INTERFACES AND CONTROL SYSTEMS FOR INTUITIVE CRANE CONTROL

A Thesis
Presented to
The Academic Faculty

by

(Kelvin) Chen Chih Peng

In Partial Fulfillment
of the Requirements for the Degree
Master of Science in the
School of Mechanical Engineering

Georgia Institute of Technology
December 2009

INTERFACES AND CONTROL SYSTEMS FOR INTUITIVE CRANE CONTROL

Approved by:

Professor William Singhose,
Committee Chair
School of Mechanical Engineering
Georgia Institute of Technology

Professor Nader Sadegh
School of Mechanical Engineering
Georgia Institute of Technology

Professor Jun Ueda
School of Mechanical Engineering
Georgia Institute of Technology

Date Approved: October 26 2009

ACKNOWLEDGEMENTS

I would like to thank my advisor, Dr. William Singhose, for his excellent guidance and helpful suggestions that nudged me in the right direction. I am most grateful for his seemingly inextinguishable flame of optimism. Even in the most dire situations, it is hard not to be infected by his contagious optimism and motivational energy; this has helped greatly me, and my colleagues. Special thanks to Dr. David Frakes and Semir Gessesse of ASU for their support in the radio-frequency-based interface. I would also like to thank my lab-mates, and to my two *senpai*'s, Dr. Khalid Sorensen and Dr. Joshua Vaughan, for their extensive tutelage over the past two years.

I am grateful to my parents for all their love and support, as I am the result of all their years of hard work, encouragement, and discipline. To both my elder sisters, Irine and Amy, thank you for your advice and support, and for paving the path before me to nearly every major milestone in life.

Finally I would like to express gratitude to my committee members, Dr. Nader Sadegh, and Dr. Jun Ueda, and also to Siemens Energy and Automation and Boeing Research and Technology (BR&T) for supporting my research.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

SUMMARY

Cranes occupy a crucial role within the industry. They are used throughout the world in thousands of shipping yards, construction sites, and warehouses. However, payload oscillation inherent to all cranes makes it challenging for human operators to manipulate payloads quickly, accurately, and safely. Manipulation difficulty is also increased by non-intuitive crane control interfaces. Intuitiveness is characterized by ease of learning, simplicity, and predictability. This thesis addresses the issue of intuitive crane control in two parts: the design of the interface, and the design of the controller.

Three novel types of crane control interface are presented. These interfaces allow an operator to drive a crane by moving his or her hand freely in space. These control interfaces are dependent on machine vision and radio-frequency-based technology.

The design of the controller based on empirical means is also discussed. Various control architectures were explored. It was concluded that a controller with an input shaper within a Proportional Derivative feedback loop produced the desirable crane response. The design of this controller is complemented with a structured design methodology based on root locus analysis and computer numerical methods.

The intuitive crane control systems were implemented on a 10-ton industrial bridge crane; simulation and experimental results are presented for validation purposes.

CHAPTER I

INTRODUCTION

Cranes play a key role in maintaining the economic vitality of modern-day industry. Their importance can be seen at shipyards, construction sites, warehouses, and in a wide variety of material-handling applications. The effectiveness of crane manipulation is an important contributor to industrial productivity, production efficiency, and workplace safety.

One inherent property of cranes that is detrimental to safe and efficient operation is the natural tendency for the payload to oscillate like a pendulum, double pendulum, or with even more complex oscillatory dynamics [44]. Significant effort has been made to develop control schemes to reduce the oscillatory response from both issued commands and external disturbances [44, 32, 42, 43, 34, 5, 21, 29, 15, 38, 39, 6, 1, 33, 27, 24]. Operators who manipulate a crane utilizing appropriate oscillation-suppression technology generate safer and more efficient crane motions than operators without such compensation [29, 12, 13, 14].

While significant strides have been made to improve the operational efficiency of cranes by controlling their dynamic oscillatory response, relatively little consideration has been given to the way in which operators issue those commands [40]. Most operators drive cranes with push-button control pendents, joysticks, or control levers, such as those shown in Figure 1. These interfaces are not intuitive to many operators because they must first establish a "mental map" from the actuation of a button/joystick/lever to the actual crane motion that will be generated. For example, the operator must know that pushing button "A" will make the crane travel in the *forward* direction. As the operator moves through the workspace and changes

(a) Control Pendent **(b) Joystick** **(c) Control Levers**

Fig 1. Typical Crane Control Interfaces (Photos Courtesy of Magnetek Inc.)

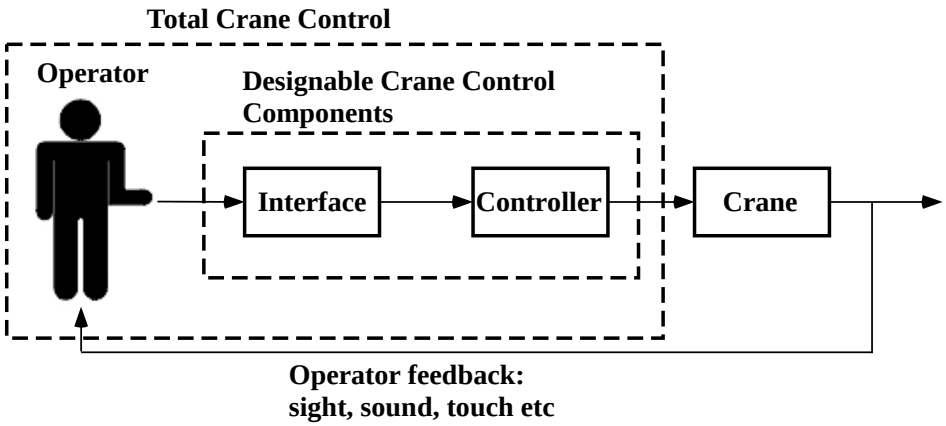


Fig 2. Crane Operator, Interface, and Controller

orientation, he or she must continuously update this "mental map". Furthermore, the interface is used to specify the motion of the overhead trolley, not the payload. Therefore, the operator must also account for the difference between the commanded motion of the trolley, which can be several meters overhead, and the delayed oscillatory response of the payload.

Figure 2 is a block diagram that illustrates the fundamental components of crane control. The operator commands the crane by manipulating some form of interface. Examples include pushing buttons on control pendents or pushing levers on a control box. The interface then interprets this interaction and issues a command signal to

the controller. Typically this signal is electrical, and could be in the form of analog AC, or digital high/low. The controller then processes this command signal using some type of algorithm, such as Proportional Integral Derivative (PID), ramp-up ramp-down, or a straight feed-through. The resulting command signal is then sent to the crane, where the motors are energized and the actuators are actuated to generate crane movement.

The crane will then behave according to the physical dynamics of the system - for example the trolley may respond like a sliding mass and the hook and payload may behave like a double pendulum. It is of interest to note that there exists a feedback loop. The behavior of the crane is observed (usually via sight) by the operator; the operator then reacts accordingly by interacting with the interface, thereby closing the loop. The components of *Total Crane Control*, indicated by the larger dashed box, are the operator, interface, and the controller. However, since we cannot "design" the operator, only a subset of this, the interface and the controller, is included in the smaller dashed box, which indicates the components that are designable by control engineers.

This thesis is organized around the two fundamental components of crane-control design: This chapter introduces the reader to the concept of intuitive crane control, and how this thesis addresses and implements this concept. Previous work from academia and industry are also discussed. Chapter 2 examines the interfaces developed for intuitive crane control. Chapter 3 describes the design of the controller component; theoretical and experimental results are also included. The thesis concludes in chapter 4.

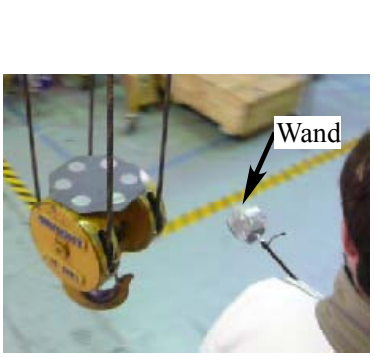
1.1 Intuitive Crane Control

As mentioned earlier, the design of crane control can be separated into two major components - the design of the interface, and the design of the controller. The interface should be intuitive to the crane operator. An "intuitive" control system is hard to define rigorously, but it is often characterized by:

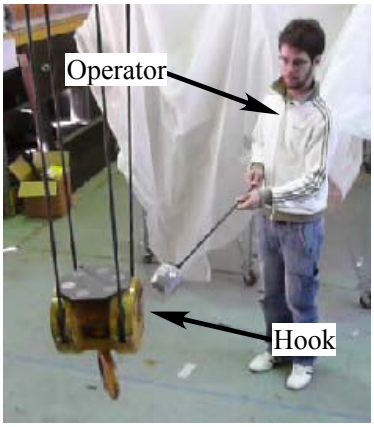
- Ease of Learning - The ease of which a novice operator is able to learn to use the interface (an interface with a fast learning curve typically has a form of interaction that the operator is already familiar with).
- Simplicity - The absence of a complex "mental map" that translates the interaction with the interface to crane motion.
- Predictability - The crane motion responds in a repeatable and predictable way to operator commands.

This thesis presents three novel interfaces that allow an operator to drive a crane simply by moving a hand-held device in 3-D space. This type of hand-motion interface addresses the issue of intuitiveness because it easy to learn, as gesturing with the hand to convey an intended motion is familiar to most people. Additionally, this interface is simple, as it does not have a complex "mental map" between the interface and the crane response. This is virtually a seamless "interface-less" interface, as the "mental map" for this interface is simply a direct feed-through. For example, there is no transformation mapping from "pushing button A" to "*forward* crane motion". Instead, the mapping is a direct feed through; when the operator moves the hand-held device to a certain position, the crane will respond by moving to the same position.

The issue of generating predictable crane behavior is largely handled by the second fundamental component of crane-control design - the controller block in Figure 2. The oscillatory nature of cranes makes the behavior hard to predict for a novice operator.



(a) Wand



(b) Operator with Wand Control

Fig 3. Wand Control

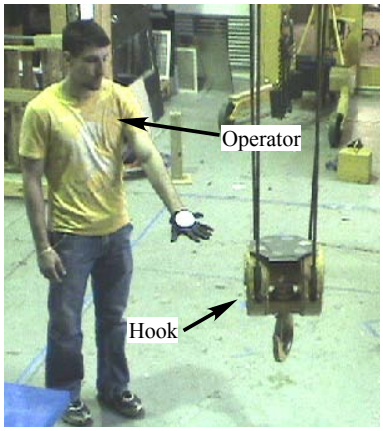
However, by implementing a technique known as input shaping (discussed in more detail in section 3.3.1), the oscillatory dynamics are greatly reduced and the crane behavior becomes similar to a rigid body with a time lag. For a human operator, rigid body behavior is much easier to predict than oscillatory behavior. The result of an intuitive interface is that it reduces the manual dexterity required for safe and efficient operation.

The three intuitive interfaces presented in this thesis are:

1. Wand control - Shown in Figure 3. The wand is a retro-reflective ball mounted to the end of a hand-held pole. A machine vision system is used to determine the position of the wand in real-time. The position of the wand is then used as the command signal to drive the crane.

2. Glove control - Shown in Figure 4. The glove is monotonically black, which is contrasted with a circular retro-reflective marker attached to the top-side of the glove. A machine vision system is used to determine the position of the glove in real-time. The position of the glove is then used as the command signal to drive the crane.

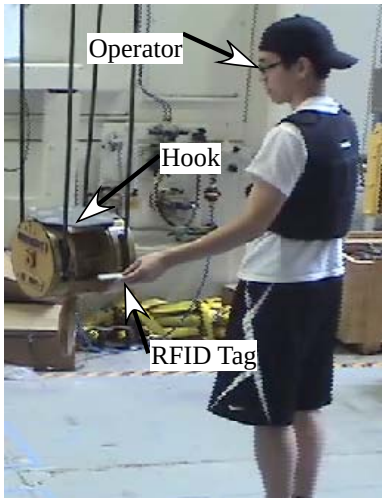
(a) Glove



(b) Operator with Glove Control

Fig 4. Glove Control

(a) RFID Tag



(b) Operator with RF-Based Control

Fig 5. Radio-Frequency-Based Control

3. Radio-Frequency(RF) based control - Shown in Figure 5. A Real-Time-Location-System (RTLS) based on Radio-Frequency Identification (RFID) technology is used to track the position of a small tag held in the operator's hand. The position of the tag is then used as the command signal to drive the crane.

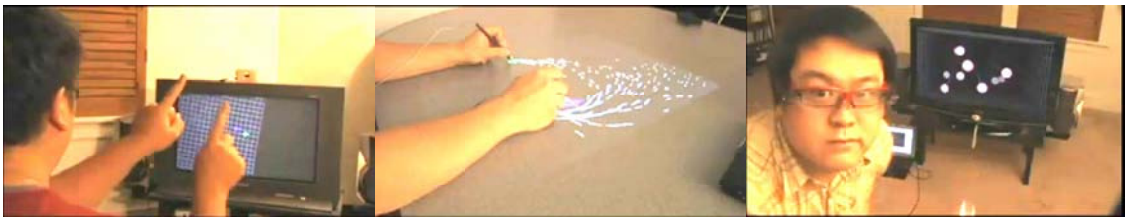
In this thesis, the wand, the glove, and the RFID tag are also known by the collective term "hand-held device".

1.2 Previous Work

In the field of Human Computer Interaction (HCI) research, there has been great interest in the area of computer User Interfaces (UI), and how people interact with digital media. In many ways, the design of an intuitive crane-control interface parallels the design of UI, in that the goals of the interface are the same: easy to learn, simple, and predictable. The designs of UI are often similar to the interfaces described in this thesis: seamlessly "interface-less" interface that leans heavily on the physical, tangible, and manipulable world (for example, grasping an apple from the table); as opposed to relying on the user to understand the concept of abstraction that is commonly seen in Graphical User Interfaces (GUI) today (for example, the use of toolbar buttons to represent various software functionalities).

Johnny Lee of Carnegie Mellon University first gained widespread fame on the internet for his videos on "hacking" the Wiimote for low-cost UI's. The Wiimote is a video game controller that comes standard with the Nintendo Wii video game console. He utilized the infrared (IR) sensors (which are capable of detecting multiple targets simultaneously) in the Wiimote to track the movement of fingers, which are illuminated by an IR-LED array and reflective material attached to the fingers. This idea is used for a low-budget implementation of a multi-"touch" user interface, as shown in Figure 6a. The same concept can be used to create an interactive whiteboard using an IR pen and a projector, as shown in Figure 6b. Similarly, IR-LED's can be mounted to a pair of glasses to track the movement of the head, as shown in Figure 6c. The display can be modified according to head movement and is a low-cost alternative to expensive Virtual Reality (VR) systems.

Lee's research follows a similar theme of using cameras, projectors, light pens, and light sensors to create foldable, movable, and interactive displays on surfaces [18, 19, 17, 20]. Interestingly, in Lee's research group, Raskar et al. combined RFID tags and photo sensors to project information in the physical world so that objects



(a) Finger Tracking **(b) Interactive Surface Whiteboard** **(c) VR Head tracking**

Fig 6. Johnny Lee's Wiimote Projects(Photos courtesy of johnnylee.net)

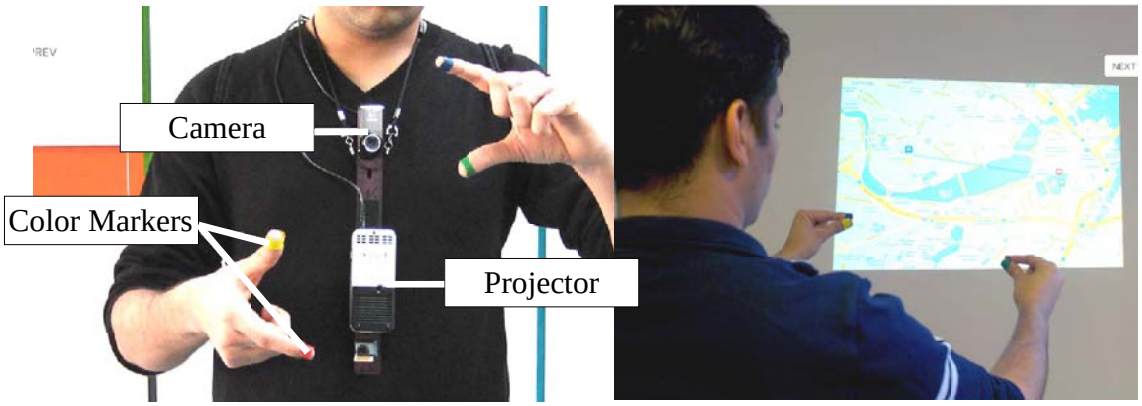


Fig 7. Sixth Sense - A Wearable Gestural Interface (Photos courtesy of Pranav Mistry)

become self-describing and interactive with the user [31].

The Sixth Sense, shown in Figure 7 and created by Maes et al., is a wearable gesture-based interface consisting of a camera, a projector, and colored markers worn on the fingers. Images are displayed on any stable surface by the projector. The user is able to interact with the image using finger gestures, which the camera is able to detect due to the colored markers [26].

In similar work, Jeffrey Han focused on implementing low-cost solutions to large-scale, multi-touch, rear-projection displays [4]. Multi-touch displays are display surfaces whereby user(s) are able to manipulate the displayed objects by touching the screen in a gestural way. For example, rotating an object can be accomplished by moving one finger in a circular motion around another finger.

Fig 8. Siftables(Photos courtesy of David Merrill)

The "Siftables", created by Merrill et al. [25] and shown in Figure 8, are displays mounted inside small plastic blocks that have numerous sensors. They also have networking capabilities that allow them to communicate with other Siftables. It makes for a more intuitive interface that exploits human's natural skill of manipulating objects with their hands to effortlessly sift and sort information. For example, a flow chart can be created by arranging Siftables in a geometric pattern; to add more color to a picture, one can emulate pouring paint into a container by tilting one Siftable that represents color, into a neighboring Siftable that represents the picture.

