated histograms of the latency, depending upon the settings of the camera and monitor. For example, we can change the shutter speed of the camera and the refresh rate of the monitor. We used this model to compare our measurements of the actual system against the histograms generated by our simulation.

The simulation model is based upon events, and uses five parameters to control the flow of information from

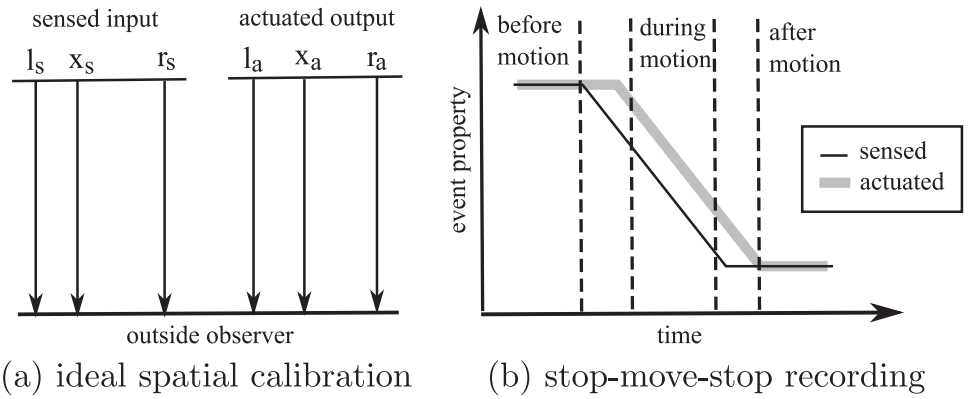


**Figure 11.** *The (a) ideal spatial calibration of the coordinate systems of the sensed input, actuated output, and outside observer can be observed in (b) the alignment of the event properties before and after motion, and the degree to which they are parallel during motion.*

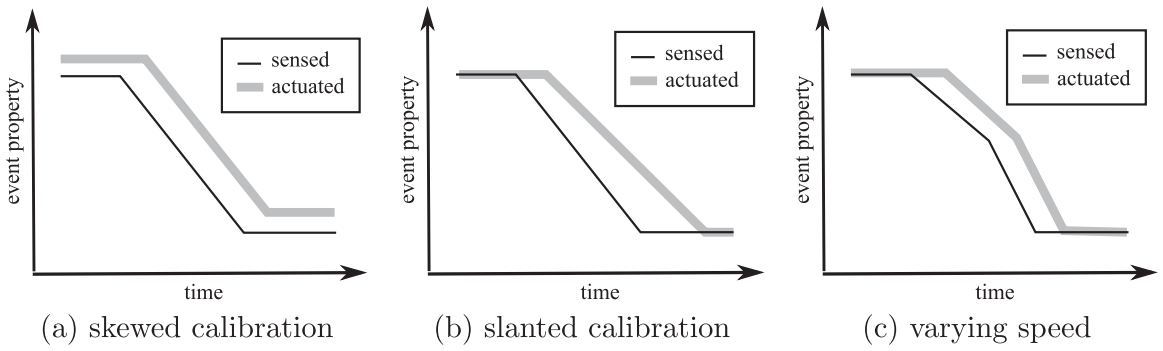


**Figure 12.** *Spatial calibration errors can cause (a) skew or (b) slant in the match of sensed to actuated measurements; (c) illustrates the effect of varying speed during the motion.*

sensing through actuation. The parameters are (1) the time that the data are being sensed, (2) the sensor clock rate, (3) the actuator clock rate, (4) the time that the data are being actuated and (5) the total time that the data are being processed by the computer. For System 1, these parameters correspond to the CCD exposure time, the CCD frame rate, the LCD refresh rate, the LCD response time, and the computer processing time. The clock rates were set to be equal to those of the real components. The total time spent in processing was determined by internal measurement within the program that processes the data; specifically, time stamps at the acquisition of data and the output of data were differenced and averaged over multiple runs. The times spent in sensing and actuation were arrived at through a combination of theoretical modeling about how the components work, as well as measurements, using the high-speed camera. The simulation runs by propagating an event, or in the case of System 1, an image, from sensing all the way through actuation. The end-to-end latency is determined as the time between the mid-point of sensing (the average of the accumulation of image charge) to the mid-point of actuation.

