

Architecture and Performance of the HI-Space Projector-Camera Interface

Richard May

Pacific Northwest National Laboratory
richard.may@pnl.gov

Bob Baddeley

Pacific Northwest National Laboratory
bob.baddeley@pnl.gov

Abstract

This paper presents the architecture and configuration of the Human Information Workspace (HI-Space) interaction environment. HI-Space is a projector-camera-microphone system that recognizes gesture, object and speech input. The system creates input stream events and uses HI-Space aware interface components allowing the creation of independent applications through an API.

To help understand performance characteristics of HI-Space, a procedure is presented that was used to determine the total system lag of the HI-Space to be 71 ms with an 11 ms standard deviation. A study is also presented that directly compares HI-Space and a mouse in the task of acquiring stationary targets. The study shows that users can acquire targets faster and just as accurately on the HI-Space system.

Finally, an example application to support analysis of 3D biological data sets is discussed. The application is an ongoing research effort to explore complex interface design using the HI-Space architecture.

1. Background and Introduction

Projector-camera systems can take many forms. Over the last decade, several projection table environments have been developed that incorporate the ideas of directly mediated interaction where the line between physical and electronic manipulation are blurred [1,4,7,8,11,13,14,16]. Figure 1 shows the basic configuration used with the HI-Space video tracking system with a picture of one of the tables.

On the HI-Space, the table interaction display is kept horizontal, at a height of 1 meter, rather than being tilted toward the user. This allows for multiple users to interact simultaneously around the table. A Panasonic BP334 (unfiltered black-and-white NTSC) camera is placed over the table to capture user interactions, and a Kodak safelight filter is used to stop visible light from reaching the camera. The projector and infrared (880nm peak wave length) sources are

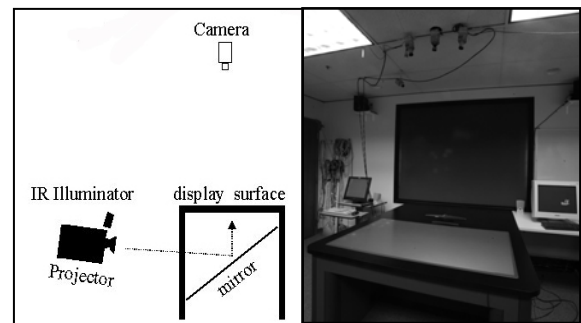


Figure 1. Left – Illustrates the type of project table used in the studies. Right – A picture of one of the tables.

under the table. The table display surface consisted of glass with a diffuser layer.

The HI-Space [8] libraries are capable of tracking multiple hands. A common task used in computer interaction is obviously the selection of a target, which requires the identification of a tip for all pointers. The tip of each hand is found by first recognizing all digits (fingers). Then the system assigns one of the digits as being the tip. As depicted in Figure 2, this defaulted to the trivial case when only one digit is used. In the case of multiple fingers, the system tries to identify the most logical digit to use. When the user is holding a pointer or stylus this is used as the tip just as a digit would be.

HI-Space also tracks objects placed on the display

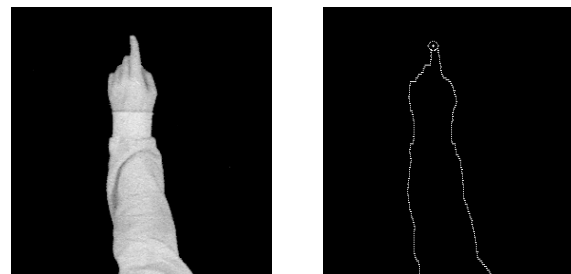


Figure 2. Left - Segmented image of arm. Right - Same arm after outlined and the tip found.

surface and captures object shape characteristics such as perimeter, size, and location. The physical configuration (camera over the table) also provides the capability to track entities anywhere over the table not just on the table surface. This creates an interaction volume over the table providing for a more flexible interaction space for some tasks.