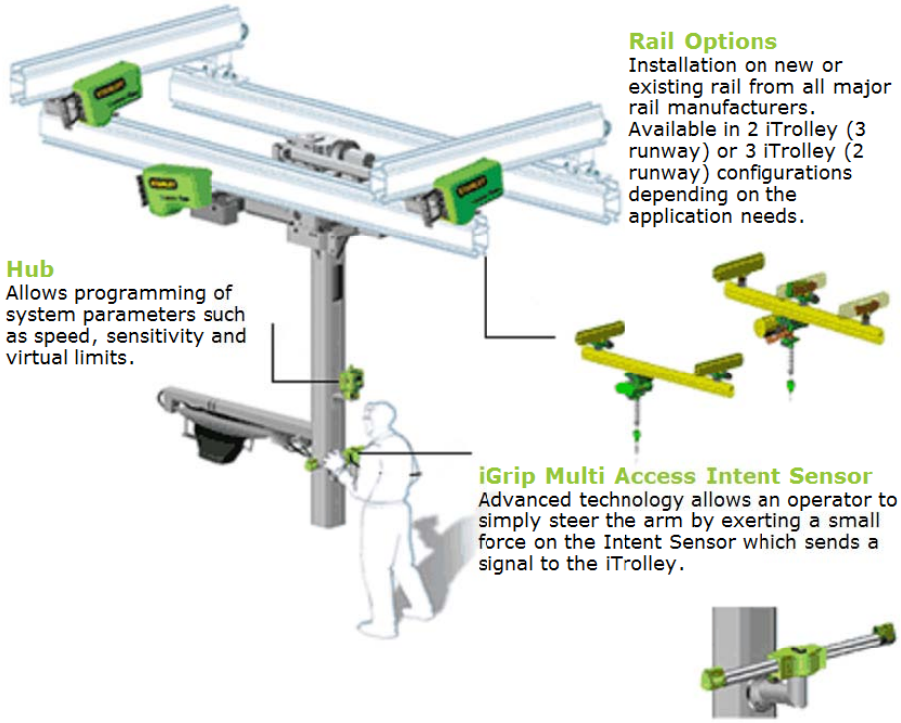
There is also related research that focused on industrial applications. Kazerooni et al. [9, 11] championed "Extenders", in which both a human and a robot apply significant force to a payload, with the robot amplifying the human effort, much like the way power steering amplifies the steering effort exerted by a driver. A working prototype of an arm-based extender was built for the application of material handling. The "Magic Glove" [10], is a concept that is similar to the glove control interface discussed in this thesis. A key difference is that the magic glove utilizes pressure sensors to measure the amount of force applied to a payload. This information is then

sent as an RF signal to actuators that provide a proportionate amount of assistance to the wearer.

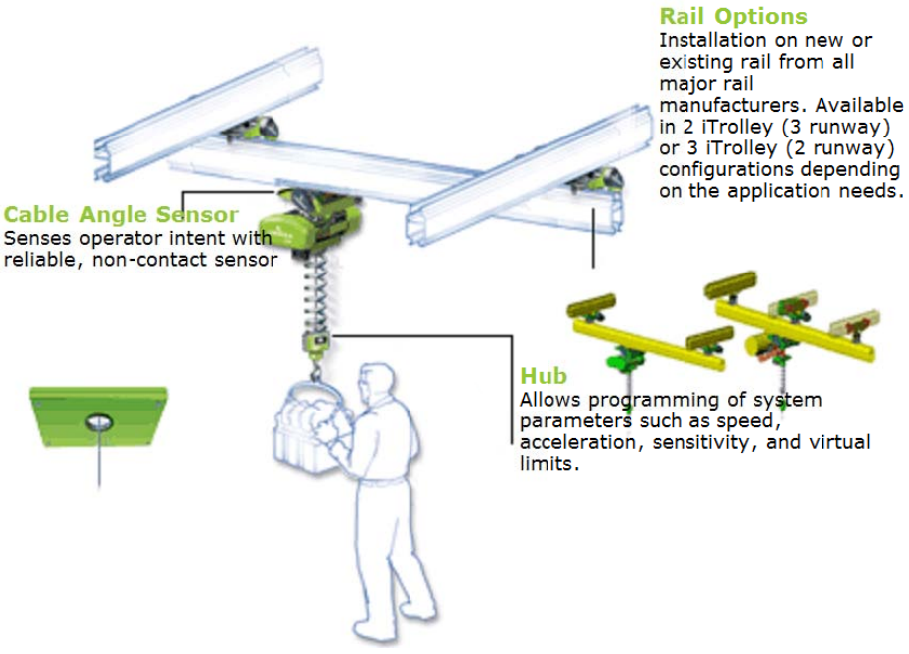
The Laboratory for Intelligent Mechanical Systems (LIMS) at Northwestern University specialize in Intelligent Assist Devices (IAD's) and Collaborative Robots, or Cobots [2, 23, 30]. IAD's are similar to extenders in that they sense human-intent and then provide assisting forces to help the operator manipulate a payload. Cobots are designed as material-handling robots that also have the capability to provide 1-D frictionless virtual walls, so that operators are constrained to move the payload along a programmable path. This was implemented in a car assembly plant where payloads must be orientated and moved through a complex trajectory to their final position.

Several commercial companies focus on ergonomic and intuitive material handling solutions. Cobotics Inc. (acquired by Stanley Assembly Technologies in 2002), was a start up company that specialized in IAD's and spun off from the LIMS group at Northwestern University. The iTrolley product range by Stanley Assembly Technologies is based on the concept of IAD's. The rigid-arm variant, shown in Figure 9a, senses operator intent with a pressure-sensitive grip. The wire-rope variant, shown in Figure 9b, uses angle sensors on the wire/rope to detect operator intent when he or she pushes on the suspended payload.

Stanley Assembly Technologies also manufacture the iLift product range, which is a type of hoist, or air-balancer. Gorbel also manufactures air-balancers in a product line they call "Intelligent Lifting Devices", shown in Figure 10. Air-balancers are a variant of cranes in that only the vertical direction (hoisting and lowering) are motorized (motors are housed inside the trolley). Lateral movement is provided by the operator pushing on the payload (which is attached to the hook) directly, as the trolley is free-wheeled on the overhead tracks. In this sense, the movement of the payload is intuitive as there is no button pushing that is translated into movement. The operator raises or lowers the control grip to hoist or lower the payload. Payloads



(a) Rigid-Arm based



(b) Wire-Rope based

Fig 9. iTrolley IAD(Pictures courtesy of Stanley Assembly Technologies)

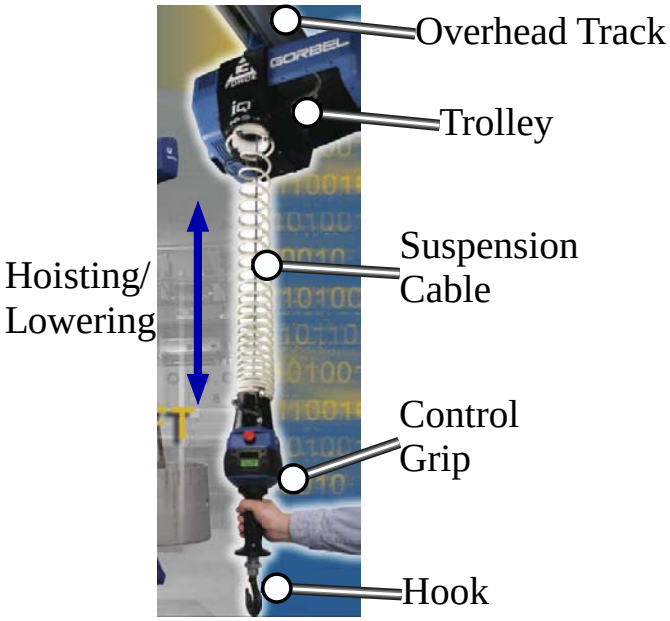


Fig 10. Gorbel Intelligent Lifting Device(Picture courtesy of Gorbel)

are typically in the order of a few hundred kilograms, and applications are often in small areas where operators are required to lift loads repetitively. Air-balancers also have a float-mode feature, where the motors exert just enough lifting force to offset the weight of the payload. The operator can apply small amounts of additional force in the vertical direction so that payloads can be precisely orientated and positioned.

Several consumer products also focus on intuitive user interfaces. The work in this thesis, compared to traditional crane control interfaces, is analogous to the comparison between the Nintendo Wiimote and traditional video game controllers. Traditional video game controllers are typically hand-held and contain numerous joysticks and buttons. A mapping exists that transform an interaction with the controller, to an action in the video game; for example, "push button A" corresponds to "swing the baseball bat". The Wiimote, shown in Figure 11a, is the hand-held controller of the popular Nintendo Wii video game console. It contains IR and inertial sensors that detect the movement of the player's hand. Using this device, the mapping from interface to video game action is virtually reduced to a straight feed-through. Players

(a) Nintendo Wii
(Photos courtesy of Nintendo Inc.)

(b) Apple iPhone
(Photo courtesy of Apple Inc.)

Fig 11. Intuitive Interfaces from Consumer Products

physically "swing the Wiimote" to exact an action of "swinging the baseball bat" in the context of the game. The Apple iPhone, shown in Figure 11b, is an example of intuitive interface for mobile phones. Rather than traditional button keypads, the iPhone utilizes a touch screen with multi-touch capabilities. Certain tasks become easier to use, such as zooming in and out of a picture by simply "pinching" the screen using two fingers.

This chapter introduced the reader to the concept of intuitive crane control. Previous work related to the concepts described in this thesis were also discussed. The next chapter discusses the first component in the design of crane control - the interface between the operator and the controller.

CHAPTER II

MACHINE VISION AND RADIO-FREQUENCY-BASED INTERFACES

The intuitive crane control interfaces presented in this thesis rely heavily on the use of sensors to detect operator commands. Of the three control interfaces discussed in section 1.1, machine vision is critical for two: the wand and glove control. RF-based crane control relies on the use of RF sensors. The fundamental role of these interfaces is to acquire the position of the hand-held device, which is then used as the command signal to drive the crane.

The goal of this chapter is to first present the hardware used for the experimental setup. Then, to comprehensively describe the hardware, software, and algorithms of the machine vision module. The chapter concludes with a description of the interface for the RF-based crane control.

2.1 Crane Hardware

A 10-ton bridge crane was used as the experimental test-bed for the work detailed in this thesis. A representation of a typical bridge crane is shown in Figure 12. Bridge cranes are load-lifting systems consisting of four main components: a hoist cable and hook, a trolley, a bridge, and a runway. The payload is attached to the hook, which is suspended from the trolley by the hoist cable. The trolley is the load-lifting component and moves on (and parallel to) a beam called the bridge. The bridge moves on (and parallel to) a stationary runway. The runway is usually comprised of two supporting members that are permanently affixed to a structure, such as the walls of a warehouse. The bridge and the runway are oriented orthogonally. The combination

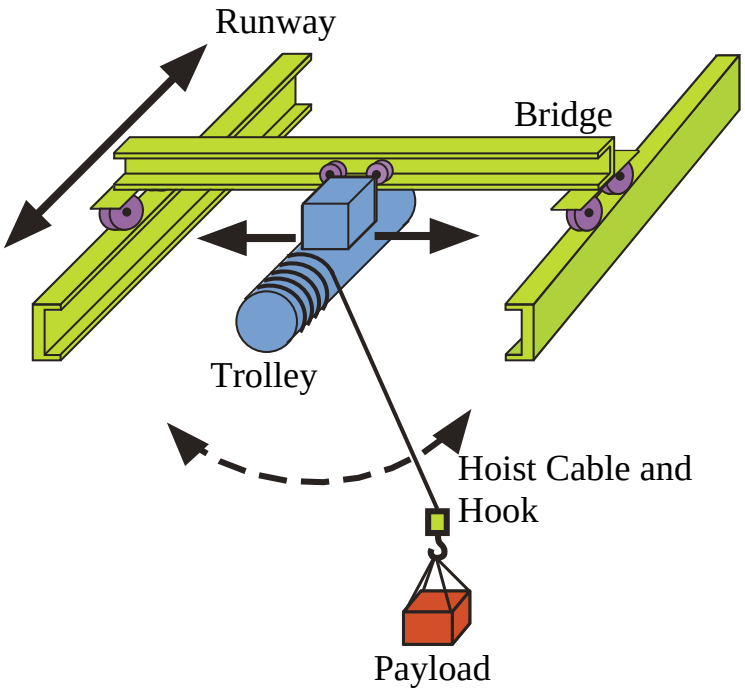


Fig 12. Typical Bridge Crane

of the degrees of freedom along the bridge and the runway, and the hoisting capability of the trolley, provide a large three-dimensional workspace reachable by the payload.

The 10-ton bridge crane at the Georgia Institute of Technology is shown in Figure 13. The bridge and runway span a 5.5 m by 40 m area. The trolley is mounted to the bridge, which is suspended above the ground at a height of 6.1 meters.

A schematic of the hardware components of the 10-ton crane are shown in Figure 14. The crane uses two 460-volt AC induction motors to drive the bridge and one to drive the trolley. Two Siemens Vector Masterdrives control the bridge and trolley motors. The drives are programmed to track a velocity reference signal sent as an analog voltage from a Siemens CPU 314C-2DP Programmable Logic Controller (PLC). Under manual operation, the PLC generates these reference signals when an operator depresses any of the four directional (forward, reverse, left, or right) buttons on the control pendent. Hoisting of the hook, however, is controlled by on/off relays. Under advanced-control operation, for example when input shaping is turned on, the

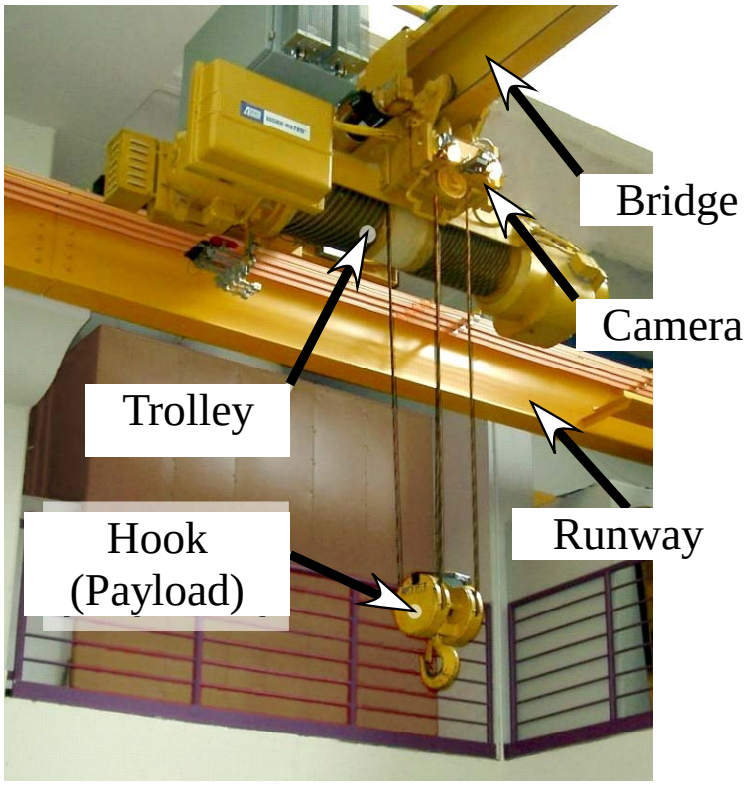


Fig 13. 10-Ton Bridge Crane in the Georgia Tech MaRC building

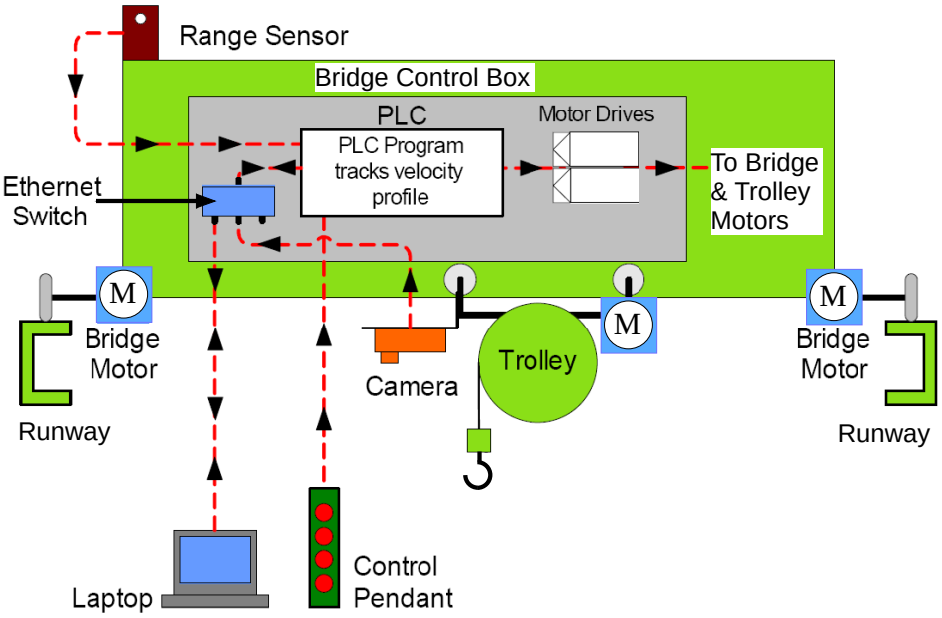


Fig 14. Hardware Configuration of the 10-ton bridge crane

PLC intercepts the signals from the control pendent to generate the shaped signals, which are then sent to the motor drives.