### 2.3 System 2

Our second system uses an orientation sensor for sensing and a motor for actuation. The orientation sensor is an InertiaCube (http://www.intersense.com/pages/18/11/); it uses the filtered results of 3-axis gyroscopes, magnetometers, and accelerometers to determine the

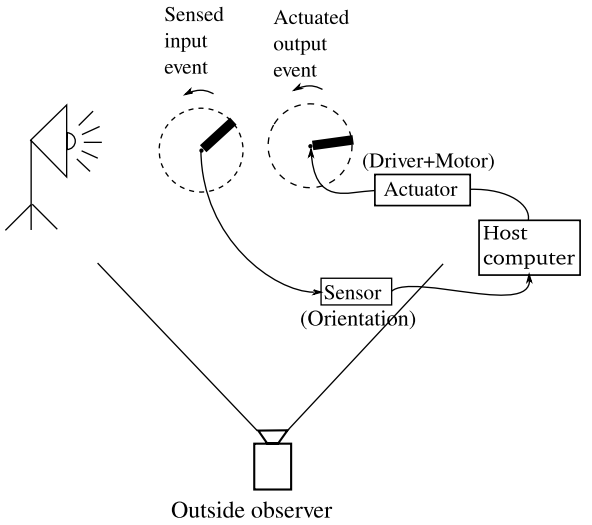


**Figure 13.** *System 2: orientation sensor to motor.*

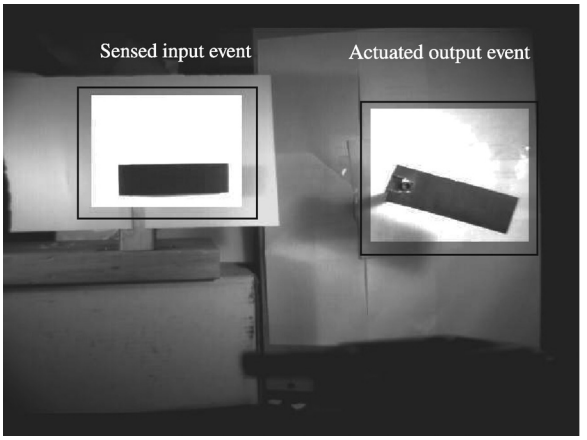


**Figure 14.** *Orientation-to-motor system as seen by the outside observer.*

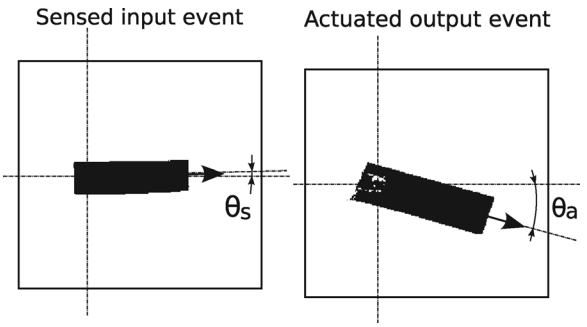


**Figure 15.** *Property (orientation) measured by outside observer.*

3-DOF angular pose at 110 Hz. The computer configuration is the same as in System 1. The motor is a Shinano Kenshi SST55D2C040 stepper motor (http://www.pikpower.com/New%20Site/New_pdfs/SKC/SKCnew.pdf). The motor driver is an Applied Motion Si2035 (http://www.applied-motion.com/products/stepper-drives/si2035).

Figure 13 shows a diagram of the experimental setup. The sensor is mounted on an apparatus that can be manually rotated. The computer reads the sensor and turns the motor to the same orientation. The outside observer is positioned to view both orientations. By comparing the two orientations, we can indirectly measure the system latency.

Figure 14 shows an example image captured by the outside observer. The sensor is mounted on a black bar that emphasizes one of the three angles of the orientation sensor. The actuator is similarly mounted with a bar attached to it so that its rotation can also be viewed by the outside observer.

**2.3.1 Sensing and Actuation Property.** For System 2, we define the property of interest as the direction of the black bar in the local coordinate system of both the sensed input event and the actuated output event. At startup, we assume the bars cannot both be manually turned to precisely 0°, so we record the initial measurement of orientation of each in its local coordinate system; these values are subtracted later from all subsequent measurements to eliminate the initial difference. We use automated image processing to determine the direction. Equalization, adaptive thresholding, and segmentation are carried out as described previously. Figure 15 shows an example result after adaptive thresholding and segmentation. The angle is computed by calculating a local eigenvector for each segmented object using moments and central moments. The $p$th and $q$th moments are computed as:

$$m_{pq} = \sum_x \sum_y x^p y^q I(x, y). \tag{3}$$

The center of the object is computed as:

$$(x_c, y_c) = \left( \frac{m_{10}}{m_{00}}, \frac{m_{01}}{m_{00}} \right). \quad (4)$$

The central moments are computed as:

$$\mu_{pq} = \sum_x \sum_y (x - x_c)^p (y - y_c)^q I(x, y). \quad (5)$$

Finally, the direction is computed as:

$$\tan(2\theta) = \left( \frac{2\mu_{11}}{\mu_{20} - \mu_{02}} \right), \quad (6)$$

where $\theta$ denotes the direction. The last step is to compensate for the difference between the initial orientations of both bars. This is done by subtracting the angle computed from the outside observer's first frame for each bar.