2. HI-Space Architecture

The HI-Space software architecture is modular in design, allowing for maximum code reuse and minimizing development time for application development. The software is designed such that the underlying HI-Space computer vision system passes events through a pipeline that routes the event to the appropriate handler. Figure 3 shows the high level blocks of the architecture. This provides an API isolating the application from the HI-Space routines allowing for a wide range of applications.

The core computer vision routines are in C++. We have a Java Native Interface wrapper around the core routines. The wrapper class uses JNI to import the information provided by the underlying C++ library.

On every video frame state information is updated about each pointer (we generically refer to any hand or other pointing device as a pointer) or object detected by the camera. The speech engine is also queried to see if a new speech command has been issued. We are only using speech to supplement our command structure at this time so dictation is not supported.

Applications are created using HI-Space aware extended Java components. This system essentially creates an interface for an application very similar to one for a mouse or keyboard. In Java, an application can implement a `MouseListener`, and handle events such as `mouseMoved`, `mouseClicked`, and `mouseEntered`. Similarly, our architecture allows applications to implement a `HiSpacePointerListener` and handle events such as `HiSpacePointerMoved`, and `HiSpacePointerPose`. We are developing a growing

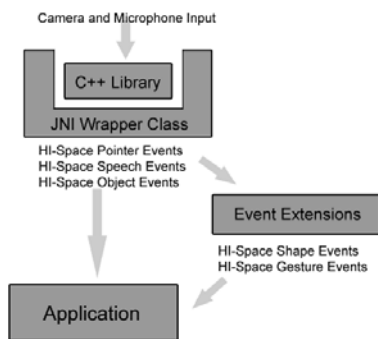


Figure 3. Block diagram of the HI-Space software architecture.

collection of events. As an example, events are triggered when a pointer enters the camera field of view or leaves the field of view. This is different from a mouse interaction device. Not only is there normally only one mouse but it is always at some screen location. In the HI-Space, there can be many pointers, and they freely enter and leave the tracking environment.

There are two other listeners, `HiSpaceObjectListener` and `HiSpaceSpeech-Listener`, available to applications. As the names imply, the former is to respond to objects placed in the environment and the latter initiates events when a speech command is given. It is up to the application to have listeners and process the events appropriately.

There are two classes of HI-Space events: low and high. Low-level events come directly from the input stream with minimal processing. High-level events are derived by monitoring low-level events and looking for combinations. The creation of high-level events occurs in the Event Extensions module.

Not all low-level event data is derived only from the information available from the input stream. There is, for example, a built-in hand pose library. Pointers are constantly being monitored to assess if they match any of the recognized poses. For objects there is a shape recognizer. If an object outline matches an entry in the shape library this information is also provided to the application.

An example of a high-level event includes gesture recognition. A gesture is composed of a series of poses and/or motion patterns over time. High-level events can also be multi-modal. A gesture can be combined with a speech command to provide more options to the user.

3. Determining Total System Lag

Total system lag is defined as the time between when a detectable action is initiated by the participant and a detectable response is given by the system. Determining lag is important for any interaction system. Lag has been shown to have a significant impact for different interaction devices [6, 15]. Because video tracking has a slow refresh rate, 30 frames per second (fps) for NTSC, understanding lag is even more important as delays in tracking could significantly increase lag.

In determining lag, the detectable action is when a tip passes a predetermined point while the detectable response is the lighting of pixels on the display screen. The measured total system lag for the HI-Space was 71 ms with a standard deviation of 11 ms. This value was derived when the system was running at 30 fps. The method used for measuring total system lag was

based on work at University of North Carolina at Chapel Hill [9].

Figure 4 shows the apparatus used to determine total system lag. The components include a spring-loaded armature, photodiode, and timer unit.