Position measurements of the trolley location are provided by two Banner LT3 Long-Range laser sensors along the bridge and trolley axes. The range sensor emits light pulses in the measurement direction. A stationary retro-reflecting surface reflects the emitted pulses back to the sensor, where they are processed to determine the absolute distance. Hook position measurement is provided by a Siemens Simatic VS723-2 intelligent camera mounted on the center of the trolley. The camera points downwards at the hook and provides a workspace field of view of approximately 3m by 3m (from a height of 6m above the floor). The PLC is the central component that communicates with all other modules of the crane - motor drives, camera, laser sensors, control pendent, and the Siemens PG field laptop. Communication is established via a Siemens network module that uses the Industrial Ethernet protocol (Profinet). Both wired and wireless forms of communication are available. The PLC has a cycle rate of 35 ms.

2.2 Machine Vision Hardware

As mentioned earlier, a Siemens Simatic VS723-2 intelligent camera mounted on the trolley is used to determine the position of the suspended hook, as shown in Figure 15. To improve the quality of the acquired images, a LED light array was installed next to the camera to illuminate the workspace, as shown by the left picture in Figure 16. A close-up view of the VS723-2 is shown on the right of Figure 16. Figure 17 shows a board with retro-reflective material arranged in a special pattern that is attached to the top of the hook. This assists image processing algorithms to track the hook. The reflective fiduciary markers enable the camera to easily segment desired regions from the background, and algorithms based on known geometry (explained in section 2.4.1) are used to calculate the position of the hook center.

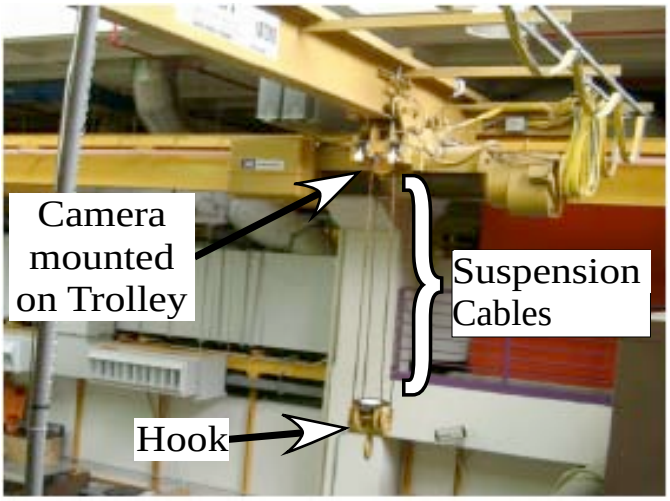


Fig 15. Trolley Mounted Camera

Fig 16. Trolley Mounted Camera - Closeup View
(Photo on the Right Courtesy of Siemens)

The Siemens Intelligent Camera VS723-2 is a complete image-processing system that consists of the light-capturing components, onboard processor, and memory, in one self-contained unit. The unit is compact, with dimensions of 60mm x 112mm x 30mm. It has a gray-scale 1/3" CCD array with a resolution of 1024 x 768 pixels. The onboard processor is a Hitachi SH4 clocked at 200MHz and capable of 360 million instructions per second. Additionally, 8 MB of non-volatile Flash and 32 MB of volatile DRAM memory are available. Communication is available using a standard ethernet RJ45 port and 8 configurable digital input/outputs; power is provided by an RJ-45 port with a rated voltage of 24V DC.

Fig 17. Reflective Material on the Hook

2.3 Machine Vision Software - Spectation

Spectation is the Siemens software used to communicate with Siemens intelligent cameras. The hierarchical structure of a typical project in Spectation is shown in Figure 18. The organization is divided into four layers (labeled by the numbers on the right in Figure 18), and each layer handles a different aspect of the entire image-processing algorithm. The four layers are:

1. System Layer - This layer includes user-specifiable constant parameters concerning the direct function of the camera, such as exposure time and trigger source.

2. Inspection Layer - This layer contains inspection programs that are used to analyze an image and return an inspection result. An inspection starts automatically after a image is acquired or can be called from a background script.

3. Softsensor Layer - This layer contains the two main types of tools required to analyze an image - soft sensors and foreground scripts. Soft sensors are application-specific, graphical-based tools that provide commonly-used, basic

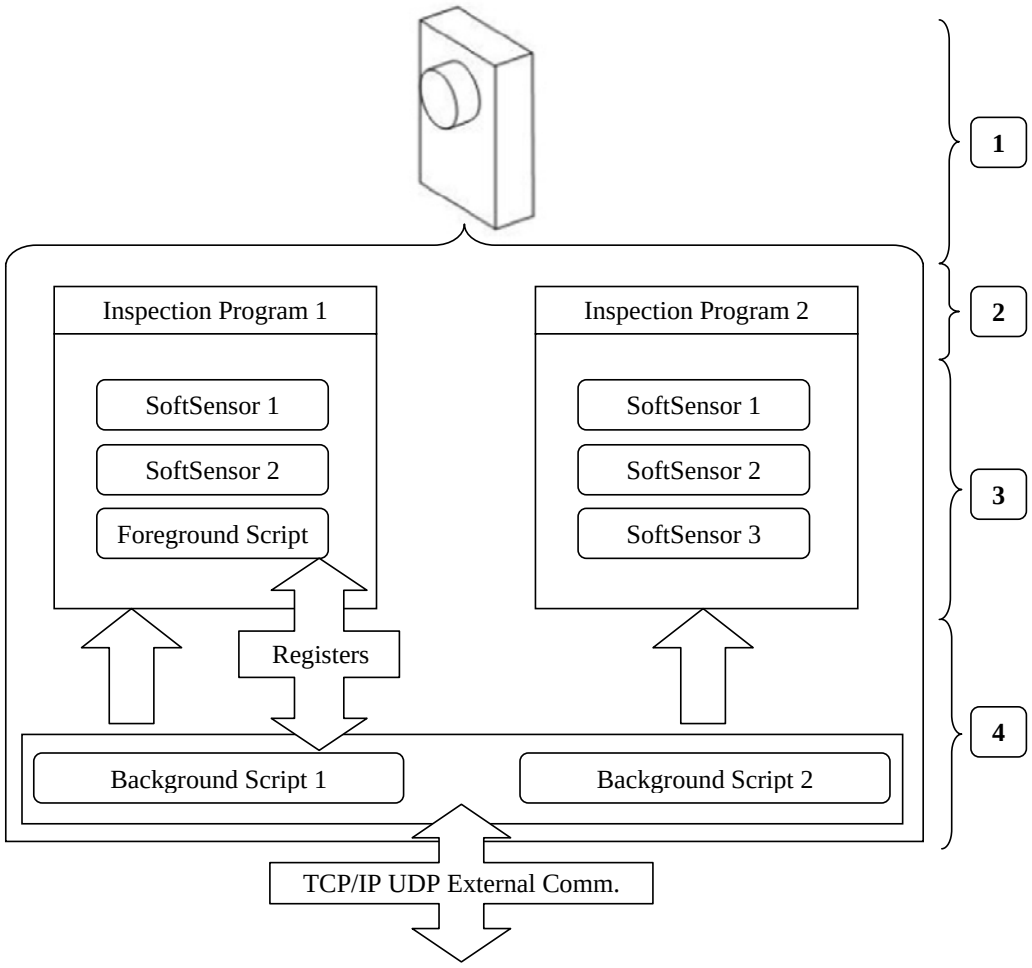


Fig 18. Structure of a Spectation Project

image-processing capabilities. As each image is a 2-D array of pixel intensity values; basic image processes involve the manipulation of this array. For example, these include the detection of blobs, ridges, edges, etc. Foreground scripts allow the user the flexibility to construct more complex inspection programs. A typical use for foreground scripts is to take the results from multiple soft sensors, perform mathematical operations on the results, and communicate with the background script.

4. Background Layer - This layer contains background scripts, which serve in a supervisory role to handle digital I/Os, call inspection programs and modify their parameters, and manage communications. Communication with external

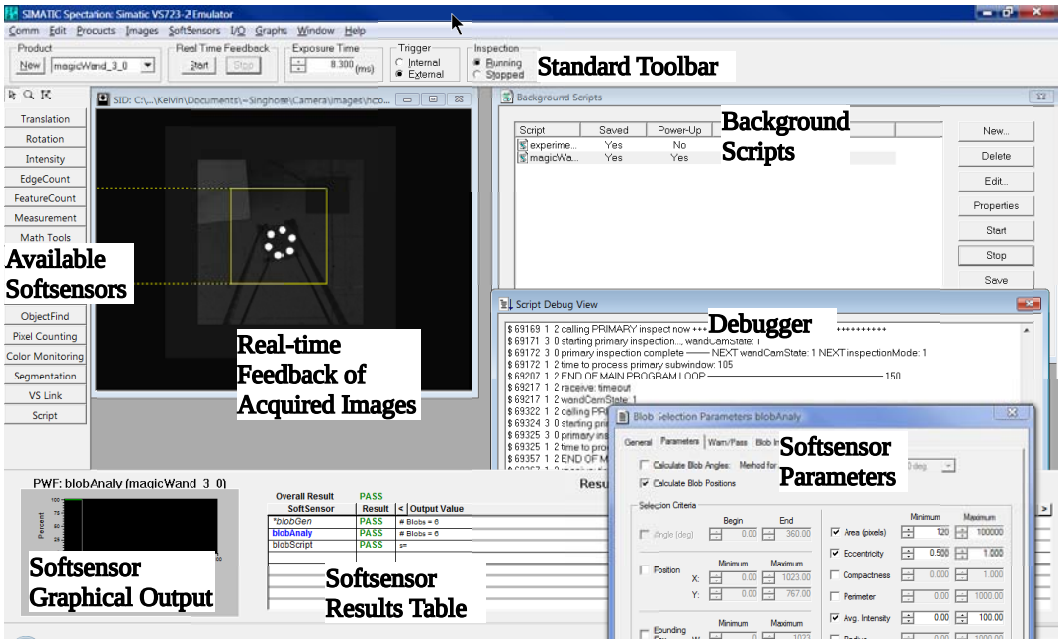


Fig 19. Spectation Worksapce

devices such as a PLC or a computer can be established using either TCP (Transmission Control Protocol) or UDP (User Datagram Protocol). Data can only be exchanged between the background script and an inspection program via memory registers, which act as global variables.

Figure 19 shows the GUI of the Spectation software. It contains a standard toolbar for commonly used software features, as well as an option to choose the active background script stored in memory. There are buttons for selecting softsensors and a window for adjusting softsensor parameters. Images acquired by the camera can be displayed in real-time, along with text-based real-time debugging information. Finally, the results from inspection programs can be returned in tabular or graphical form.

The two soft sensors that are relevant to this thesis are the "blob generator" and the "blob selector". As mentioned earlier, soft sensors provide basic image-processing tools that typically perform some operation on an image. A "blob" is a region of interconnected pixels that have similar intensity values. The "blob generator", shown

Fig 20. Spectation Softsensor Blob Generator - Detects and "Generates" Blobs

in Figure 20, is a softsensor that detects and generates these blobs based on user-specified conditions. Typical conditions include minimum average blob intensity or minimum blob size. Once the blobs are generated, their information are passed to the "blob selector", which selects blobs of interest by imposing more stringent conditions. Examples include blob eccentricity, position, or angle. In Figure 20, the square depicts the area of the image over which the blob generator is active. The regions inside the square represent blobs that were generated by the blob generator.

2.4 Machine Vision Algorithms

The machine vision system used in this research has two primary goals:

1. To track the position of the hook (relative to the trolley mounted camera).

2. To acquire the operator command signal (for wand and glove control). This signal is given by the position of the wand or glove.

The camera must be able to accomplish both goals in real-time and send the information to the PLC controller. This section will discuss the algorithms that were implemented to achieve the above goals.

2.4.1 Determining the Hook Position

Previous work [3] focused on tracking the position of the hook in a robust manner, and is briefly summarized here. A fundamental first step of tracking with machine vision is to highlight the target in such a way that it can be easily distinguished from the background and noise. One simple way to achieve this is with the use of retro-reflective fiduciary markers attached to the target. A light source illuminates the fiduciary marker thereby making it appear as a bright spot (or a blob with high pixel intensities) to the camera. This procedure allows the camera to easily isolate and track the target, without the need for complex computing algorithms.

Early versions of the fiduciary marker designs had simple geometries consisting of only a few markers. One such example is shown on the left column of Figure 21. The upper left picture shows the physical view of looking down at the hook from the trolley. The lower left picture shows the image as captured by the camera. The problem with using only a few markers was that it was susceptible to spurious sources such as other reflective surfaces around the workspace. Reliable Operation during day time was also marred by reflected sunlight and other ambient sources of lighting. These lights appear as regions of high pixel intensities to the camera. The problems were exacerbated by the fact that the suspension cables would obstruct the view of the fiduciary markers when the hook was swinging.

To improve the robustness of hook tracking, the number of fiduciary markers was increased to six. The markers were arranged in a hexagonal pattern, as shown in the right column of Figure 21. Again, the top right picture shows the physical view while the lower right picture shows the image as captured by the camera. The centers of the six circular markers coincide with the vertices of a regular hexagon, and therefore all lie on the same circumcircle, as shown in Figure 22. The circumcenter, labeled as as $[x_c, y_c]$ in Figure 22, can be determined from the positions of a subset of the six markers, with the minimum number in the subset being three. This subset is shown

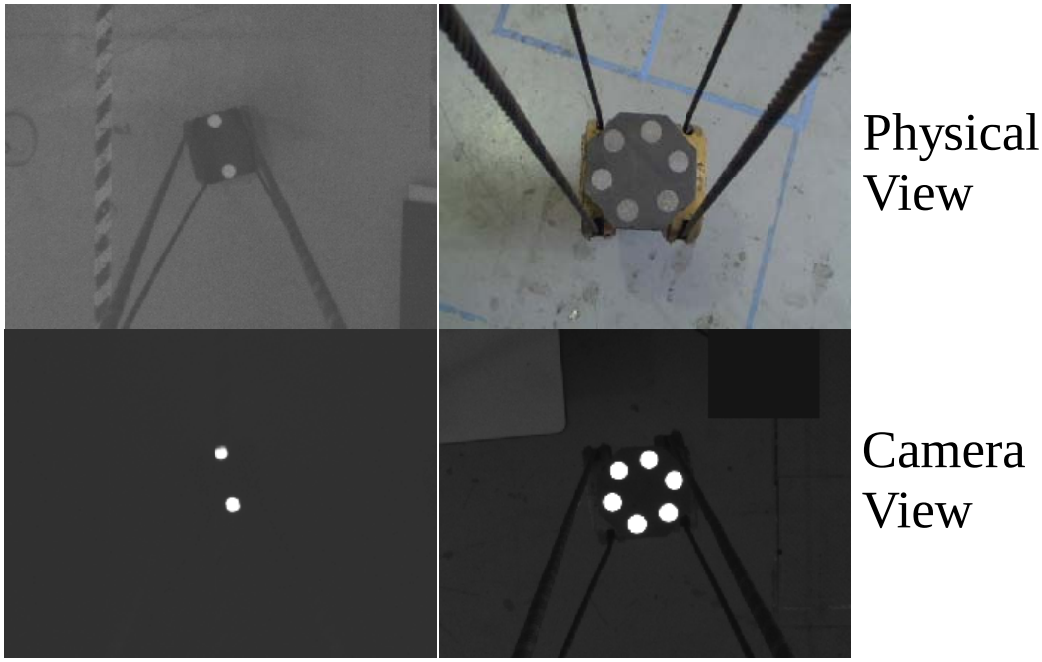


Fig 21. Hook Tracker Fiduciary Marker Designs

by the shaded circles in Figure 22, labeled Blob 1, Blob 2, and Blob 3. Picking any two pairs from the minimal subset of three markers (now referred to as blobs), the intersection of their perpendicular bisectors (labeled as B_1 and B_2 in Figure 22) is the circumcenter. The equations of the perpendicular bisectors joining the centers of these blobs, $[x_1, y_1]$ to $[x_2, y_2]$, and $[x_2, y_2]$ to $[x_3, y_3]$ are given by:

$$(y_1 - y_2)\left(X - \frac{1}{2}(x_1 + x_2)\right) - (x_1 - x_2)\left(Y - \frac{1}{2}(y_1 + y_2)\right) = 0 \quad (1)$$

$$(y_2 - y_3)\left(X - \frac{1}{2}(x_2 + x_3)\right) - (x_2 - x_3)\left(Y - \frac{1}{2}(y_2 + y_3)\right) = 0 \quad (2)$$

where X and Y are variables. Then, two linear equations with two unknowns can be formed:

$$\begin{bmatrix} (y_1 - y_2) & -(x_1 - x_2) \\ (y_2 - y_3) & -(x_2 - x_3) \end{bmatrix} \begin{bmatrix} X \\ Y \end{bmatrix} = \begin{bmatrix} \frac{1}{2}(y_1 - y_2)(x_1 + x_2) - \frac{1}{2}(x_1 - x_2)(y_1 + y_2) \\ \frac{1}{2}(y_2 - y_3)(x_2 + x_3) - \frac{1}{2}(x_2 - x_3)(y_2 + y_3) \end{bmatrix} \quad (3)$$