For each outside observer frame, we measure $\theta_s$ and $\theta_a$. These can be plotted over time as shown previously in Figure 10. Latency can then be calculated as described previously.

## 3 Results

### 3.1 System 1

Figure 16 shows the result for measuring latency continuously for System 1 over a 700-ms period of time. Comparing this result to Figure 10 shows that the latency is not constant (both lines are not straight). Instead, the latency varies by approximately 17 ms over a 33-ms period. This is due to the interplay between the 30-Hz clock of the sensor (camera) and the 60-Hz clock of the actuator (monitor). The default exposure time for the camera is 33 ms, equal to its clock rate; therefore, the snapshot of information captured in an image is an integral (or blur) across 33 ms. The default refresh time for the monitor is 17 ms, equal to its clock rate; therefore the actuation (or delivery) of its information takes place evenly across 17 ms. Figure 5 emphasizes this idea, that neither sensing nor actuation happens in an instant. Motion picture technologies, including cameras and displays, take advantage of apparent motion to fool the human visual system into perceiving continuous motion at rates near 24 Hz (Palmer, 1999). Our method for measuring latency shows how the latency actually looks at 1-ms resolution, as the amount of sensed data observed to have completed actuation varies.

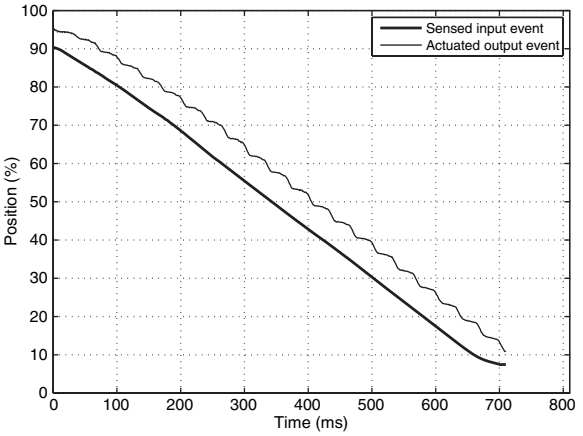


**Figure 16.** *Measured sensed input and actuated output for System 1.*

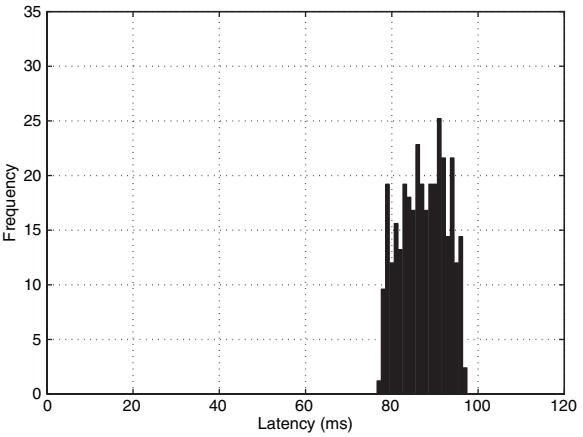


**Figure 17.** *Distribution of latency measured for System 1.*

Figure 17 shows the distribution of latency calculated from the data shown in Figure 16. If this histogram were the only result observed, one might conclude that the latency could be described by a random distribution. However, as is emphasized in Figure 16, this is not the case. The actual latency is cyclical. This demonstrates the problem with previous methods for measuring latency that do not observe it continuously, and instead report only averages.