Armature: The armature contains a spring-loaded arm mounted on a pivot point and an infrared sensitive photodiode/infrared sensor pair. A reflective surface was attached to the arm to facilitate detection by the photodiode. When the armature's reflective surface passes under the sensor, a signal is transmitted to the timer unit through the A input. The armature was used by pulling it back, away from the sensor, to a fixed point and releasing it. This allowed the spring to recoil as the armature rotated back to its resting state. The tip of the armature had a sharp point to allow better tracking of the tip.

Photodiode: This was a separate photodiode that was sensitive to visible light and placed at a different location on the table surface from the armature. When the armature tip was detected passing a predetermined screen location, the screen region under the photodiode was lit with white light for a single frame. The photodiode detected the lighted state and transmitted a signal to the timer unit. This light-detecting photodiode was connected to the B input on the timer unit.

Timer Unit: The timer unit was used to calculate the difference between when the armature transmitted a signal and when the photodiode transmitted a signal. A Phillip's PM6680 A/B timer counter was used.

The calibration between the HI-Space tip tracking and the armature was critical for getting an accurate measure of the total system lag. Calibration was conducted by finding the exact armature position that triggered the sensor to transmit the signal to the timer unit.

Six hundred trials were run to determine total

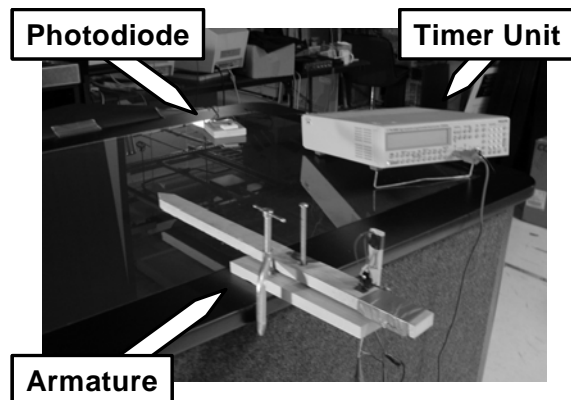


Figure 4. Apparatus used to capture delay time between an action on the table surface and the updating of the display.

system lag generating a 71 ms mean with a standard deviation of 11 ms when tracking at 30 fps. The procedure involved pulling the armature back to a set point, releasing it, then reading the difference between the two times as recorded by the A/B timer.

Tracking will not always occur at 30 fps. There are many reasons why the effective tracking rate could slow down in a system like the HI-Space. The impact on total system lag depends on where the slowdown occurs in the pipeline. In the three reduced frame rate conditions presented next, the procedure was the same as the previous study with only minor exceptions. The only procedure changes were that the frame rate was restricted to 10fps and 200 trials were run for each of the 10fps conditions.

The first condition tested total system lag if the computer vision routines were only able to use 1 in 3 of the video frames because of the amount of time to process the images. In other words, the HI-Space C++ library was only triggering events at 10 fps. For this condition the total system lag was 215 ms with a standard deviation of 31 ms.

Another possible location for lag is in the application itself. The tracking would still be reporting at 30 fps but the application is only using one third of those due to processing delays. In this condition the total system lag was 117 ms with a 32 ms standard deviation.

The final condition was more of a contrived case to help evaluate the previous two. The system (tracking and application) were running at 30 fps but only one third of the frames were actually rendered. The application simply failed to render all the frames. In this condition the total system lag was 100 ms with a 29 ms standard deviation.

The relative ordering of these three conditions makes sense. In the third condition, we are using the most recent tracking information only showing it at 10 fps. In other words the tracking information is only roughly 1/30th of a second old for each update. Contrast this to the first case where the tracking data being used is closer to 1/10th of a second old for the display update. The second case is somewhere between as the application was given the most recent tracking data but had delays before rendering.

This shows the importance of making video based tracking systems as fast as possible. Any delay in the update frame rate due to the tracking system has the greatest impact on lag at the other end of the pipeline.