Using Cramer's rule we can solve for the coordinates of X and Y that satisfy

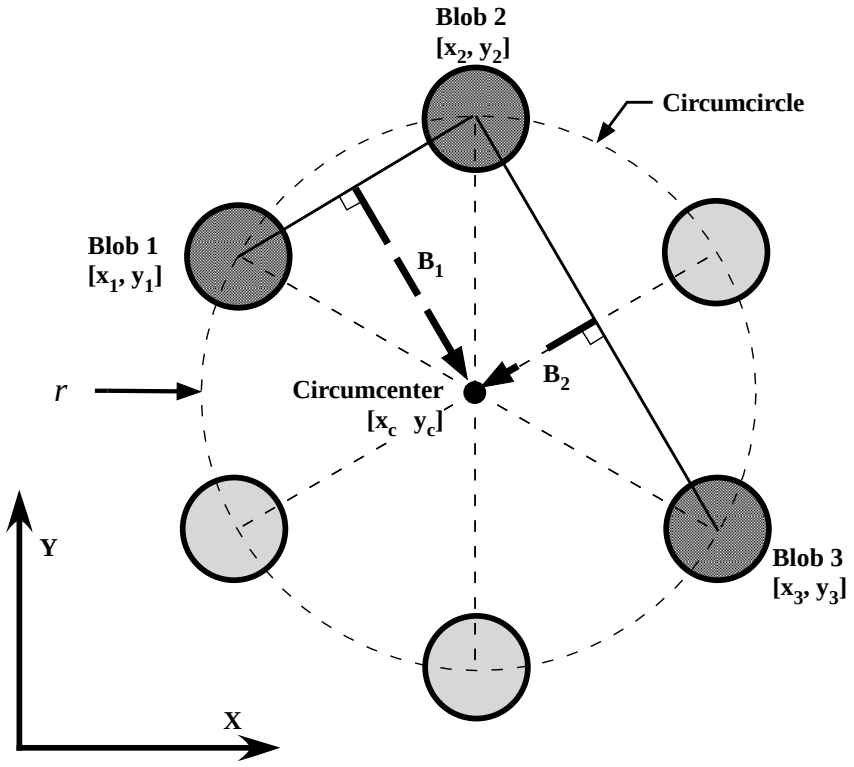


Fig 22. Hook Tracking Fiduciary Markers Arranged in a Hexagonal Pattern

both equations simultaneously. These coordinates are the same as the circumcenter, $[x_c, y_c]$, and is the intersection of the two bisectors:

$$x_c = \frac{\begin{array}{c} -[\frac{1}{2}(y_1 - y_2)(x_1 + x_2) - \frac{1}{2}(x_1 - x_2)(y_1 + y_2)](x_2 - x_3) \\ +[\frac{1}{2}(y_2 - y_3)(x_2 + x_3) - \frac{1}{2}(x_2 - x_3)(y_2 + y_3)](x_1 - x_2) \end{array}}{-(y_1 - y_2)(x_2 - x_3) + (y_2 - y_3)(x_1 - x_2)} \tag{4}$$

$$y_c = \frac{\begin{array}{c} -[\frac{1}{2}(y_1 - y_2)(x_1 + x_2) - \frac{1}{2}(x_1 - x_2)(y_1 + y_2)](y_2 - y_3) \\ +[\frac{1}{2}(y_2 - y_3)(x_2 + x_3) - \frac{1}{2}(x_2 - x_3)(y_2 + y_3)](y_1 - y_2) \end{array}}{-(y_1 - y_2)(x_2 - x_3) + (y_2 - y_3)(x_1 - x_2)} \tag{5}$$

The implemented algorithm calculates an average circumcenter using all possible combinations of three-blob subsets. The robustness of this algorithm comes from the

fact that any three blobs can be used to calculate the circumcenter. Considering that there are six blobs available, there is a 100% redundancy factor. This redundancy has two primary benefits:

1. The hook can still be tracked even if some of the blobs are obstructed from view by the suspension cables or other obstacles.

2. The effect of spurious blobs are mitigated, but not eliminated, as the hook position is calculated from the average of all circumcenters. It is expected that at any one time there would only be one or two spurious blobs, which are outweighed by the six true, hexagonally arranged blobs.

Due to the fact that the markers appear larger when the hook is raised closer to the camera, a relationship can be formed relating the hook height to the size of the circumcircle radius, labeled as r in Figure 22. Knowing the height of the hook, the cable length can be determined, which yields important information such as the natural frequency of the pendulum-like crane system.

2.5 Discerning the Hook Blobs from the Wand/Glove Blob

In the above sections, previous work on the camera was described. For the remainder of this chapter, we discuss the contributions of this thesis.

Under wand and glove control, the camera must not only track the position of the hook, but also concurrently track the position of the wand/glove. As shown in Figures 3 and 4, the purpose of the hand-held device is to provide an input signal to the control system based on the operator's hand motion. The hand-held device is essentially a fiduciary marker that is designed to be easily manipulated in 3-D space by a human operator.

There are three different types of images that are captured by the camera. Figure 23(a) shows the case when the hand-held device is not present. Figure 23(b) shows the

Fig 23. Camera Views During Wand Operation

case when the hand-held device (in this case, the wand) is present, and Figure 23(c) is when the hand-held device is present but partially occluded by the suspension cables. As can be seen from the images, the blob resulting from the wand is similar to the blobs resulting from the fiduciary markers used for hook tracking. Apart from some minor differences such as blob area and average intensity value, there is not a single distinguishing characteristic that can segregate the hook blobs from the wand/glove blob. Therefore, there is a need to find a robust method to discern between the two types of blobs.

A well known technique called the K-means clustering algorithm [22] was selected for the task of discernment. This technique is based purely on the position of the blobs. Effectively it assumes all blobs are identical, despite the fact that this is not strictly true. As mentioned above, while it is possible to discern the wand/glove blobs from the hook blobs based on size and average pixel intensity, these variables are not ideal for robust discernment because during crane operation, these variables undergo large fluctuations. For example, the area of the wand/glove blob varies depending on the vertical elevation of the wand/glove. The average pixel intensity values will vary greatly depending on the angle of illumination on reflective surfaces by the trolley mounted LED lights.

Thus, if discernment is to be purely based on the blob positions, it effectively

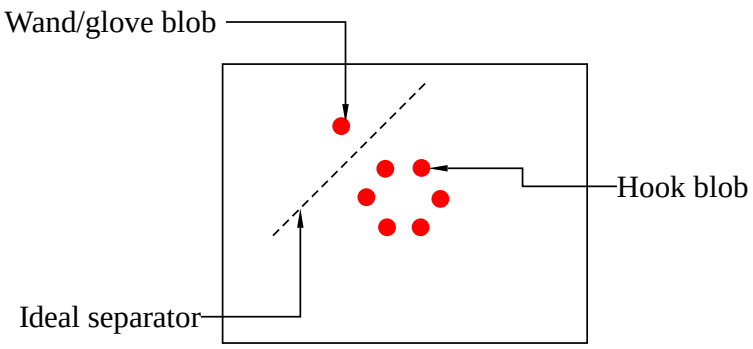
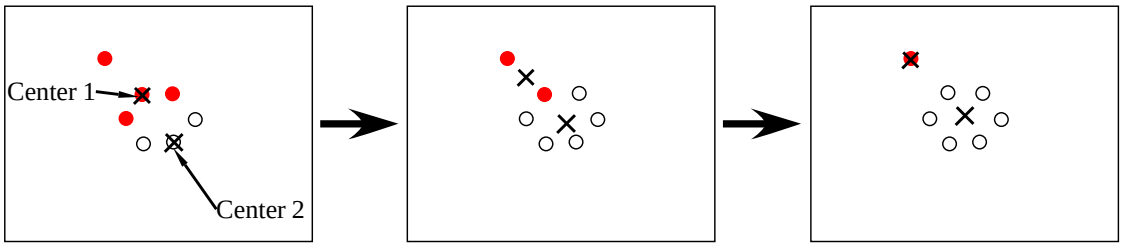


Fig 24. Ideal Blobs Discernment

becomes a cluster grouping problem, where the goal is to group blobs together based on their relative distance to other blobs. Figure 24 shows the ideal discernment where the wand/glove blob is separated (represented by the dashed line) from the hook blobs. Notice that all blobs are identical. The only distinguishing factor is the position of each blob.

The K-means clustering algorithm is depicted in Figure 25. As we are only searching for two clusters of blobs (the wand/glove, and hook blobs), K is set to two, so the algorithm will produce two cluster groupings. The algorithm starts by randomly picking K blobs as the initial coordinates of cluster centers, depicted as crosses in Figure 25(a). Then, the algorithm starts the iterative process where it calculates the Euclidean distance between each blob and each cluster center, and assigns a cluster label to each blob corresponding to whichever cluster center it is closest to. Cluster 1 blobs are solid and cluster 2 blobs are white in Figure 25(b). The coordinates of new cluster centers are then calculated using the mean X and Y coordinates from all the blobs in that cluster. Following this, the algorithm reiterates. The algorithm stops when the maximum number of iterations is reached or when the cluster centers stop moving. Even without the maximum iteration stopping condition, the K-means algorithm is guaranteed to terminate in finite time.

The camera will report successful detection of wand/glove if and only if one of



(a) Randomly select initial cluster centers and assign cluster grouping

(b) Iteratively calculate new cluster centers and reassign

(c) End when cluster centers stop moving

Fig 25. K-means Clustering Algorithm

the cluster contain only one blob, as shown in Figure 25(c). This condition will be satisfied if the lone blob is beyond the minimum distance from the main cluster of blobs. The minimum distance is given by r, the radius of the circumcircle in Figure 22. The pseudo-code for the general K-means blob algorithm is:

```
Choose K
Randomly choose K blobs and assign their centers
as the cluster centers, C_j.
Loop
    Partition the blobs into K clusters, G_j, according to the
    C_j that they are closest to.
    For each G_j, compute the mean of its elements and assign
    it as the new cluster center C_j.
Stop when cluster centers stop moving.
```

The simplicity of this algorithm is ideal, as it minimizes the computational overhead required for real-time operation.

2.6 Improving Processing Speed by Using Dual Image Acquisition Windows

As mentioned earlier, the two goals of the machine vision system are 1) tracking the hook, and 2) tracking the wand/glove. The challenge, however, is that the simultaneous tracking of two independently moving objects not only lengthens the image acquisition times from the CCD array, but also increases the computational overhead on the camera processor.

Figure 26 shows the image acquisition and processing times for the Siemens VS723-2 smart camera. The X-axis indicate the side length (in pixels) of a square image acquisition window. The acquisition time is defined as the time it takes to capture all the pixels inside the defined acquisition window. That is, the time starting from the initial trigger and ending at the time at which all the intensity values of the pixels within the window are stored in memory. The total processing time is defined as the acquisition time plus the time it takes to execute the blob generator (see section 2.3) over the entire window. Clearly, there is an exponential relationship between the image processing time and the window size. The acquisition times also increase with the window size, albeit in a linear fashion. It is beneficial to minimize the image processing time as much as possible so that camera tracking can be updated at the fastest possible rate, thereby minimizing the possibility of tracking failure. However, as the wand/glove moves around the visible workspace, as was shown in Figure 23, the size of a single image acquisition window must increase in order to track both the wand/glove and the hook. Consequently, this motion will have detrimental effects on the total image processing time.

One solution is to track the wand/glove and the hook using two independent image acquisition windows. The windows are centered on each tracked target and their areas are set such that they cover an area slightly larger than the target. In this way, the size of the acquisition windows are kept at a minimum. The benefit of using

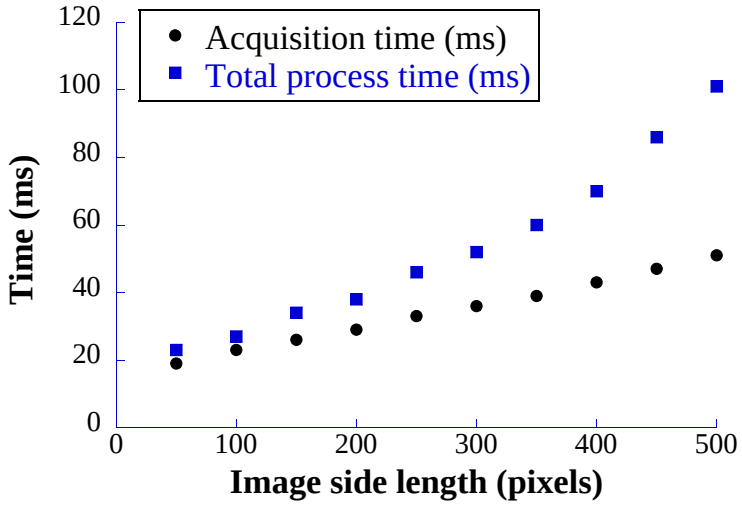


Fig 26. VS723-2 Image Acquisition and Processing Times

independent image acquisition windows is two fold:

1. Image processing time is reduced as the image area that is processed is less.

2. The majority of the area that is visible to the camera is ignored because only two small, independent tracking windows are considered. This reduces the impact of spurious signals that may be present around the workspace.

The operation of the camera is divided into three operational modes, as shown in Figure 27. Effectively, the camera becomes a finite state machine that switches between states when certain conditions are met.

In state 0, which is the default state after camera start up and initializations, the camera will only use one image acquisition window that spans the entire visible area. It will search within this window for the hexagonal patterned hook blobs using the algorithm described in section 2.4.1. If the hook is successfully located, the camera will switch to state 1.

In state 1, the camera is actively tracking the hook and is also searching for the wand/glove blob in the vicinity around the hook. One acquisition window is again used in state 1. However, it is centered around the hook position (center of the

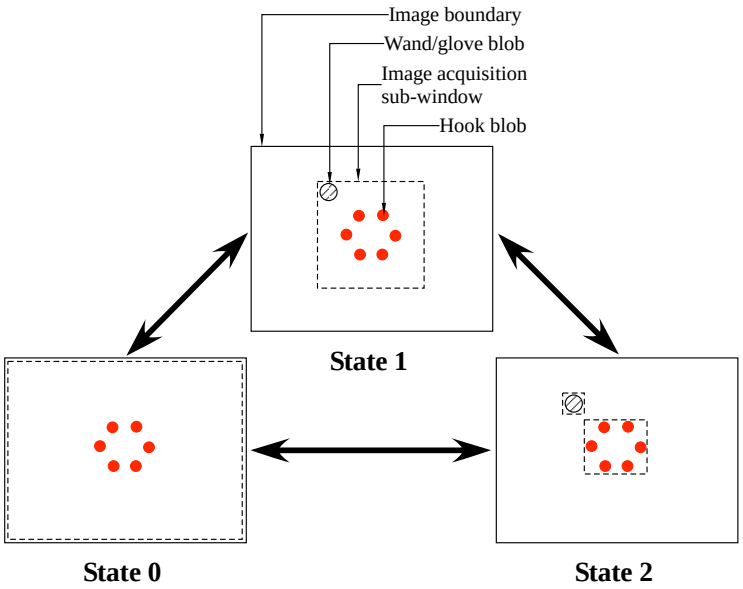


Fig 27. Camera Operational States

hexagon) and its side lengths are reduced such that it covers an adequate area around the hook in order to search for the wand/glove blob. The camera will determine the presence of a wand/glove blob using the K-means clustering algorithm described in section 2.5. Once the wand/glove is detected, the camera switches to state 2.

In state 2, there are two independent acquisition windows that track the positions of the hook and the wand/glove. These windows are centered about their respective targets with side lengths that fit tightly around the diameter of each target. If the hook position is lost at any time during state 1 or state 2, the camera will switch back to state 0 to relocate the hook.

A flow chart of the camera algorithm is displayed in Figure 28. After some initialization routines such as resetting memory registers, the camera establishes communication with the PLC in UDP protocol. UDP is chosen over TCP protocol for its faster speed and smaller bandwidth overhead. After establishing a communication link, the camera enters an infinite loop (indicated by the bordered box) where it executes different parts of the algorithm depending on which of the three states it is in. This is visible in the flow chart where the program splits into three different

branches after querying the state of the camera.

During state 0, the leftmost branch of the flow chart, the acquisition window is set to maximum with the goal of locating the hook. If the hook is found, then the camera changes to state 1 and the hook position information is sent to the PLC. If no hook is found, then the camera remains in state 0.

In state 1, represented by the middle branch on the flow chart, the acquisition window is set to cover the proximate area around the hook. Following this, the K-means clustering algorithm is used to locate the presence of the wand/glove. However, the clustering algorithm only checks for the presence of multiple clusters, and does not guarantee the presence of the hook. Therefore, the hook-locating algorithm is executed next. If no hook is found, regardless of the presence/absence of the wand/glove, then the camera switches to state 0. If no wand/glove is found, but the hook is still present, then data on the hook location is still sent to the PLC and the camera remains in state 1. If both wand/glove and hook are present, then data on both hook and wand/glove is sent to the PLC and the camera switches to state 2.