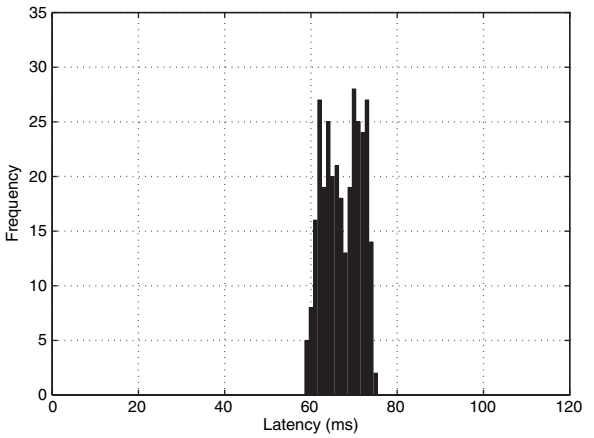


**Figure 18.** *Distribution of latency measured for System 1, with sensor (camera) using a faster shutter speed.*

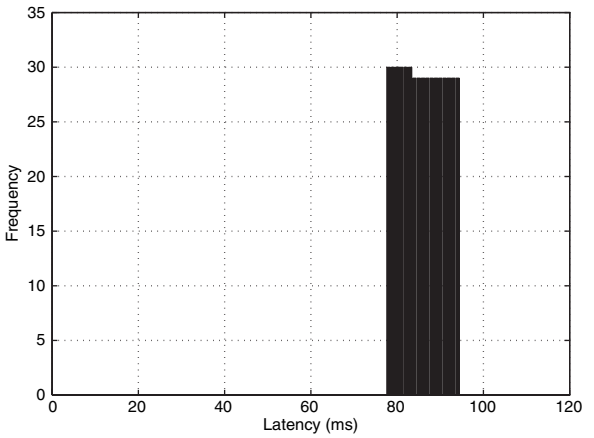


**Figure 19.** *Simulated distribution of latency for System 1.*

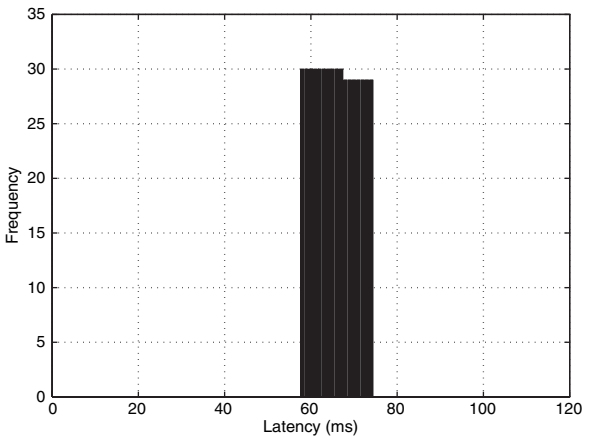


**Figure 20.** *Simulated distribution of latency for System 1, with sensor (camera) using a faster shutter speed.*

For a second test of the same system, we changed the exposure time of the sensor (camera) from 33 ms to 2 ms. Note that this did not change the clock rate of the sensor, only the amount of time integrated into an image during sensing (see Figure 5). Therefore, we expect an approximately 17-ms decrease in the distribution of latency. Figure 18 shows the result for measuring the distribution of latency for the faster shutter, confirming our expected decrease but otherwise showing the same shape.

As discussed previously, we created a model of System 1 in order to simulate measuring its latency and compare that against our real measurements. The only variables in the model are the clock rates of the sensor and actuator, and the amount of time spent in sensing, processing, and actuation. Figure 19 shows the result when the sensor (camera) has a 33-ms shutter speed, and Figure 20 shows the result when the sensor has a 2-ms shutter speed. Comparing these distributions to those shown in Figures 17–18 shows that they match against our measured results. This indicates that for purposes of modeling the latency, the necessary variables are the sensor and actuator clocks and the times spent in each of the three steps.

### 3.2 System 2

The experiment for System 1 was repeated many times and always showed the same latency distribution. However, for System 2, the distribution changed between trials. Figure 21 shows the measured distribution of latency for one trial and Figure 22 shows the distribution for a second trial. Looking only at these plots, or similarly only calculating averages, it is uncertain what is causing the difference in measured latency. Using our method to plot the latency continuously at 1-ms resolution reveals more information.

Figure 23 shows the continuous measurement of the orientation property of both the sensed input and actuated output, for the first trial. First, note that the step-like shape of the actuated line is similar to that observed for System 1 (see Figure 16), showing the interplay of the sensor and actuator clocks (approxi-