4. Fitts' Law and Data Analysis

Fitts' model [2] combines information theory with human motion. The equations can be summarized as stating the relationship between acquiring a target and

the size and distance of the target. Today, Fitts' model is a commonly used tool to assess target acquisition performance characteristics in various computer environments. There are many variations of the original work by Fitts [5]. The study presented here use the following form:

$$IP = \frac{ID}{MT} \quad (1)$$

where

IP = Index of Performance: capacity in bits/second

ID = Index of Difficulty: minimum amount of information

MT = Mean Time: average time per response.

The term ID, from Equation 1, is expressed in the following form:

$$ID = \log_2 \left(\frac{2A}{W} \right) \text{bits / sec} \quad (2)$$

where

A = amplitude or distance to target

W = width of target.

A typical Fitts' study has the user repeatedly acquiring targets at various sizes (width) and distances (amplitude) to evaluate performance based on ID. Participants are typically instructed to move back and forth between two targets as quickly and as accurately as possible. For this work a variation of this approach was used called discrete acquisition. In discrete acquisition, the participant moves back to a starting point after acquiring the target. Once at the starting location a target can again be acquired. This better mimics the process of typical computer interaction tasks such as icon selection. This procedure was used for both the HI-Space and mouse conditions.

Mean times, error rates, y-intercepts, and index of performance values were calculated using Microsoft® Excel. ANOVA and other confidence tests were calculated using SPSS 10.1.

5. Study to Compare Acquiring Targets by Direct Pointing and a Mouse

Shneiderman [12] provided a definition of 'direct manipulation' that has a broad connotation and can include many interaction devices including HI-Space or a mouse. However, we feel there are some fundamental differences between interacting with a mouse and the intimacy between people and objects that can be found in real-world direct manipulations. Ishii [3] expanded on the idea of direct manipulation

to include the overlapping of functional and cognitive processes between electronic and real-world interactions. We wanted a simple study to see if our technology was sufficiently developed to evaluate different types of direct manipulation.

In this study, participants performed a Fitts' style selection task by making horizontal motions to acquire a stationary target. This task was performed with different sized targets and varying distances from the starting point. All participants performed the interaction by standing at the HI-Space system (see Figure 5) using a hand held stylus and sitting in front of a computer monitor using a mouse.

5.1 Participants

Ten participants took part in the study. All were right handed and 40 years of age or younger. All participants were expert users of the mouse. That is, they used a mouse on a daily basis and had for a minimum of 5 years. Participant experience on HI-Space varied from occasional to no previous experience.

5.2 Apparatus

A rear-projection display table with a 1024-mm wide and 575-mm deep (used region) display was used as the interaction surface. The display resolution was set to 1024x575 pixels. This provided a 1 pixel per mm conversion rate. A 31-cm stylus was used for all HI-Space interactions by all participants as a pointing device. The stylus was not instrumented, and the tip was tracked using only a HI-Space video tracking system. The location of the tip of the stylus was represented on the screen by a 10-mm by 10-mm crosshair. All targets were displayed as solid filled circles.

For mouse based interactions on a traditional desktop system a 20-inch flat LCD panel display was used. The display was set to 1024x768 resolution but only 1024x575 were used by the study. A roller-ball mouse was used with the ratio set so that all interactions from start point to target could be made in one stroke across the mouse pad. The mouse cursor was a 10-pixel by 10-pixel crosshair.

5.3 Procedure

Participants used equivalent procedures with both a mouse on a typical desktop environment and a hand-held stylus on the HI-Space. For the HI-Space, all participants were directed to hold the stylus by grasping it toward the back to maximize the stylus length. Participants stood in front of the display table

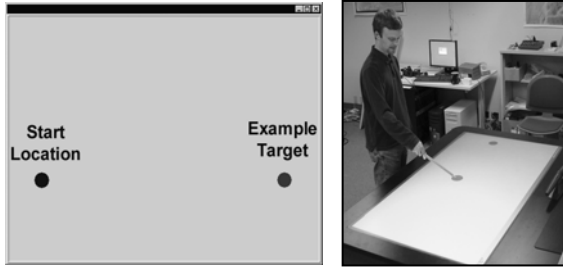


Figure 5. Left - Example screen indicating start and target locations. Right - Participant selecting the target.