In state 2, which is located on the far right section in the flow chart, a binary state variable "Inspection Mode" is used to alternate the image-processing between the two independent acquisition windows. As parallel processing is not an available option with the onboard processor, the camera must process each acquisition window in a sequential manner. If Inspection Mode equals 2, then the camera is set to process the acquisition window that tracks the wand/glove. If a wand/glove blob is detected, then it is subject to a consistency check. This check ensures that important blob characteristic parameters have not changed substantially from the last cycle. Some examples of these parameters include the blob area, size, and geometry such as eccentricity. Upon a successful consistency check, information about the wand/glove is saved. If no wand/glove is found, or if the consistency check fails, then a status flag denoting wand/glove not found is set to true. Then, the Inspection Mode is always

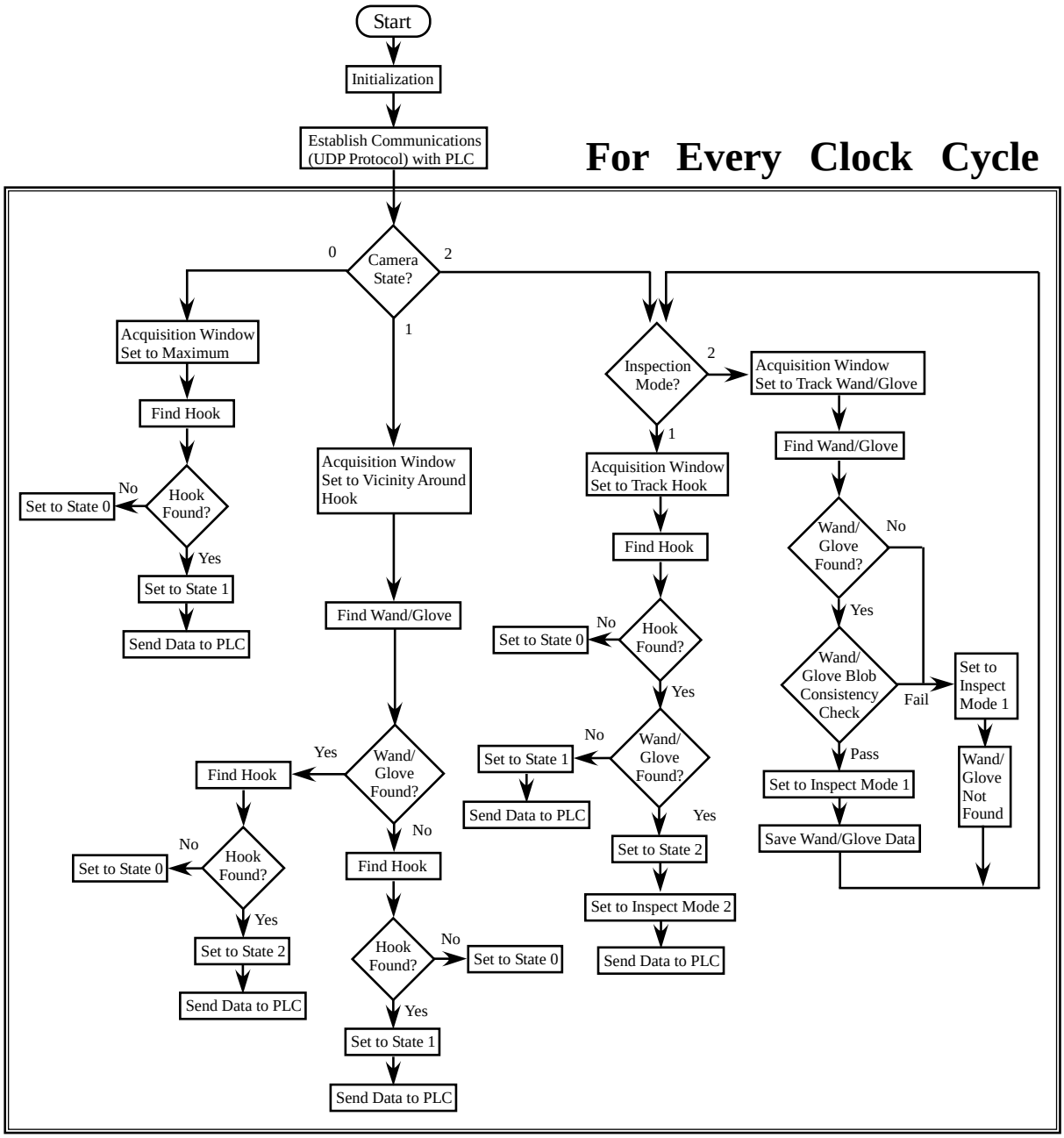


Fig 28. Operational Flow Chart of the Camera Algorithm

set back to 1 and the acquisition window responsible for tracking the hook is executed next.

The procedure of execution for Inspection Mode 1 and Camera State 2 begins with acquiring and processing the acquisition window responsible for tracking the hook. Then, the hook-search routine is performed. If no hook is found, then the camera changes state to 0. If the hook is present but no wand/glove is found, then the camera is set to state 1 and only the hook data is sent to the PLC. If both hook and glove/wand are present, the camera remains in state 2, Inspection Mode is set to 2 for the next cycle, and both hook and wand/glove data are sent to the PLC.

With the above algorithms implemented on the Siemens VS723-2 smart camera, and while the camera is in state 2, the system is able to acquire, process, and send data to the PLC in under 70 ms for each acquisition window. The cycle rate is set at 140 ms. When the system is in state 1, the cycle rate is maintained at 140 ms even though the total processing time is actually much less. It is prudent to maintain consistent inter-state and intra-state cycle rates for the following reasons:

1. Communication with the PLC is established via two-way UDP protocol. Even though UDP protocol is connectionless, a mismatch in the rate at which data is sent and received between the two devices will result in lost packets of data or excessive flooding. As mentioned before, the PLC cycle time is 35 ms, which is four times faster than the camera. Therefore, the PLC receives data from the camera every four PLC cycles.

2. A consistent cycle rate is essential to maintain the fidelity of the hook predictor that is based on crane dynamics. This hook predictor will be described in section 2.7.

However, when the camera is in state 0, the cycle rate is 1000 ms. The requirement for a consistent cycle rate is relaxed because the camera needs to search the entire

image for the hook. Furthermore, no data is sent to the PLC during state 0.

The camera cycle rate is compared with the period of hook oscillation in order to assess if the camera update speed is adequate. Assuming a typical crane operation where the suspension length is 5 meters, the period of oscillation is approximately 4.5 seconds. In this case, when the camera is tracking the hook, a cycle rate of 140 ms is only 3.1% of the period of oscillation. In other words, the camera refreshes 32 times every period. If the hook is hoisted such that the suspension length is 2 meters (this height is near the limit at which the camera is still able to reliably track the hook at the extremities of its oscillation within its visible workspace), then the period of oscillation is approximately 2.84 seconds. In this case, the 140ms cycle rate is about 4.9% of the period of oscillation. The camera refreshes 20 times every period. In both cases, the camera refresh rate more than satisfies the Nyquist criterion for minimum sampling rate. For all practical purposes, this is fast enough for real-time tracking.

2.7 Predictive Tracking

So far, algorithms have been described for determining the location of the hook and distinguishing between the wand/glove blob and the hook blobs. Processing speed has been improved and robustness increased with the use of two acquisition windows (as opposed to one window) for the independent tracking of the hook and the wand/glove. Each acquisition window is centered around the target that they are tracking, and the size of each acquisition window is minimized in order to reduce processing time. As the targets move around the visible workspace, the acquisition windows must follow their positions in a timely and accurate manner to maintain successful tracking.

For this reason, tracking performance can be improved if the motion of the tracked target can be predicted. Accomplishing this will involve 1) predicting the future position of the moving object at least one sample time ahead, and 2) moving the center of the acquisition window to the predicted position. A simple method is to use

a predictive filter that calculates the future position based on the current and last observations. The X and Y coordinates (in unit pixels) are separated for simplicity:

$$\hat{X}(n+1) = X(n) + [X(n) - X(n-1)] \tag{6}$$

$$\hat{Y}(n+1) = Y(n) + [Y(n) - Y(n-1)] \tag{7}$$

Where n corresponds to the index of the image frame, i.e. $\hat{X}(n+1)$ refers to the predicted X position in the next time step, $X(n)$ refers to the observed X position in the current time step, and $X(n-1)$ refers to the observed X position in the last time step. Alternatively, the predictive filter can be constructed based on the weighted sum of the previous M observations:

$$\hat{X}(n+1) = \sum_{i=0}^{M} w_i X(n-i), \tag{8}$$

where w_i denotes the i^{th} weight. A notable downside of this method is that increasing the value of M will increase the lag of the prediction. This is because the algorithm essentially acts as a smoother based on previous observations. The main draw of this type of predictive filter is its simplicity. However, it fails to utilize the readily-available knowledge and data on the dynamic movements of the tracked objects. For example, in the absence of disturbances, the oscillatory movement of the pendulum hook is well known and highly predictable. Knowledge of pendulum dynamics can easily be exploited to construct a more accurate predictor. The movement of the wand/glove is more difficult to predict because its dynamic motions are driven by a human operator. However, it is still constrained by real world dynamics and can be modeled as an inertial mass subjected to a finite force.

We explore a smarter predictor for the hook. To do so, the dynamic nature of the pendulum hook must first be understood.

2.7.1 Crane Dynamics

This section derives the equations of motion for a planar trolley and suspended hook/payload system. It parallels the work in [41].

Planar motion of the mechanical system can be modeled as a multi-body system consisting of two rigid bodies, as shown in Figure 29. As the bridge crane's trolley is able to move in both X and Y axes, we assume that the dynamics are identical but independent and decoupled between the two axes. The mass of the trolley and the mass of the payload are labeled as m_t and m_p, respectively; acceleration due to gravity is represented as g; a viscous damping force, which acts on the payload, can be described by the damping coefficient b; and the length of the cable is labeled as L. For simplicity, the cable length is assumed to vary slowly with time, thus the time derivatives of this quantity are neglected. The trolley position, x, will be considered as the controlled variable. An obvious benefit to such a choice for the system input is that a model of crane motors and industrial drives is unnecessary at this juncture. These systems will later be considered when determining appropriate reference signals for the crane.

With the crane modeled this way, the system reduces to one degree of freedom with the cable angle, θ, used as the generalized coordinate. Using the Lagrangian formulation to derive the system equations, we begin with the kinetic energy of the trolley and the payload:

$$T_t = \frac{1}{2} m_t \dot{x}^2 \tag{9}$$

$$T_p = \frac{1}{2} m_p v_p^2 \tag{10}$$

where v_p^2 is the squared velocity of the payload. v_p^2 may be computed from the position

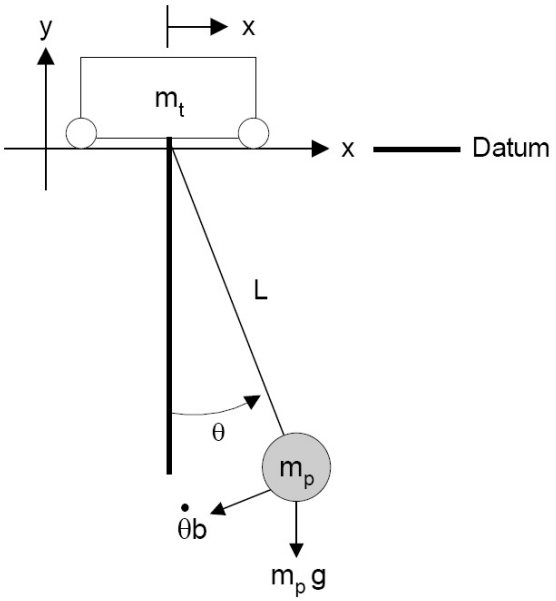


Fig 29. Multi-Body Model of the Crane Along One Axis

vector, r_p, of the payload as follows:

$$v_p^2 = \dot{r_p} \cdot \dot{r_p} \tag{11}$$

$$\dot{r_p} = \frac{dr_p}{dt} \tag{12}$$

$$r_p = (x + Lsin\theta)\hat{i} + (-Lcos\theta)\hat{j} \tag{13}$$

Substituting (11) through (13), and summing (9) with (10), the total kinetic energy is given by:

$$T = \frac{1}{2}m_t\dot{x}^2 + \frac{1}{2}m_p\left[(\dot{x} + L\dot{\theta}cos\theta)^2 + (L\dot{\theta}sin\theta)^2\right] \tag{14}$$

With reference to the potential energy datum shown in Figure 29, the potential energy U of the system is given by:

$$U = -m_p gLcos\theta \tag{15}$$

The Lagrange equation for the generalized coordinate, θ, is:

$$\frac{d}{dt}\frac{\partial T}{\partial \dot{\theta}} - \frac{\partial T}{\partial \theta} + \frac{\partial U}{\partial \theta} = Q_\theta \tag{16}$$

where Q_θ is the generalized force acting on the system. The left hand elements of (16) are computed as:

$$\frac{d}{dt}\left(\frac{\partial T}{\partial \dot{\theta}}\right) = m_p(-L\dot{x}\dot{\theta}sin\theta + L^2\ddot{\theta} + L\ddot{x}cos\theta) \tag{17}$$

$$\frac{\partial T}{\partial \theta} = m_p(-L\dot{x}\dot{\theta}sin\theta) \tag{18}$$

$$\frac{\partial U}{\partial \theta} = m_p gLsin\theta \tag{19}$$

The right hand element of (16) comes from a virtual work formulation of the non-conservative forces acting on the system:

$$\delta W = F \cdot \delta r_p = (Q_\theta)\delta\theta, \tag{20}$$

where F is the non-conservative force vector:

$$F = (-b\dot{\theta}cos\theta)]\hat{i} + (-b\dot{\theta}sin\theta)\hat{j}. \tag{21}$$

Substituting (21) into (20) and using the expression for r_p from (13), we see that:

$$\delta W = (-bL\dot{\theta})\delta\theta. \tag{22}$$

Therefore,

$$Q_\theta = -bL\dot{\theta}. \tag{23}$$

Substituting (17) through (19) and (23) into the Lagrange equation, (16), the differential equation governing the motion of the system is:

$$\ddot{\theta} + \left(\frac{b}{Lm_p}\right)\dot{\theta} + \left(\frac{g}{L}\right)sin\theta = \left(\frac{-cos\theta}{L}\right)\ddot{x}. \tag{24}$$

We can use the small angle approximation to linearize the system to:

$$\ddot{\theta} + \left(\frac{b}{Lm_p}\right)\dot{\theta} + \left(\frac{g}{L}\right)\theta = \left(\frac{-1}{L}\right)\ddot{x}. \tag{25}$$

Recognizing that (25) represents a second-order damped oscillatory system, we may write:

$$\frac{b}{Lm_p} = 2\zeta\omega_n \tag{26}$$

$$\frac{g}{L} = \omega_n^2 \tag{27}$$

$$\frac{-1}{L} = -\frac{\omega_n^2}{g}. \tag{28}$$

As it is common to issue desired velocity signals directly to the motor drives, we wish to obtain a final system representation relating the trolley velocity to the angle of the cable. For this reason, we substitute $\dot{v}_t$ for $\ddot{x}$. Assuming zero initial conditions, and using the relations in (26) through (28), we obtain the following transfer function relating the cable angle to the velocity of the trolley:

$$\frac{\Theta(s)}{V_t(s)} = \frac{\left(\frac{-\omega_n^2}{g}\right)s}{s^2 + 2\zeta\omega_n s + \omega_n^2}. \tag{29}$$

A control canonical state space representation of the system is obtained directly from the coefficients of the transfer function. By letting q_1 and q_2 be the states of the system, we have:

$$\dot{q} = \begin{bmatrix} 0 & 1 \\ -\omega_n^2 & -2\zeta\omega_n \end{bmatrix} q + \begin{bmatrix} 0 \\ 1 \end{bmatrix} v_t(t), \tag{30}$$

and

$$\theta(t) = \begin{bmatrix} 0 & \frac{-\omega_n^2}{g} \end{bmatrix} q. \tag{31}$$

By observing the relationship between q_2 and θ established in (31), one may recognize that the state is equal to $-\theta L$. That is, $-q_2$ represents the relative horizontal displacement between the trolley and the payload. The state variable, q_1, has less physical meaning; $-q_1$ is a quantity whose derivative yields the relative displacement between the trolley and the payload.

2.7.2 Predictive Tracker based on Hook Dynamics

The process for creating the predictive hook tracker starts with discretizing the continuous crane velocity transfer function in (29) with a sampling rate equivalent to the camera's refresh rate. As there are dynamic variables such as the sampling rate T_s, the damping ratio ζ, and the natural frequency ω_n, we use the Tustin's approximation to obtain simple algebraic rules to construct the difference equation. The Tustin's approximation (otherwise known as the Bilinear transform) converts from the S domain to the Z domain using the substitution $s = \frac{2(z-1)}{T_s(z+1)}$. Making this substitution into (29) we get:

$$\frac{\Theta(z)}{V_t(z)} = \frac{\frac{-200}{981}T_s\omega_n^2(1-z^{-2})}{\left(\begin{array}{c}(4+\omega_n^2T_s^2+4\zeta\omega_nT_s)+(2\omega_n^2T_s^2-8)z^{-1}\\+(4-4\zeta\omega_nT_s+\omega_n^2T_s^2)z^{-2}\end{array}\right)}, \tag{32}$$

which has the corresponding difference equation:

$$\theta(n) = \begin{array}{c}\frac{1}{4+\omega_n^2T_s^2+4\zeta\omega_nT_s}\{(\frac{-200}{981}T_s\omega_n^2)V_t(n)+(\frac{200}{981}T_s\omega_n^2)V_t(n-2)\\-(2\omega_n^2T_s^2-8)\theta(n-1)-(4-4\zeta\omega_nT_s+\omega_n^2T_s^2)\theta(n-2)\}\end{array}. \tag{33}$$

Using a time shift and approximating $V_t(n+1)$ with $V_t(n)$, which is valid as the camera's sampling rate is much faster than the highest frequency that the trolley is capable of moving at, the difference equation for the predictive filter based on hook dynamics can be obtained:

$$\hat{\theta}(n+1) = \begin{array}{c}\frac{1}{4+\omega_n^2T_s^2+4\zeta\omega_nT_s}\{(\frac{-200}{981}T_s\omega_n^2)V_t(n)+(\frac{200}{981}T_s\omega_n^2)V_t(n-1)\\-(2\omega_n^2T_s^2-8)\theta(n)-(4-4\zeta\omega_nT_s+\omega_n^2T_s^2)\theta(n-1)\}\end{array}. \tag{34}$$

Note that the values for $\theta(n-k)$, for $k = 0, 1$ are not obtained from past predictions but instead from past observations - measurements made by the camera. The trolley velocity is obtained from encoders on the motor and is sent to the camera from the PLC via a two way UDP communication protocol. The predicted pixel position of the hook is then calculated from:

1. $\hat{\theta}(n+1)$ from (34);

2. X_0, the expected pixel position of the hook when θ is zero (calculated from the hook height);

3. A constant gain, K_{camera}, which relates pixel position to angular displacements in the camera view.

In the X direction, the predictor is defined by:

$$\hat{X}(n+1) = \hat{\theta}(n+1)K_{camera} + X_0. \tag{35}$$

2.7.3 Comparison Between the Dynamics-Based Predictor and the Simple Predictor for Hook Tracking

Given a choice between the simple predictor (Defined by (6) and (7)), and the dynamics-based predictor (Defined by (35)), it would be useful to obtain a quantitative measure that compares the performance between the two predictors when they are both applied to the task of tracking the hook. The simple predictor is easy to implement, but lacks accuracy. The dynamics-based predictor will yield more consistent tracking, but at the cost of being more computationally intensive, which puts more overhead on the already-limited capabilities of the camera's onboard-processor. The question we would like to answer is: given a user-specified tolerable amount of predictor error, under what conditions would the simple predictor perform just as well as the dynamics-based predictor?