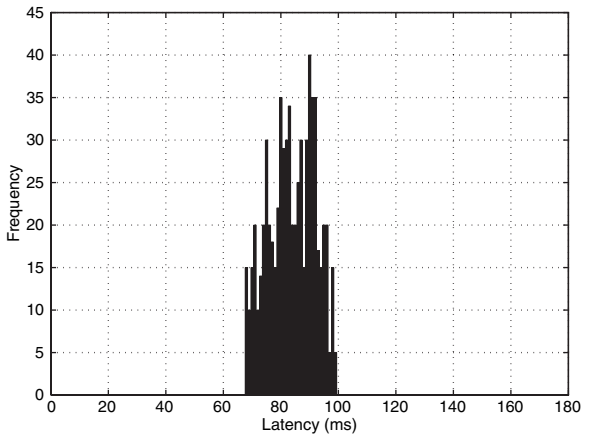


**Figure 21.** *Distribution of latency measured for System 2, first trial.*

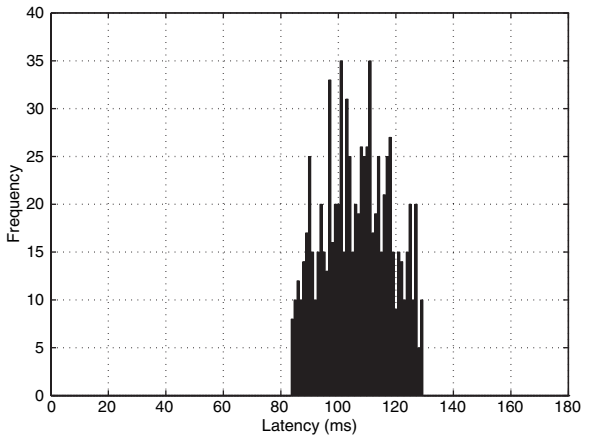


**Figure 22.** *Distribution of latency measured for System 2, second trial.*

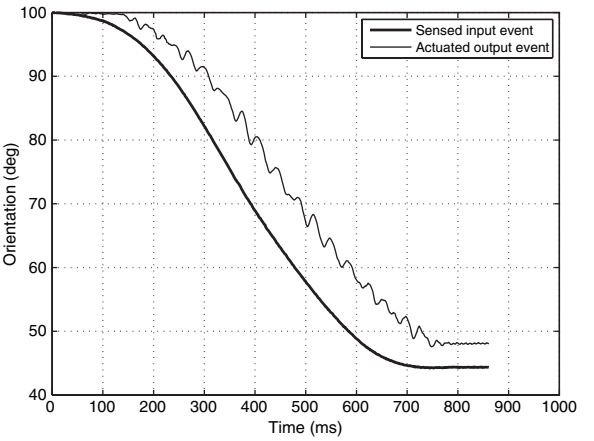


**Figure 23.** *Measured sensed input and actuated output for System 2, first trial.*

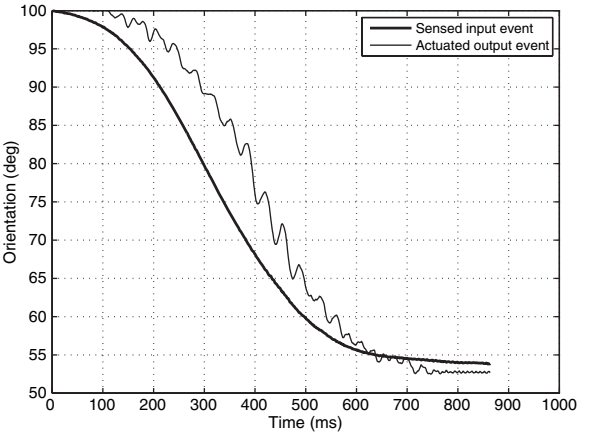


**Figure 24.** *Measured sensed input and actuated output for System 2, second trial.*

mately 33 Hz in Figure 23). Second, note that the lines are not parallel. The angular difference between the orientation sensor and motor was artificially set to 0° at initialization, but drifted to 5° after 800 ms at the end of the trial, as the sensor was rotated through approximately 50°. This is consistent with the amount of error our group has observed in the angular reading provided by this sensor (Waller, Hoover, & Muth, 2007). The result of this drift in sensor error is that the latency, which is the horizontal distance between the two lines, appears to change slowly throughout the trial. It is important to note that this is not so-called real latency, insofar as the system is not taking a differing amount of time to propagate the sensor readings through the system. However, it looks like latency to the end user of the system, because the time for the state of the output to match the state of the input is changing. In other words, a change in orientation sensor error can appear to the user to be like a change in latency; we henceforth refer to this phenomenon as apparent latency.

Figure 24 shows the same plot for the second trial. In this case, the sensor error was approximately −2° by the end of the 800-ms trial. Note again that the amount of horizontal distance between the two lines is varying. From Figures 23–24, we can conclude that a different amount of sensor error causes a different apparent latency.