as depicted in Figure 5. The stylus did not provide a button or other instrumentation to allow indication of target selection by the participant. Instead, selection was indicated with dwell time. If the stylus remained still for approximately 133 ms (4 video frames), the selection operation was triggered. For mouse interaction, participants were provided with the opportunity to place the mouse in a comfortable location. To keep the interactions equivalent between the two environments, the same dwell time procedure was used rather than a mouse button click for selection. If the mouse cursor remained still for 133 msec, the selection operation was triggered.

All timed interactions, for both environments, were moving from a starting location, on the left, to a destination target, on the right. The subsequent return to the starting location was not monitored. The starting location was a constant 40-pixel-diameter circle. The start circle always appeared at the same screen location on the left side of the screen. Target circles varied in width and amplitude from the starting location but were always located directly to the right. Participants were instructed to select the target on the right as quickly and accurately as possible but that there was an emphasis on accuracy. They were also informed that the return motion to the starting location was not timed and to take as much time as was needed.

For the HI-Space, the 10-pixel by 10-pixel crosshair was the electronic representation of the stylus tip and appeared under the tip as would be expected. For mouse-based interactions, the 10-pixel by 10-pixel crosshair was no different than any other electronic cursor and represented the pixel of interest.

At the start of a trial, both the starting location and destination target were presented. Participants placed the crosshair inside the start circle. Once the start circle was selected, in other words the crosshair did not move for 133 msec and was inside the start circle, an audible start signal was given. The start circle would disappear, and the target circle would change color. Participants would then move to select the target circle. For the HI-Space, the enforced

procedure required lifting the stylus tip from the table and placing it back on the display surface within the target circle. For mouse interaction, it was the act of moving the mouse cursor to the target circle. Once the target was selected, another audible completion signal was given. The start circle would reappear, and the next target circle would be presented. The participant would then return the crosshair back to the start circle to initiate the next trial.

Throughout all practice and recorded sessions for both interaction environments an audio cue was provided to indicate hitting or missing a target. The audio tone was different depending on whether participants placed the crosshair inside the target (hit) or outside the target (miss).

5.4 Design

The study was a fully within-subject repeated measure design. The factors were the type of interaction environment used (HI-Space or mouse), target amplitude (210 pixel, 420 pixel, 840 pixel), and target diameter (14 pixel, 28 pixel, 56 pixel, 112 pixel). This generated 6 indexes of difficulty from 1.91 bits to 6.91 bits (Fitts' model).

The study was blocked by interaction technique, so each participant received all trials for a single interaction condition before proceeding to the next. Additionally, all trials of a given amplitude-width condition were presented before proceeding to the next amplitude-width condition within a block. Each participant had 2 blocks. These blocks were

- Practice: 20 trials each of 6 of the 12 possible amplitude-width combinations.
- Recorded: 20 trials each of all 12 amplitude-width combinations with the combined cursor and audio feedback.

The practice trials were constant across all participants for order and amplitude-width combinations. The amplitude-width combinations in block 2 were in a unique order for each participant.

During a trial, the system tracked the starting pixel location, ending pixel location, and travel time. Travel time is calculated as the time from which the participant started moving the stylus or mouse after the start signal was given to when the stylus or mouse stopped moving.

5.5 Results

Table 1 presents the performance results for each of the interaction conditions tested over the 12 amplitude-width combinations. The error rate is the percent of trials the user failed to place the crosshair inside the target circle. There is a significant main

Table 1. Performance results for interaction conditions tested over all ID conditions.

	Interaction Type	
	Mouse	HSE
Mean Time (ms)	667	505
Error Rate	4.2%	3.2%
Index of Performance (bits/s)	7.1	10.4
Y intercept (ms)	45	81

effect for interaction technique on movement time ($F_{1,9} = 77.241, p < 0.0001$) and index of performance ($F_{1,9} = 55.608, p < 0.0001$) but not for error rate ($F_{1,9} = 3.095, p > 0.11$). Figure 6 shows the movement times vs. index of difficulty for the two interaction conditions.