To answer this question, a measure to quantify the performance of the predictor must first be defined. To this end, we define an error measure $Error_{pixels}$ to quantify the predictor performance in pixels:

$$Error_{pixels} = X(n+1)_{actual} - \hat{X}(n+1)_{predictor}, \tag{36}$$

where $X(n+1)_{actual}$ and $\hat{X}(n+1)_{predictor}$ corresponds to the actual position and the predicted position of the hook (in pixels) in the X direction, respectively. A similar

measure is used for the Y direction.

The variables that influence predictor performance must first be identified. The fidelity of tracking is predominantly determined by how much the hook moves across the camera's visible workspace between each sample time. Thus, the sample time is the first major factor that influence predictor performance - the hook will have moved relatively few pixels in one cycle if the camera refresh rate is fast. A slow camera refresh rate will not only suffer from sparse data and relatively few observations, but aliasing may also be present.

The second factor is the speed of hook movement. The suspension length determines the speed at which the hook moves across the camera's view. If the cable length is short, the frequency of oscillation will be high. Additionally, the hook is close to the camera with a short hook length, which means that even small lateral hook movements will translate to large pixel movements in the camera view.

A Matlab simulation was used as the test-bed to compare predictor performance. As mentioned earlier, crane trolley movement and crane maximum speed will have significant impact on the speed of hook movement, and thus on the performance of the predictor. To ensure a fair comparison, the input to the Matlab crane model was a step command in velocity with an amplitude equivalent to the maximum speed (0.3577 m/s) of the 10-ton crane at Georgia Tech. First, a dynamic simulation subject to the above step input was performed and a time history of the pixel positions of the hook was recorded at each sample time. This was repeated for cable lengths ranging from 2 to 6 meters, and for sampling times of 0.05 to 0.5 seconds. This yielded a 4-D matrix containing the time history of actual hook pixel positions, X_{actual}, as it appeared to the camera, for a range of cable lengths and sampling times.

The second part was to simulate the simple predictor (6) and the dynamics-based predictor (35). For each cable length and sample time, we obtained a time history of the predicted pixel positions of the hook for both types of predictors, i.e. $\hat{X}(n+1)_{simple}$

and $\hat{X}(n+1)_{dynamics}$. Note that the predictor difference equations require current and past observations $X(n)$ and $X(n-1)$. These values were obtained from X_{actual}. In other words, we assumed the predictor has successfully tracked the hook up to the current cycle, and the variables we wish to compare are $\hat{X}(n+1)_{predict}$, the predicted position, versus $X(n+1)_{actual}$, the actual position, with the error function defined by (36).

We also assumed the dynamic-based predictor used the actual values for ζ and ω_n (refer to (34)). Of course, the real values for these two parameters will not be known exactly for a real crane, so this assumption yields the most ideal dynamics-based predictor performance. This is desirable, as this will yield conservative estimates when the performance of the simple predictor (which does not rely on any knowledge about the hook dynamics) is compared against the performance of the ideal dynamics-based predictor. The velocity of the trolley, V_t, which was the step input that was issued to the model, was also known to the dynamics-based predictor.

The predictors were simulated to produce a 4-D matrix containing the time history of predicted hook pixel positions, $\hat{X}(n+1)_{predict}$, for a range of cable lengths (2 to 6 meters) and sampling times (0.05 to 0.5 seconds). This was compared with the 4-D matrix of actual hook pixel positions, X_{actual}, produced earlier. The result of this comparison is a 4-D matrix containing the time history of $Error_{pixels}$, for the same range of cable lengths and sampling times.

The maximum error was taken from each set of time histories from the $Error_{pixels}$ 4-D matrix and plotted against cable length and sample time, as shown in Figure 30. Superimposed on Figure 30 is an additional mesh corresponding to the acceptable level of pixel error - arbitrarily defined here as 10 pixels. Clearly, there is more predictor error when the sampling time is long and the cable length is short. The dynamics-based predictor shows excellent performance, as a large proportion of the predictor error lies beneath the acceptable limit. The same cannot be said, however,

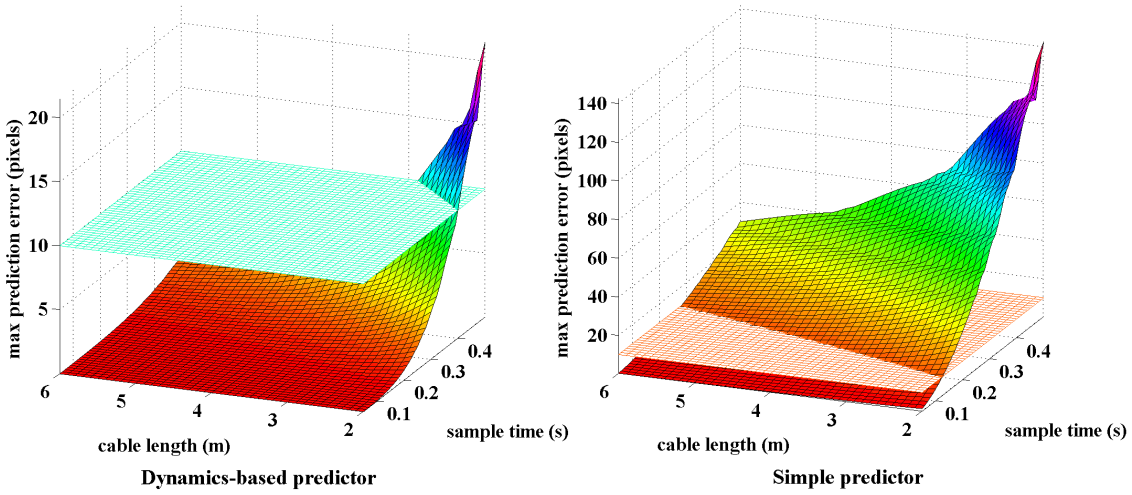


Fig 30. Predictor Error Comparison - Isometric View. Acceptable error = 10 pixels

for the simple predictor. Figure 31 shows the top view of the same graphs, which clearly delineates the areas where the predictor error exceed the acceptable limit. Interestingly, for our wand/glove application, which has a sampling time of 0.14 seconds, the simple predictor's performance satisfies our arbitrary limit. That is to say, there is no perceivable advantage to use the dynamics-based predictor over the simple predictor. Experimentally, this limit of 10 error pixels is more than adequate to ensure good tracking.

The utility of Figures 30 and 31 comes from the fact that a user can easily distinguish the regions of operation where a simple predictor will suffice and regions where the advanced dynamics-based predictor is required. Furthermore, this method for comparing predictor performance can easily be generalized for other classes of predictors by modifying the predictor difference equations, and for different cranes and crane movements by modifying parameters such as the damping ratio, crane maximum speed, camera sampling time, hook cable length, and input profiles to the model.

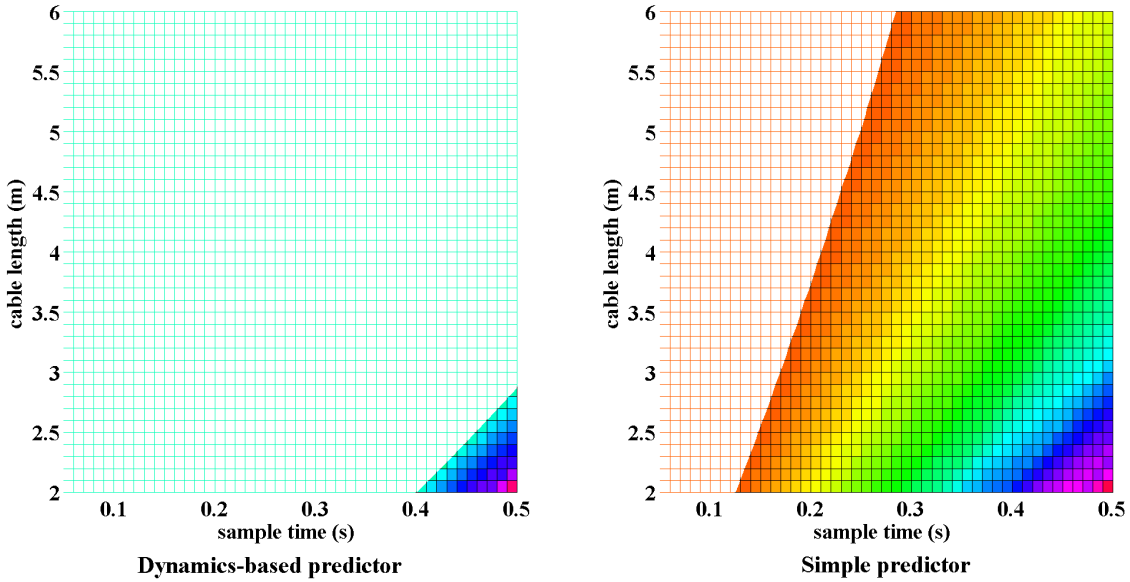


Fig 31. Predictor Error Comparison - Top View. Acceptable error = 10 pixels

2.8 Radio-Frequency-Based Crane Control

Reliance on machine vision makes the tracking of the wand and glove susceptible to occlusions by the hook cables, as it was shown in Figure 23. It is also susceptible to other corruptive sources such as ambient light and spurious signals generated by reflective surfaces. Furthermore, simultaneous tracking of the crane hook and the wand/glove places significant computational requirements on the processor of the camera. An alternative interface is proposed that uses radio frequency wireless technology as the means for the operator to issue command signals.

In contrast to the wand and glove control, the interface for the RF-based crane control does not rely on machine vision to detect the position of the hand-held device. The hand-held device for this interface is a small RFID tag, which was shown in Figure 5. Its position is acquired with the use of multiple RF-based sensors placed around the workspace. The position of the tag is then used as the command signal to drive the crane.

The RF-based system used in this thesis is the academic version of the commercial

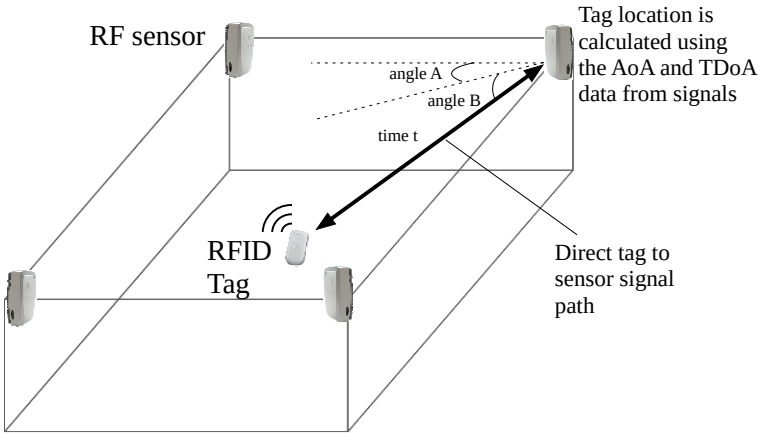


Fig 32. RF Sensors and Tag of the RF-based System

off-the-shelf product from Ubisense. The technology uses Ultra-Wide-Band (UWB) Radio Frequency signals. UWB RF has the advantage that spurious surface-reflected signals can easily be distinguished from signals that traveled on the true, direct tag-to-sensor paths. This allows an accuracy of 0.15 meters in 3-D, whereas traditional non-UWB, RF-based location technologies deliver typically only 3 to 10 meters of accuracy. The Ubisense kit consists of four 7000 series sensors, ten Ubisense RFID tags, and related software. The four sensors are mounted 6m above the floor to cover a 8m by 6m section of the workspace used by the 10-ton bridge crane in the Georgia Institute of Technology's MaRC building. The schematic of the RF-based system is illustrated in Figure 32. The tag emits UWB RF signals, which is then received by the sensors. Using the signals' Time Difference of Arrival (TDoA) and the Angle of Arrival (AoA) data as it reaches the sensors, the location of the tag in 3-D space can be determined.

The advantage of using RFID technology over machine vision is that it has a lower requirement for a clear line-of-sight from the hand-held device to the sensors. Under wand/glove control, the only sensor available is the camera, and if the line-of-sight from the wand/glove to the camera is broken (e.g. occlusion by the hook cables), tracking fails. Under RF-based control, there are four sensors available to track the

RFID tag. There is also a redundancy, as the tag needs a direct line-of-sight to just three sensors for accurate tracking. More sensors could be added for robustness and reliability. Tracking is still possible even if there are no direct lines-of-sight to any sensor because Ubisense algorithms can still calculate the tag position from reflected signals. Thus, it is unlikely that tracking would ever fail when under RF-based control. Furthermore, RF-based control will work within the larger workspace that is defined by the corner positions of the RF-sensors. This workspace covers a much greater area than the area that is visible to the camera. Both the wand and glove interfaces are limited to the the visible workspace of the camera.

Additional features are also available under RF-based control, as the crane operator will be able to send miscellaneous commands such as hoisting and lowering using the buttons on the tag. The tag uses the UWB channel for real-time location and a separate RF channel for the bi-directional communication with the sensors. These are used for the buttons and LED lights on the tag. The buttons can also be used to signal an emergency stop. This has a safety advantage over the vision-based wand/glove control because it offers the operator a means to directly stop the crane. Under wand/glove control, the command signal to move the crane will always exist if a wand/glove blob is visible to the camera. However, any reflective surface that produces blobs that are similar to the wand/glove blob can also generate a command signal. The camera cannot distinguish between this spurious surface and the wand/glove. Thus, in the presence of such a surface, the operator has no means to stop the crane.

In its current state, under RF-based control, the position of the hook is tracked by the camera and the tag is tracked by the RF sensors. However, the RF system is expandable. The position of the hook can also be tracked using RF by attaching a separate tag to it. Furthermore, if the hook is carrying a payload, then the position of the payload can also be tracked. With machine vision, the payload cannot be

tracked by the overhead camera because it is occluded by the hook. For crane-type applications, it is typically very difficult to locate the position of the payload without the use of expensive sensors. RF-based tracking and locating is a cheaper and more elegant alternative.

2.9 Summary

This chapter began with a description of the 10-ton crane hardware that was used for the experimental verifications of this thesis. The reader was then introduced to the hardware, the software, and the algorithms behind the machine vision module, which served as the backbone to the interface of wand and glove control. The chapter concluded with a description of the RF system used for the RF-based crane control.

The contributions of this thesis that were discussed in this chapter are summarized below:

1. The use of two camera acquisition windows for the independent tracking of hook and wand/glove; this facilitated the fast camera cycle time that is necessary for real-time tracking.

2. The use of the K-means algorithm to distinguish between hook and wand/glove blobs.

3. The use of predictors to increase the accuracy and efficiency of tracking the hook.

4. The RF-system that was installed for the RF-based crane control.

CHAPTER III

CONTROLLER DESIGN FOR INTUITIVE CRANE CONTROL

Chapter 2 described the technology and algorithms behind the hand-motion control interfaces. The goal of the interface is to acquire the position of the hand-held device. Figure 33a shows that under wand/glove control, both the positions of the hook and the wand/glove are tracked by the camera. Figure 33b shows that under RF-based control, the RFID tag is tracked by the RF sensors while the hook is still being tracked by the camera.

Figure 33 also illustrates the position of the hand-held device being compared with the position of the crane trolley to produce an error measurement, $e1$. This measurement generates a command signal that needs to be processed by the controller in order to produce a suitable crane trolley motion, and subsequently the desired hook trajectory.

However, the desired hook trajectory is hard to define in the context of intuitive crane control because it is highly subjective. What may be intuitive to one person may seem counter-intuitive to another. Some may prefer a rapid and immediate response, while others may prefer a more relaxed and placid approach. Despite individual preferences, one feature that is universally desirable is the reduction of residual oscillations commonly associated with crane-type systems. To address this, input shaping was employed.

This chapter describes the control architectures that were designed to address the task of generating the desired hook trajectory. The architectures investigated were 1) Proportional Derivative (PD) with low gains, 2) PD with high gains, and 3) PD with

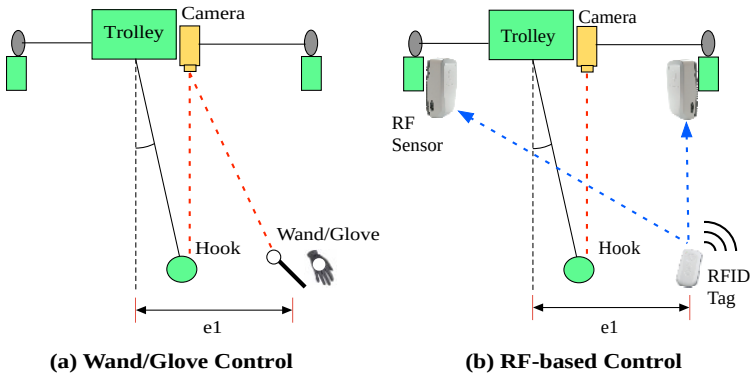


Fig 33. The Generation of $e1$ Under Wand, Glove, and RF-based Control

input shaping. Simulation and experimental results are presented. Because the underlying controller is the same regardless of the interface (glove, wand, or RF-based), the generic umbrella term "Hand-Motion Control" will be used interchangeably with the 'glove/wand/RF-based controller' in this chapter.