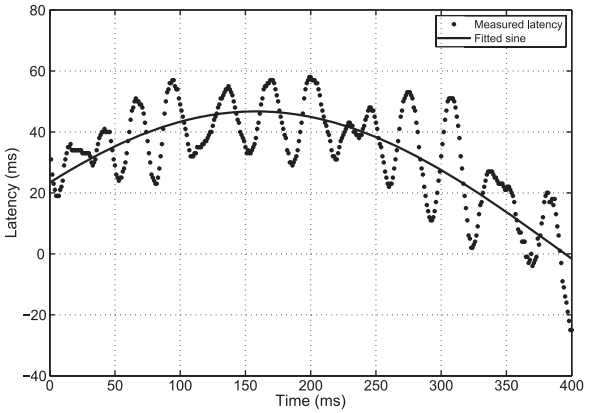


**Figure 25.** *Raw measurements of latency with fitted sine curve for a trial with 50° rotational motion.*

In order to characterize this variation, we fit sinusoidal curves to the apparent latencies (the horizontal differences between the lines in Figure 23). Figure 25 shows the raw measured latencies along with the fitted sine curve. The data were taken from the middle 400 ms of the trial where the calculation of latency is meaningful (at the beginning and end of the trials, when the object is not in motion, the latency cannot be determined). Note that the raw measurements are step-like because of the previously discussed interplay between the sensor and actuator clocks (appoximately 33 Hz with an amplitude ranging from 10–20 ms). The fitted sine shows the gradual change in latency as the sensor error drifts. From this figure, it can be observed that the frequency of the apparent latency is in the 0.5–1.0 Hz range, and that the magnitude of the apparent latency is approximately 20–30 ms.

We repeated this process for 10 trials. Table 1 lists the frequencies and magnitudes found for the fitted sines. Note that they vary due to differing amounts of sensor error in each trial, but the frequencies are generally in the 0.5–1.0 Hz range, and the magnitudes are generally in the 20–100 ms range. This amount of latency is certainly within the range perceivable by human end users. It is also well known that frequencies in this range, such as those caused by ocean waves and vehicle motions, are among the worst for causing sickness in humans (Golding, Phil, Mueller, & Gresty, 2001).

**Table 1.** *Frequencies and Magnitudes of Apparent Latency, for Ten Trials with 50° Rotational Motion*

| Latency (ms) | Frequency (Hz) |
|---|---|
| 27.2 | 0.73 |
| 24.0 | 0.38 |
| 72.9 | 0.59 |
| 13.4 | 0.83 |
| 22.6 | 1.78 |
| 82.2 | 0.72 |
| 23.3 | 1.01 |
| 46.7 | 1.07 |
| 50.6 | 0.67 |
| 88.5 | 0.64 |
| 97.4 | 0.95 |

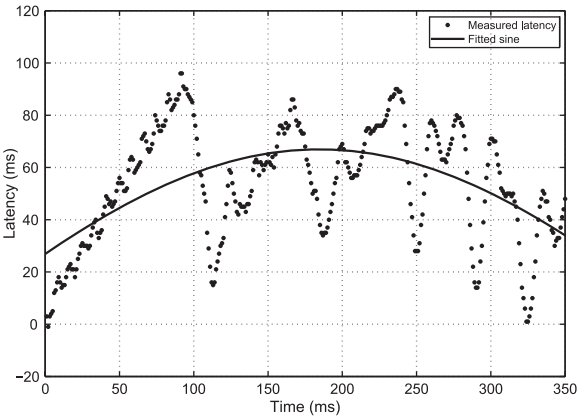


**Figure 26.** *Raw measurements of latency with fitted sine curve for a trial with 10° rotational motion.*

The motion in our first 10 trials was approximately 50° of constant velocity rotation in 800 ms. For a human turning his or her head, this motion is not unreasonable, but it is relatively far. We repeated this test with a slower, shorter rotation of approximately 10° in 800 ms. We conducted seven trials and fit sinusoidal curves to the apparent latencies. Figure 26 shows an example of raw measured latencies and fitted sine for one of the trials. Table 2 shows the calculated frequencies and magnitudes for the seven trials. We found that they are in the same range as for the first set of tests. Although we only systematically tested two motions (80° and 10°

**Table 2.** *Frequencies and Magnitudes of Apparent Latency, for Seven Trials with 10° Rotational Motion*

| Latency (ms) | Frequency (Hz) |
|---|---|
| 32.3 | 1.08 |
| 111.5 | 0.34 |
| 74.9 | 1.12 |
| 58.6 | 2.27 |
| 57.8 | 0.46 |
| 33.2 | 0.70 |
| 76.8 | 0.73 |

over 800 ms), these two results suggest that the sensor error may be somewhat independent of the speed of the motion, which matches our previous findings for evaluating the performance of the sensor (Waller et al., 2007). It also suggests that the sinusoidal variation in apparent latency, perceived by the user of a system incorporating this sensor, is somewhat independent of the speeds of motion made by the user.