There was no significant main effect for interaction technique on error rates. However, 82% of the errors occurred in the six most difficult amplitude-width conditions. So, if there are any differences in error rates, they could be masked by the amplitude-width conditions where essentially no errors were occurring.

Performance results for the six amplitude-width conditions with the highest index of difficulty (4.91, 5.91, and 6.91) are shown in Table 2. There is again a significant main effect for interaction techniques on mean time ($F_{1,9} = 57.057, p < 0.0001$) but still not for error rates ($F_{3,11} = 0.657, p > 0.438$). Linear regression analysis was not conducted because of the reduced range in the indexes of difficulty.

This study shows that in this environment stylus based interaction actually allows the user to acquire stationary targets faster, more efficiently (i.e., better index of performance), and with no more errors than using the mouse. This result occurred in spite of participants' inexperience with the HI-Space when compared to years of experience with the mouse.

The fact that error rates on the HI-Space were as good as those from the mouse also occurred despite the limited accuracy of the HI-Space. Unlike the mouse, which has pixel-level accuracy, the HI-Space will not allow users to consistently and repeatedly acquire a single pixel target. In fact, the minimum target width of 14 pixels was based on observational data that indicated smaller targets would be too

Table 2. Performance results for the 3 highest ID conditions.

	Interaction Type at high ID	
	Mouse	HSE
Mean Time (ms)	824	610
Error Rate	6.8%	5.9%

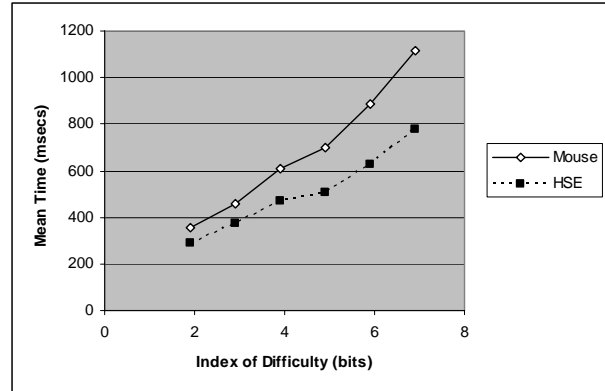


Figure 6. Mean performance time vs. ID for interaction conditions.

difficult to select on the HI-Space for the purpose of this study. The limit of these findings is indicated in Table 3. This shows the error rates for the two interaction types based on target size. For the smallest target size of 14 pixels, the data suggest that error rates are starting to favor the mouse. Given even smaller targets, the mouse would likely have the advantage. This study was also performed under the constraints of “as quickly and accurately as possible.” Removing the time constraint would likely allow a participant using the mouse to acquire smaller targets more accurately because the mouse is capable of accessing individual pixels.

Existing research also supports the advantages of direct interaction. Another study [10] looked at different types of pointing devices on a large screen. That study was a Fitts' style tapping task and used a SmartBoard for directly mediated interaction but also used a mouse to interact on the same screen. They found that directly mediated interaction was faster and more accurate than using the mouse.

5.6 Study Conclusion and Discussion

It is important to keep in mind that while the task was the same for both environments; the functional process to complete the task was quite different. On the HI-Space, this experiment was a discrete Fitts' style tapping task. With the mouse, it was really a selection task, but it cannot be considered a tapping

Table 3. Error rates for interaction techniques based on target size.

	% misses by target Width (pixels)			
	14	28	56	112
Mouse	9.38%	4.37%	2.39%	0.51%
HSE	10.85%	1.51%	0.34%	0.00%

task. The act of selecting a target with the mouse requires a different set of physical interactions, and no tapping of the target is involved. This is not considered a problem for the study since the purpose was to evaluate performance at acquiring targets. The fact that two different physical processes are required is at the heart of the question.