3.1 Standard Pendent Control

Before describing the specifics of the hand-motion controller, it is worthwhile to explain the control architecture of the standard pendent-based crane control. Figure 34 shows the control block diagram. By depressing any of the buttons on the control pendent (e.g. forward, backward, left, right) the pendent issues a reference velocity command, V_R to the block labeled "Motor, Trolley", which describes the dynamics of the two components. "Motor" is a subsystem describing the inertial mass of the rotor inside the induction motor, the motor drive, and feedback provided by the motor encoders. To control the speed of the motor shaft such that the velocity of the entire trolley assembly, V_T, tracks V_R, the motor drive has its own PI (Proportional Integral) controller that controls the voltage and frequency of the driving circuit. "Trolley" describes the dynamic properties of the entire trolley assembly that moves along the bridge and runway. The true underlying dynamics of the entire "Motor, Trolley" is complex, high-ordered, and contain several notable non-linearities such

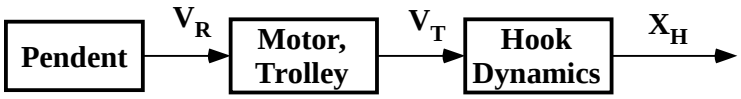


Fig 34. Standard Pendent Control Block Diagram

as motor saturation, rate limit on the motor velocity, a switching-element [41] that prevents the motor from reversing while it is still in forward motion, and dead-zone due to friction. However, for our purposes, it can be adequately approximated with a second-order model:

$$\frac{V_T(s)}{V_R(s)} = \frac{\omega_{n,MT}^2}{s^2 + 2\zeta_{n,MT}\omega_{n,MT}s + \omega_{n,MT}^2} \tag{37}$$

where $\omega_{n,MT}$ and $\zeta_{n,MT}$ denote the natural frequency and damping ratio with experimentally obtained values of $6.98 rad/s$ and 0.86, respectively.

"Hook Dynamics" in Figure 34 is the second-order system that describes the position of the hook relative to the trolley (X_H), with respect to the velocity of the trolley (V_T). This has a form similar to (29), except that it is multiplied by the length of the hook cable, L, (under small angle assumption) so that $X_H(s)$ replaces $\theta_(s)$ as the output variable:

$$\frac{X_H(s)}{V_T(s)} = L\frac{\left(\frac{-\omega_n^2}{g}\right)s}{s^2 + 2\zeta\omega_n s + \omega_n^2}. \tag{38}$$

Typical responses to ramp-plateau inputs using the pendent controller are shown in Figure 35. A ramp-plateau input in position (equivalent to a pulse input in velocity) is the equivalent of pushing a button on the pendent for a specified amount of time, which commands the crane trolley to move a certain distance. Due to the pendulum-like nature of the hook dynamics, this type of movement will, in general, induce significant residual oscillations. Figure 35 shows the hook response to trolley move distances of approximately 2 meters and 3 meters. Note that the amplitude of residual hook oscillations is dependent on the magnitude of the trolley move distance.

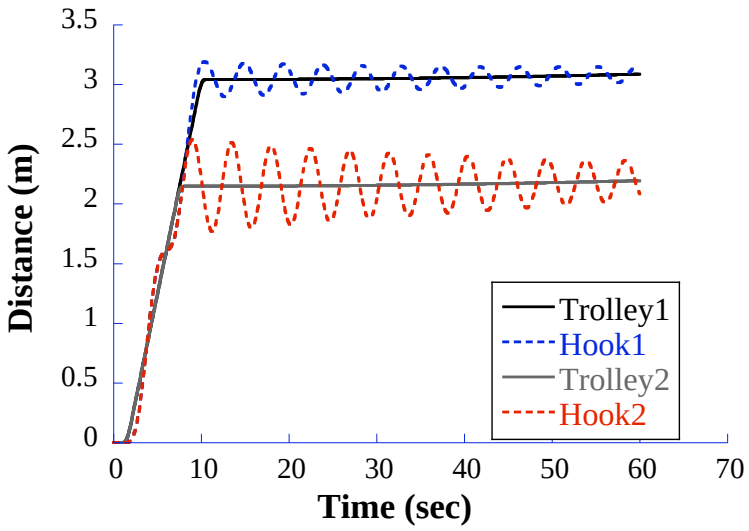


Fig 35. Typical Point-To-Point Response Using Pendent Control

3.2 Proportional Derivative Hand-Motion Controller

Figure 36 shows the control block diagram for the Proportional Derivative (PD) hand-motion controller. The trolley position, obtained by integrating the velocity of the trolley, V_T, is subtracted from the position of the hand-held device to yield an error signal, $e1$. Note that the position of the hook is actually the output variable that we wish to ultimately control. Therefore, it may seem conceptually sound to use the position of the hook, X_H, in the feedback loop. There are four reasons why we do not do this:

1. The transfer function of the "Hook Dynamics" block, given by (38), shows that it has stable poles. This means that the hook will always come to rest directly beneath the trolley. Therefore, accurate positioning of the trolley leads to accurate positioning of the hook, if the hook does not oscillate.

2. Excluding the "Hook Dynamics" block from the feedback loop also simplifies analysis of the overall system, which allows for a more easily implementable controller.

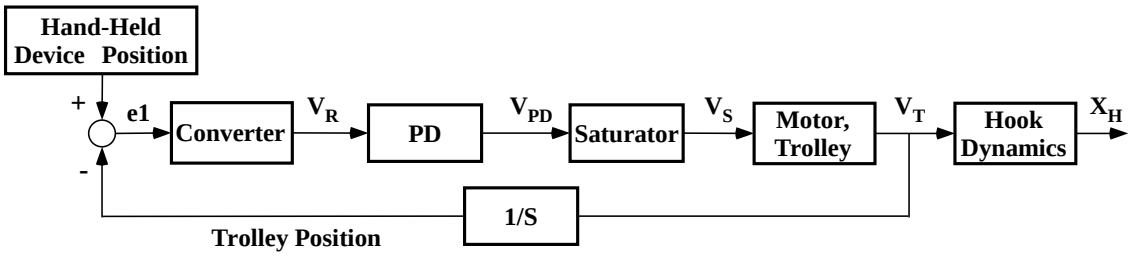


Fig 36. PD Hand-Motion Control Block Diagram

3. The tracking of the hook (by machine vision) is not as robust or reliable as the tracking of the trolley (by laser range sensors). Measuring the hook position with the camera may fail if the hook is occluded by obstacles, such as the suspension cables.

4. The camera is a non-collocated sensor, which is well known for causing a range of stability issues [**?**]. To prevent potentially unstable crane movements when tracking fails, the hook position should not be used for feedback.

The error, $e1$, was also shown conceptually in Figure 33. This error signal is converted by the "Converter" block from a distance measurement to digital units that corresponding to the PLC's internal representation of the reference velocity, V_R (ranging from -100 to 100). The "Converter" block is a piecewise function given by:

$$V_R = \begin{cases} 100 \times \frac{e1}{e_{max}} & : abs(e1) \leq e_{max} \\ sign(e1) \times 100 & : e1 > e_{max} \end{cases} \tag{39}$$

where, e_{max} is an arbitrarily set maximum saturating error distance.

The reference velocity signal, V_R, is sent to the PD controller to produce V_{PD}. In essence, the role of the proportional component is to drive the crane faster when the hand-held device is farther from the trolley. The derivative component is a means to capture the movement speed of the hand-held device. If the the device is moving away from the trolley at a high speed, we wish for the crane to move faster in order to catch up.

We do not include an integrator (i.e. an "I" component, e.g. PID) in the controller because there is already an integrating component in the feedback loop, where the trolley velocity, V_T, is integrated to yield the trolley position. This integration is enough to ensure that the value of the steady state error, $e1$, is zero for a step input, according to the final value theorem. Additionally, due to sensor noise, the inclusion of an extra integrator within the controller will result in integrator windup, which can potentially drive the system to a limit cycle.

V_{PD} is passed through an artificial saturator (to prevent large amplitude signals from damaging the motors) to produce V_S, which is then sent to the "Motor, Trolley" and "Hook Dynamics" blocks in a similar fashion to that shown in Figure 34. Under this control architecture, the PD controller continuously drives the error signal $e1$ to zero. The resulting effect is that the crane trolley follows the hand-held device around the workspace.

3.2.1 Wand Control Operator Study

To investigate the effectiveness of the PD hand-motion controller, an operator study using the wand, as shown in Figure 3, was conducted. The feedback gains were set to 1.2 and 0.6 for the proportional and derivative gains, respectively. They were chosen by empirical means such that the motion of the hook was appropriately responsive to operator commands, while at the same time did not yield large and unstable oscillations. Twelve novice crane operators were included in the study. Each operator drove the crane through the obstacle course illustrated in Figure 37 using the following three control interfaces:

1. Pendent control where the direction labels written on the push buttons corresponded to the operator's orientation. For example, pushing the "LEFT" button yields a crane motion to the left relative to the operator.
2. Pendent control with *reversed* orientation. For example, pushing the "LEFT"

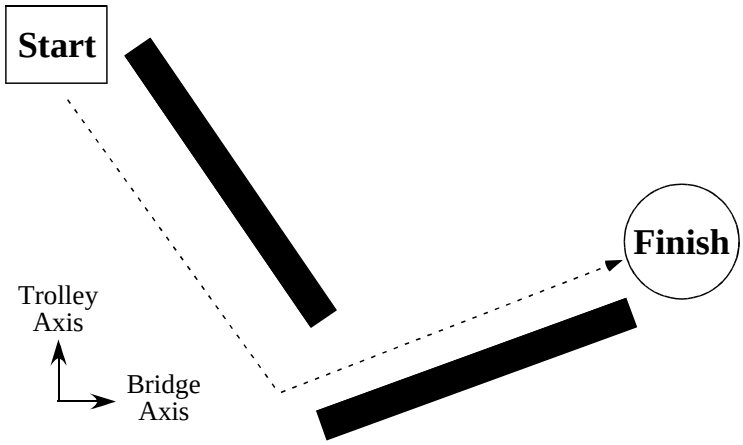


Fig 37. Obstacle Course for the Wand Operator Study

button yields a crane motion to the right relative to the operator. This control method is intended to study the effects of the change in the mental map that an operator must make when his or her orientation changes.

3. Wand control.

The goal was to move the crane hook from start to finish as quickly as possible without colliding with the barriers. The barriers in the obstacle course were about 4 feet in length and were arranged in a non-45-degree angle relative to the trolley and bridge axes of the bridge crane.

Figure 38 shows the course completion times for each operator. The pendent control with "reversed" orientation was the most difficult. It had an average completion time of over 100 seconds. The average completion time for wand control was approximately 50 seconds, which was half the time of the average time using normal pendent control. One-way ANOVA analysis of the data shows that the mean completion time of the wand control is significantly different from that of the pendent control methods. Furthermore, from a post-hoc Tukey analysis, there is a 95% confidence that the lower mean completion times using the wand are not due to chance.

Figure 39 shows the number of collisions each operator made while completing the obstacle course. Most operators had few or no obstacle collisions while using

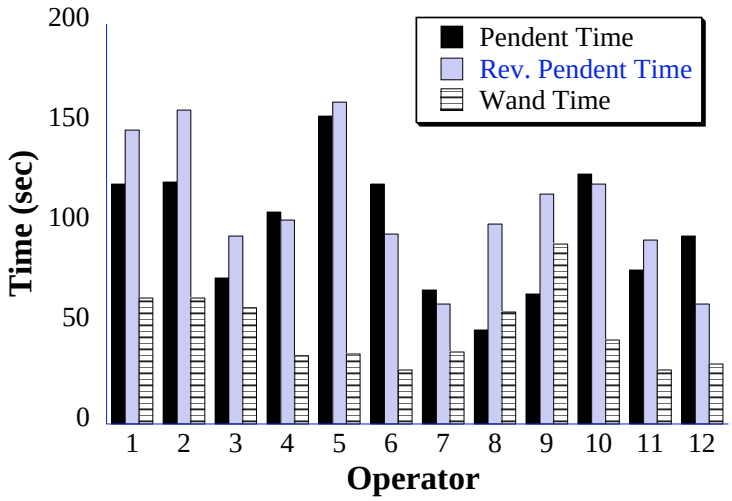


Fig 38. Obstacle Course Completion Times for the Wand Operator Study

wand control. The average number of collisions was only 0.92. With pendent control, obstacle collisions were nearly unavoidable. The average number of collisions with normal pendent control was approximately 5; with *reversed* orientation, the average increased to approximately 7. One-way ANOVA analysis of the data shows that the mean number of collisions for the wand control is significantly different from that of the pendent control methods. Furthermore, from a post-hoc Tukey analysis, there is a 95% confidence that the lower mean number of collisions using the wand is not due to chance.

The higher average completion time and average collisions under the *reversed* pendent control is an indication of the increased difficulty under this control interface. When the operator's orientation in the workspace is changed, they are forced to update their interface-to-crane-motion "mental map".

In order to understand why wand control achieves such superior performance, it is helpful to examine the time responses of the crane. Figure 40 shows the time responses of the trolley and hook in the bridge axis direction for three of the test subjects using normal pendent control. Figure 41 shows the data for the same operators in the trolley axis direction. Clearly, pendent control without oscillation compensation produces

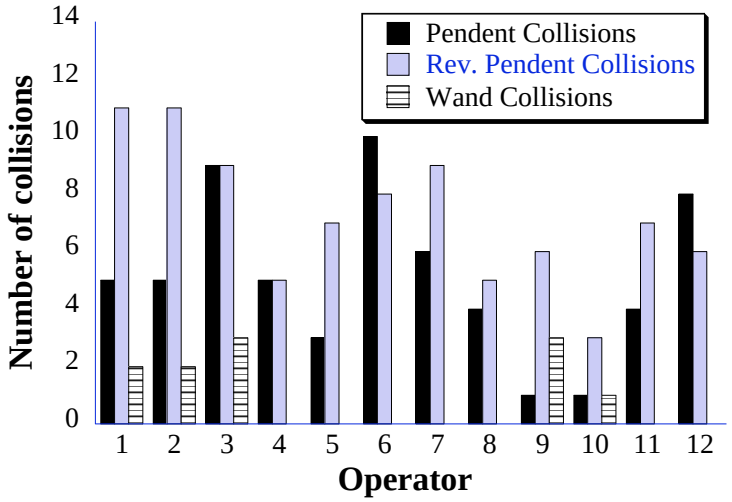


Fig 39. Obstacle Collisions for the Wand Operator Study

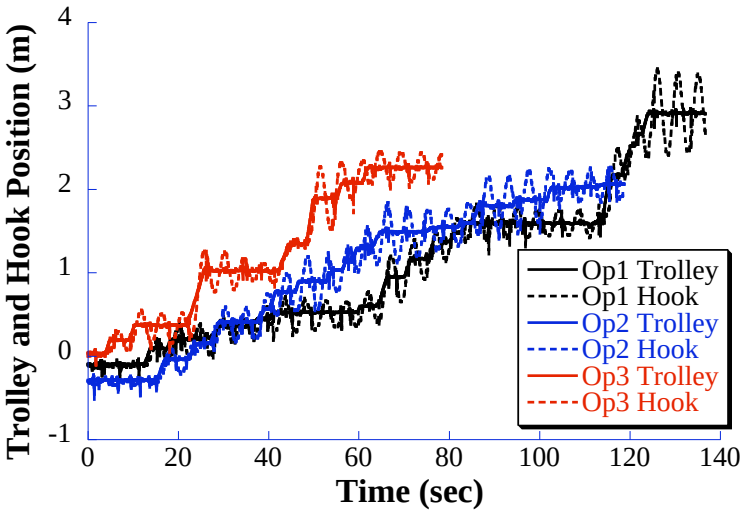


Fig 40. Trolley and Hook Response Using the Pendent, Bridge Axis Direction

significant hook oscillations.

Figure 42 shows the time response of the trolley, hook, and wand positions in the bridge axis direction for a typical case using wand control. The data has been plotted using the same scales as the corresponding data for normal pendent control in Figure 40. This format makes it immediately obvious that with wand control, the movement is completed much faster and with less hook oscillation.

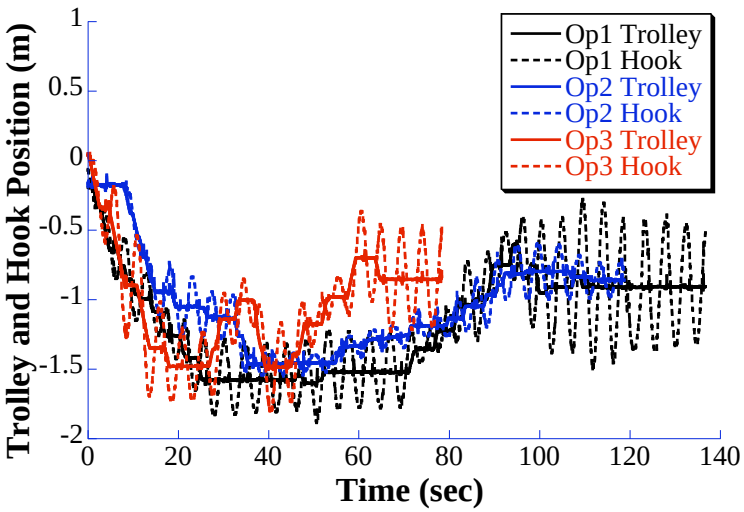


Fig 41. Trolley and Hook Response Using the Pendent, Trolley Axis Direction

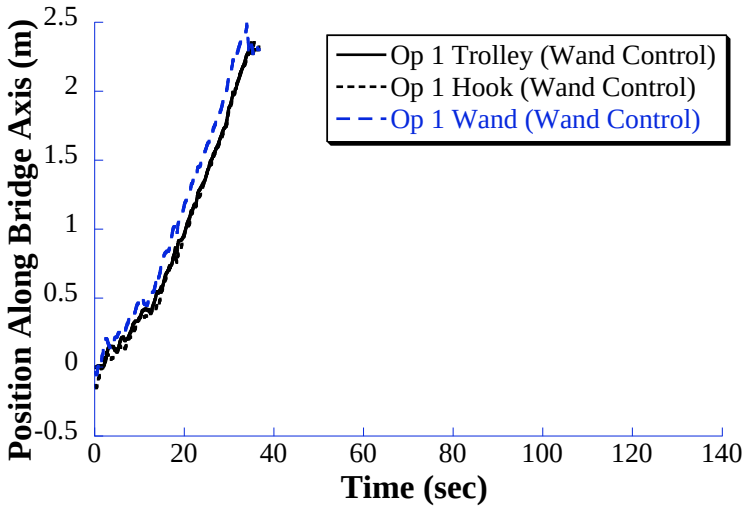


Fig 42. Trolley and Hook Response Using Wand, Bridge Axis Direction

Figure 43 shows a close-up view of the data. This plot clarifies the low oscillation amplitude and also shows the slight lag between the wand position and the trolley and hook. Figure 44 shows the same test subject's data in the trolley axis direction. In contrast to the pendent control responses that were shown in Figure 41, the amplitude of hook swing is small, yielding a smoother hook response.