### 3.3 Interpretation of Results

As discussed in the section on spatial calibration, several possible sources of error could affect the accuracy of the latency measurements. Although our method measures latency at the 1-ms resolution, we did not perform experiments that justify a quantitative claim of accuracy. Nonetheless, some reasonable interpretations of our results are justified. For the first system tested, the important result is that the latency can be seen to be varying approximately 17 ms at a 33-ms interval. It is difficult to conceive of a spatial calibration error that could cause this; a much simpler explanation is that it is due to the interplay of the sensor and actuator clocks. The measured magnitude and frequency also matches what would be theoretically expected based upon the clock rates of the components. The absolute range of values shown in Figure 17 may contain some error due to spatial calibration, but in our second test of System 1, in which we changed the exposure time of the sensor, we observed the theoretically expected shift in the latency distribution.

For the tests of System 2, the important result is that the latency can be seen to contain both the higher frequency variation seen in System 1, and an additional lower frequency variation. The higher frequency variation is approximately 33 Hz, again matching the theoretical interplay of the sensor and actuator clocks. The lower frequency variation is not constant; it changes from trial to trial. Although the higher frequency result is not easily explained by spatial calibration error, the lower frequency result could conceivably be explained by a combination of skew and slant (see Figure 12). However, any errors in spatial calibration should cause the same errors in repeated measurements of the same motion, which is not what we observed. Another possible explanation is that the measurement curves show a change in the motion of the feature being tracked. However, as Figure 12(c) illustrates, this would cause the two curves to change slope but stay parallel, which is not what we observed. Another possible explanation is that the error is in the stepper motor, but according to the data sheets for the components, the stepper motor and driver have a much higher accuracy than the orientation sensor. In addition, the drift of sensor error, in the range of $\pm 5°$, fits with our previous evaluation of this sensor (Waller et al., 2007). In order to help understand how orientation sensor error could produce the results we have observed, a practical example is useful. Consider a person wearing a head-mounted display that is being tracked by an orientation sensor. As the person rotates his or her head, if the head tracker shows a drift in error through the motion, then the images that are shown to the user appear to have a drift in latency. The latency can even appear to be negative, if the sensor reports a rotation so far ahead of the actual rotation that the images are displayed before the user's head achieves the reported rotation. The drift in orientation error seen in a standard compass can be seen with the naked eye, but state-of-the-art micro-electrical-mechanical system (MEMS) orientation sensors show a lower range of drift error. Our findings demonstrate that even a small drift in orientation sensor error can cause a noticeable oscillatory variation in system latency. It is important to note that this could only be observed by measuring the latency continuously.

## 4 Conclusion

In this paper we have described a new method for measuring system latency. The main advantage of our method is that it measures latency continuously at 1-ms resolution. This allows for the observation of changes in latency over sub 1-s intervals of time. While many other works in this area have measured latency at a precision comparable to our method, the standard practice has been to calculate averages of repeated measurements. Figures 16, 23, and 24 show the types of information our method can reveal that cannot be seen in average values, from which we emphasize two conclusions. First, we have found that differences in the clock frequencies of sensors and actuators cause a cyclical variation in latency; for the components we tested this was in the range of 15–50 Hz at magnitudes of 10–20 ms. Second, we have also found that the error drift in sensor readings causes variations in apparent latency. For the orientation sensor we tested, which is popular in virtual reality and robotics research, the variation in apparent latency was in the range of 0.5–1.0 Hz at magnitudes of 20–100 ms. This magnitude of latency is known to be perceivable by humans, and this range of frequencies is known to be near the frequency that causes maximum sickness in humans (Golding et al., 2001). These results suggest that the relationship between sinusoidal variation in latency and simulator sickness warrants further study. Other types of rotational and position tracking sensors should be tested using our method to discover whether similar frequencies of latency variation can be observed.

Adelstein et al. (1996) and Di Luca (2010) have both previously noted that system latency is not a constant. Their data show that the type of motion, and especially its frequency, affected the latency. The hypothesized cause was filters used in the tracking system to smooth and predict the motion. Our work agrees with theirs, that the latency can change across different motions. However, we believe it is the drift error in tracking that specifically causes the change in latency. Presumably, if a tracking error drifted similarly for multiple trials of the same motion, then a correlation between the tracking system error and the latency would be found. This may partly explain their findings. For our System 2 tests, we did not pursue this idea, but in our limited trials we observed noticeably different sensor errors. A larger number of trials needs to be performed to more fully explore this possibility.