The HI-Space condition was faster for target acquisition. This was despite participants' unfamiliarity with the technology and the accuracy limitation of its current state of development. This could indicate that direct interaction as supported by HI-Space is drawing upon the participants' innate pointing ability. By using hand-eye coordination future alternate interaction environments could provide more intuitive ways of interacting with electronic data.

6. An Example Application

The prototype application being developed is an increasingly complex interface that focuses on biologists' need for a tool to analyze volumetric datasets. The task is to develop a tool that provides the same functionality as is available in products biologists use today but also add new functionality. This application is also a demonstration and exploration of the depths to which we can exploit HI-Space technology.

We began by dividing the task into two parts: the interface, and the visualization. Each task is assigned to a different computer. One machine acts as the renderer, loading datasets and allowing manipulations to be performed on them. The rendering occurs on a 8 ft rear-projected wall screen. The other computer serves as the HI-Space table interface. Manipulations are done on the HI-Space table, which interprets those actions and sends commands to the rendering machine that performs the command.

The interface is a unique design that incorporates

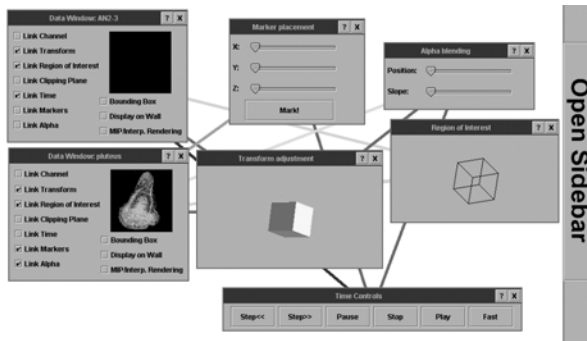


Figure 7. Image of the HI-Space biology application interface

familiar concepts to computer users with novel ones whose purpose is to leverage the enhanced abilities and interactions that the direct interaction provides. The core of the application centers around two types of windows: data and control. A data window is a representation of a dataset. It is a simplified view, since the primary data visualization is handled by the other machine, which allows manipulations to be done to the dataset. A control window is essentially a tool that can be applied to a dataset. Some of the tools we have incorporated include a transform window, which allows the user to rotate the data set, a clipping plane window, which adds an adjustable clipping plane to the volume, a time window that controls playback of time-based datasets, and a histogram window, which allows the user to retrieve voxel intensity information from the dataset along a user controlled line.

Windows can be opened by several different techniques. Users can select from a menu by pointing and selecting the window to open. Once opened, the window can be moved by pointing and gesturing. This technique lets the user organize the windows in a manner similar to point and click with a mouse. Placing the appropriate objects on the table also open windows and locate them on the objects. Each window has been assigned a specific object. A window can also be opened by a spoken command.

We also introduce two other concepts called linking and locking. Linking refers to the ability to create links between windows and is adapted from visual programming techniques. A link is shown as lines from one window to the next. Linking a control window to a data window indicates that you want to use the tool on that dataset only. This is particularly useful when multiple datasets are being viewed, but the user only wants to manipulate one.

Locking allows the user to perform manipulations while maintaining focus on the dataset. By placing a particular token (physical object) in a window, all events that take place anywhere on the table are directed towards that window, so the user can move their hand over the whole space to change a dataset. Otherwise only interactions within the window space will affect that window. By linking two control windows and locking them, the user can build a macro that allows them to perform multiple manipulations, like rotating the dataset and placing markers at regions of interest.

The application architecture that allows this is fairly simple. All events pass through a listener, which will either act on the event or pass it to the appropriate window. Each window also has a listener, which will again either act on the event or pass it to the appropriate component within the window. Components can be simple Java Swing or AWT

elements that have been extended to process HI-Space events. These listeners are intelligent enough to know that when a window is locked, all events should be passed directly to that window. When a data window is linked to a control window, all manipulations that take place in the control window are passed to the data window, which then passes it to the communications class that implements the message passing to the rendering computer.