The methods of Miller and Bishop (2002) and DiLuca (2010) may be modifiable to measure latency continuously. In particular, the method of DiLuca could presumably be operated at tens of KHz. In the future it would interesting to combine our approaches. This would allow for the evaluation of latency in systems that use sensors and actuators operating in the KHz range. In order to achieve this, it would be necessary to avoid using any high-pass filtering (as described in DiLuca), which removes variations of the type we are measuring, and to avoid using correlations for measurements, which only calculates averages.

## References

Adelstein, B., Johnston, E., & Ellis, S. (1996). Dynamic response of electromagnetic spatial displacement trackers. *Presence: Teleoperators and Virtual Environments, 5*(3), 302–318.

Akatsuka, Y., & Bekey, G. (2006). Compensation for end to end delays in a VR system. *Proceedings of IEEE Virtual Reality Annual International Symposium*, 156–159.

Bryson, S., & Fisher, S. (1990). Defining, modeling, and measuring system lag in virtual environments. *Proceedings of the SPIE, International Society for Optical Engineering*, *1257*, 98–109.

DiLuca, M. (2010). New method to measure end-to-end delay of virtual reality. *Presence: Teleoperators and Virtual Environments, 19*(6), 569–584.

Golding, J., Phil, D., Mueller, A., & Gresty, M. (2001). A motion sickness maximum around the 0.2 Hz frequency range of horizontal translational oscillation. *Aviation, Space and Environmental Medicine, 72*(3), 188–192.

He, D., Liu, F., Pape, D., Dawe, G., & Sandin, D. (2000). Video-based measurement of system latency. *Proceedings of the International Immersive Projection Technology Workshop*.

Liang, J., Shaw, C., & Green, M. (1991). On temporal-spatial realism in the virtual reality environment. *Proceedings of the 4th Annual ACM Symposium on User Interface Software and Technology*, 19–25.

Lippi, V., Avizzano, C., Mottet, D., & Ruffaldi, E. (2010). Effect of delay on dynamic targets' tracking performance and behavior in virtual environments. *Proceedings of the 19th IEEE International Symposium on Robot and Human Interactive Communication*, 446–451.

Liu, Y., Hoover, A., & Walker, I. (2004). A timing model for vision-based control of industrial robot manipulators. *IEEE Transactions on Robotics, 20*(5), 891–898.

Miller, D., & Bishop, G. (2002). Latency meter: A device to measure end-to-end latency of VE systems. *Proceedings of Stereoscopic Displays and Virtual Reality Systems*, 458–464.

Mine, M. (1993). *Characterization of end-to-end delays in head-mounted display systems.* The University of North Carolina at Chapel Hill, Technical Report TR93-001.

Olano, M., Cohen, J., Mine, M., & Bishop, G. (1995). Combatting rendering latency. *Proceedings of the 1995 Symposium on Interactive 3D Graphics*, 19–24.

Morice, A., Siegler, I., & Bardy, B. (2008). Action-perception patterns in virtual ball bouncing: Combating system latency and tracking functional validity. *Journal of Neuroscience Methods, 169*, 255–266.

Moss, J., Austin, J., Salley, J., Coats, J., Williams, K., & Muth, E. (2011). The effects of display delay on simulator sickness. *Displays, 32*(4), 159–168.

Palmer, S. (1999). *Vision science: Photons to phenomenology.* Cambridge, MA: MIT Press.

Roberts, D., Duckworth, T., Moore, C., Wolff, R., & O'Hare, J. (2009). Comparing the end to end latency of an immersive collaborative environment and a video conference. *Proceedings of the 13th IEEE/ACM International Symposium on Distributed Simulation and Real Time Applications*, 89–94.

Steed, A. (2008). A simple method for estimating the latency of interactive, real-time graphics simulations. *Proceedings of ACM Symposium on Virtual Reality Software and Technology*, 123–129.

Swindells, C., Dill, J., & Booth, K. (2000). System lag tests for augmented and virtual environments. *Proceedings of the 13th Annual ACM Symposium on User Interface Software and Technology*, 161–170.

Teather, R., Pavlovych, A., Stuerzlinger, W., & MacKenzie, I. (2009). Effects of tracking technology, latency, and spatial jitter on object movement. *Proceedings of the IEEE Symposium on 3D User Interface*, 229–230.

Waller, K., Hoover, A., & Muth, E. (2007). Methods for the evaluation of orientation sensors. *Proceedings of the 2007 World Congress in Computer Science, Computer Engineering, and Applied Computing.*

Ware, C., & Balakrishnan, R. (1994). Reaching for objects in VR displays: Lag and frame rate. *ACM Transactions on Computer–Human Interaction, 1*(4), 331–356.