We have found the modular architecture extremely useful in our interface and interaction research. It has been expanded many times to incorporate new features in the effort to create more complex and realistic applications. It has also been stripped down and reused for new programs to incorporate just the bare essentials.

7. Conclusions and Future Work

As research on projector-camera systems like HI-Space and others expand, we need to better quantify their actual usefulness for real-world problem solving. A better understanding of how and where more direct interaction works and doesn't work will provide the guidance needed to focus future research.

When NTSC video is used as the tracking technology, maintaining maximum frame rates is required to minimize system lag. This puts a greater premium on optimizing the computer vision routines. Our research showed a total system lag of 71 ms which is still significant when compared to common input devices like the mouse. Yet our study showed that users can still perform some tasks exceptionally well, even better than they can be done on a traditional desktop system despite the lag.

By developing a robust architecture we can create more complex interaction environments to better understand the limitations of technologies like the HI-Space. Our future efforts will continue to quantify user performance characteristics by developing new ways of performing the everyday real-world tasks that computers are used for.

8. Acknowledgments

Parts of this work were conducted with support from the U.S. Department of Energy. We also thank Thomas Furness and Jim Thomas for their support and guidance.

9. References

[1] Ashdown, M. and Robinson, P., Experiences Implementing and Using Personal Projected Displays, IEEE

International Workshop on Projector-Camera Systems (Procams) 2003.

[2] Fitts, P. M. The Information Capacity of the Human Motor System in Controlling the Amplitude of Movement. *Journal of Experimental Psychology* 47(6): 381-391. 1954.

[3] Ishii, H., M. Kobayashi, et al. Interactive Design of Seamless Collaboration Media. *Communications of the ACM* 37(8): 83-97. 1994.

[4] Koike, H., Y. Sato, et al. Interactive Textbook and Interactive Venn diagram: Natural and Intuitive Interfaces on Augmented Desk Systems. SIGCHI '00.

[5] MacKenzie, I. S. Fitts' Law as a Research and Design Tool in Human-Computer Interaction. *Human-Computer Interaction* 7: 91-139. 1992

[6] MacKenzie, S. I. and C. Ware. Lag as a Determinant of Human Performance in Interactive Systems. *InterCHI '93*.

[7] Matsushita, N. and J. Rekimoto. HoloWall: Designing a Finger, Hand, Body, and Object Sensitive Wall. *UIST '97*.

[8] May, R. HI-Space: A Next Generation Workspace Environment. Master's Thesis in Electrical Engineering Computer Science. Pullman, Washington, Washington State University. 1999.

[9] Mine, M. R. Characterization of End-to-End Delays in Head-Mounted Display Systems, University of North Carolina at Chapel Hill. 1993.

[10] Myers, B. A., R. Bhatnagar, et al. Interacting at a Distance: Measuring the Performance of Laser Pointers and Other Devices. *SigChi 2002*

[11] Rekimoto, J. and M. Saitoh. Augmented Surfaces: A Spatially Continuous Work Space for Hybrid Computing Environments. *SIGCHI '99*.

[12] Shneiderman, B. *Designing the User Interface: Strategies for Effective Human-Computer Interaction*, Addison-Wesley Publishing Company Inc. 1992.

[13] Ullmer, B. and H. Ishii. The metaDESK: Models and Prototypes for Tangible User Interfaces. *UIST '97*.

[14] Underkoffler, J. and H. Ishii. Urp: A Luminous-Tangible Workbench for Urban Planning and Design. *SIGCHI '99*.

[15] Ware, C. and R. Balakrishnan. Reaching for Objects in VR Displays: Lag and Frame Rate. *ACM Transactions on Computer-Human Interaction* 1(4): 331-356. 1994.

[16] Wellner, P. Interactions with Paper on the DigitalDesk. *Communications of the ACM* 36(7): 87-96. 1993.