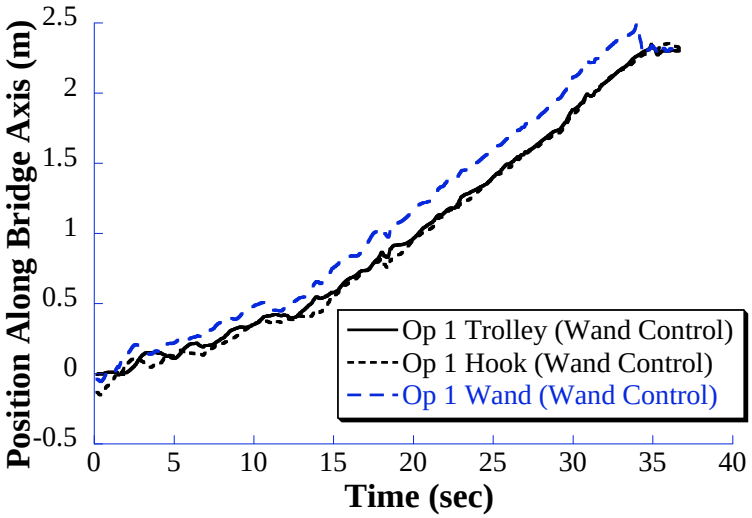


Fig 43. Close-up View of Trolley and Hook Response Using Wand, Bridge Axis Direction

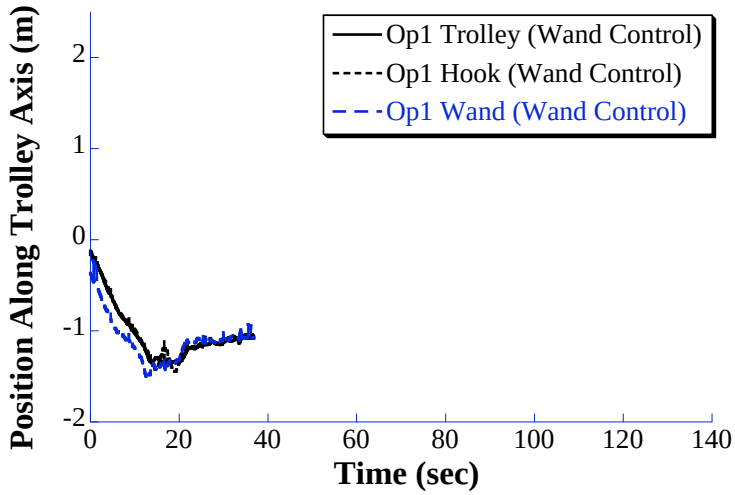


Fig 44. Trolley and Hook Response Using Wand, Trolley Axis Direction

Figure 45 shows typical two-dimensional hook responses from a single operator using both the normal pendent control and the wand control. It is evident that wand control allows the hook to be controlled more precisely. The reduction in the amplitude of hook swing and the overall smoothness of hook travel are major contributors to fewer obstacle collisions and more efficient navigation through the obstacle course.

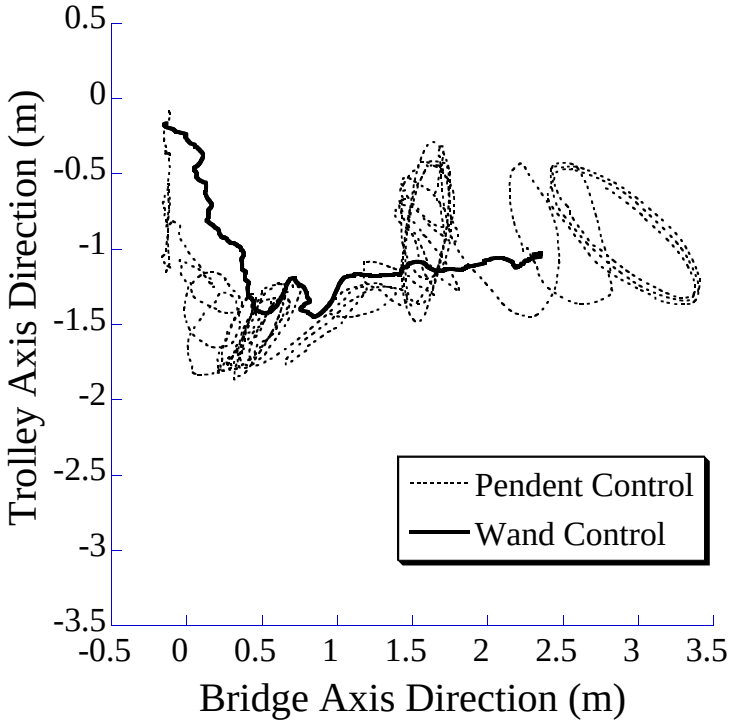


Fig 45. Overhead View of Hook response

As part of the study, operators were asked to fill out a questionnaire giving qualitative analysis of the crane's performance, as well as suggesting potential improvements to the system. It was unanimously agreed amongst the operators that it was more intuitive and easier to use the wand control interface in comparison to traditional push buttons. Furthermore, completion of the obstacle course, payload oscillation reduction, and ease of learning were all rated highly for wand control. One of the criticisms of the system was aimed at the PD controller that commands crane movements. Many operators found the response too sluggish. It was also noted that with the PD controller, it was possible to induce large hook oscillations by moving the wand in an oscillatory motion about the hook.

3.3 PD with Input-Shaping Hand-Motion Controller

While it is clear that the Proportional Derivative controller described in section 3.2 performs better than the standard pendent, it lacks a component designed specifically

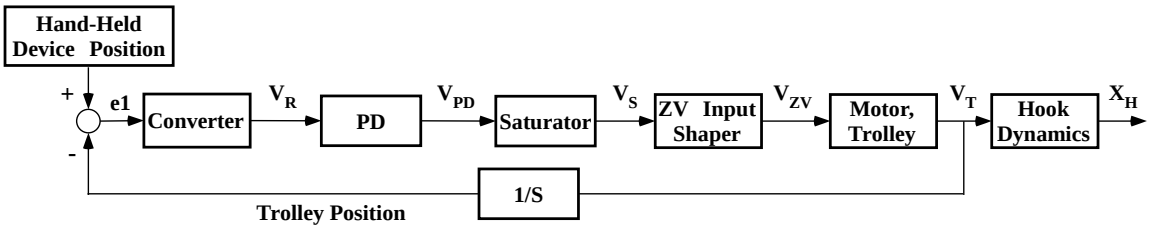


Fig 46. PD with ZV Input Shaper Hand-Motion Control Block Diagram

to reduce residual hook oscillations. To this end, input shaping was added to the control system. Figure 46 shows the block diagram for a controller with PD feedback and Zero Vibration (ZV) input-shaping components.

The control architectures shown in Figure 46 and Figure 36 are very similar. The only difference is the insertion of the "ZV Input Shaper" block. The saturated velocity signal, V_S, is passed to the input shaper block to produce the shaped velocity signal, V_{ZV}, which is sent to the motors and trolley.

As will be explained in the following sections, the idea behind the insertion of a ZV input shaper is to remove all energy at the oscillatory frequency from the command signal before it is issued to the plant. Thus, the "ZV Input Shaper" block was designed specifically to cancel the oscillatory frequency of the "Hook Dynamics" block.

The ordering of the blocks is important for successful oscillation reduction. In order for an input shaper to work *inside* the feedback loop, the final actuating signal (the quantity that enters the "Motor, Trolley" block) must preserve its oscillation-canceling properties. If the input shaper is placed anywhere before a non-linear element, such as the "Saturator" block, then the shaped signal may be corrupted before it reaches the "Motor, Trolley" block [8].

The other possibility is to place the input shaper *outside* the feedback loop, i.e. immediately after the "Hand-held Device Position" block, such that the shaped signal is created before it enters the feedback loop. The difficulty with this method, however, is that one would need to design an artificial saturator and place this between the

"Hand-held Device Position" block and input shaper blocks. This saturator would need to have appropriate saturation limits such that the shaped signal would not be corrupted by any of the non-linear elements (such as the "Converter" block and "Saturator" block in Figure 46) that are inside the loop. Therefore, placing the input shaper outside of the loop greatly increases the design complexity.

Furthermore, placing the input shaper outside the loop requires the absolute position of the hand-held device. Under wand and glove control, the position of the hand-held device is provided by the *trolley*-mounted camera. This means that for these two interfaces, the variable "e1" is measured directly by the camera, i.e. the hand-held device position is measured *relative* to the trolley. It is possible to use the camera measurement and the trolley positional measurement (from the laser range sensors) to reconstruct the absolute position of the hand-held device, however this needlessly complicates the system. Under RF-based control, the RF-sensors measure the absolute position of the RFID tag. Despite this, placing the input shaper outside the loop would still require a fundamentally different control architecture. The simplicity of using one controller architecture for all control interfaces is appealing.

This section begins with an introduction to input shaping, followed by an empirical design process that advances from a controller with only PD components, to a controller with PD *and* input-shaping components. Simulation and experimental evidence are included.

3.3.1 Input Shaping

This introduction to input shaping is based on the work from [37, 35, 41]. In the past, a variety of techniques such as input shaping and closed-loop feedback control have been developed for controlling the dynamic response of flexible systems. One such flexible system is a crane, with typical responses that were shown in Figure 35.

A feedback controller's inherent strength lies in the fact that it can detect errors and respond accordingly. Such a controller is well suited to precisely position the final location of a crane's bridge or trolley. If, however, the feedback controller must minimize hook oscillation, the control task becomes much more problematic. Accurate sensing of the hook must be implemented, which is often costly or difficult. When sensing of the hook is available, the feedback utilized to control the oscillation responds only if oscillation exists. In this way, the controller is inherently reactive.

Another technique used for negating a system's flexible modes is input shaping. Input shaping does not require the feedback mechanisms of closed-loop controllers. Instead, the control scheme reduces oscillations in an anticipatory manner, as opposed to the reactive manner of feedback. Vibration suppression is accomplished with a reference signal that anticipates an error before it occurs, rather than with a correcting signal that attempts to restore the system back to a desired state. In the context of crane control, this means that sensing of the hook oscillation is not necessary. As a result, input shaping is easier to implement than feedback control.

In general, input shaping is the process by which a reference command for a system is modified. The type of command modification depends on the desired effects on the system response. If the system exhibits flexible dynamics, then it is often desirable to modify the reference command so that the flexible modes will be minimally excited. In broad terms, input shaping involves modifying a reference command in such a way that resonant modes of a system combine destructively, resulting in low residual oscillation.

To demonstrate destructive interference, consider the system response of a lightly damped second-order system, such as a crane, to a series of two impulses, as shown in Figure 47. Figure 47 shows the system response to the impulses if they were each applied alone. Figure 48 shows the combined system response. These figures

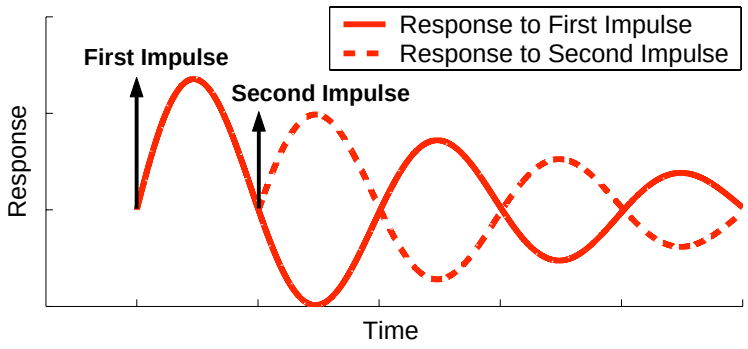


Fig 47. System Response to Each Impulse

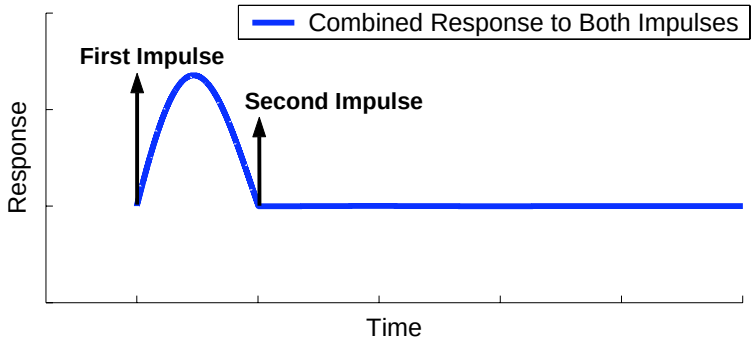


Fig 48. System Response to Both Impulses

demonstrate that when an impulse of appropriate magnitude is applied one-half-period after the first impulse, the responses of the system combine destructively, resulting in zero oscillation.

The input-shaping process is an extension of the concept illustrated in Figure 48 that can shape any arbitrary reference command to result in zero residual oscillation rather than just a sequence of impulses. This extension is made by convolving the sequence of impulses, known as an input shaper, with an arbitrary reference signal to produce a shaped command that is then used to drive the system. This process is illustrated in Figure 49. Once the sequence of impulses is convolved with the original command, the shaped command has the same oscillation-reducing properties as the original set of impulses.

The amplitudes and time locations of the impulses that will limit the unwanted

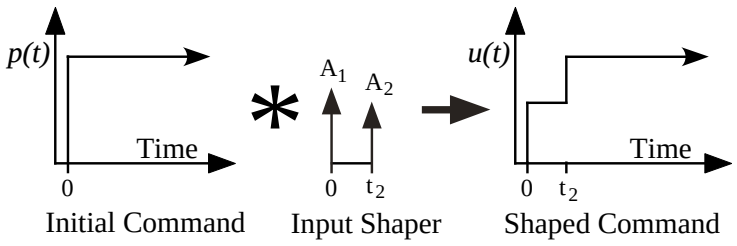


Fig 49. The Input Shaping Process Applied to a Reference Step Input

dynamic response are determined by solving a set of constraint equations. The constraint equations are usually categorized as residual oscillation constraints, robustness constraints, impulse constraints, and time optimality.

To constrain the residual oscillation, we need an expression for the residual oscillation amplitude as a function of the impulse sequence. If we assume the system can be modeled as a second-order harmonic oscillator, or a superposition of second-order systems, then we start with the response from a single impulse:

$$y_0(t) = \frac{A_0\omega}{\sqrt{1-\zeta^2}} e^{-\zeta\omega(t-t_0)} sin\left(\omega\sqrt{1-\zeta^2}(t-t_0)\right) \tag{40}$$

where A_0 is the amplitude of the impulse, t_0 is the time the impulse is applied, ω is the natural frequency, and ζ is the damping ratio.

The response from a sequence of impulses is just a superposition of the responses given in (40). Using the simplification:

$$\omega_d = \omega\sqrt{1-\zeta^2}, \tag{41}$$

the response to a sequence of impulses after the time of the last impulse is:

$$y_\Sigma(t) = \sum_{i=1}^{n} \frac{A_i\omega}{\sqrt{1-\zeta^2}} e^{-\zeta\omega(t-t_i)} sin\left(\omega_d(t-t_i)\right), \tag{42}$$

where A_i and t_i indicate the amplitude and time of the i^{th} impulse and n is the total number of impulses.

Given (42), an expression for the amplitude of oscillation can be formed by using the trigonometric identity:

$$\sum_{i=1}^{n} B_i sin(\omega t + \phi_i) = A_{\sum} sin(\omega t + \psi), \tag{43}$$

where,

$$A_{\sum} = \sqrt{\left(\sum_{i=1}^{n} B_i cos(\phi_i)\right)^2 + \left(\sum_{i=1}^{n} B_i sin(\phi_i)\right)^2}. \tag{44}$$

The expression for the phase shift, ψ, is unimportant for our purposes here and ϕ_i is the argument of the sine term in (42). From the expression in (42):

$$B_i = \frac{A_i \omega}{\sqrt{1-\zeta^2}} e^{-\zeta\omega(t-t_i)}. \tag{45}$$

To obtain the residual oscillation amplitude, we evaluate (44) at the time of the last impulse, $t = tn$. Substituting (45) into (44) and bringing the constant portion of the coefficients out of the square root term yields:

$$A_{\sum} = \frac{\omega}{\sqrt{1-\zeta^2}} e^{-\zeta\omega t_n} \sqrt{[C(\omega,\zeta)]^2 + [S(\omega,\zeta)]^2}, \tag{46}$$

where,

$$C(\omega,\zeta) = \sum_{i=1}^{n} A_i e^{\zeta\omega t_i} \cos(\omega t_i \sqrt{1-\zeta^2}) \tag{47}$$

$$S(\omega,\zeta) = \sum_{i=1}^{n} A_i e^{\zeta\omega t_i} \sin(\omega t_i \sqrt{1-\zeta^2}). \tag{48}$$

To form a non-dimensional vibration amplitude, (46) is divided by the amplitude of residual vibration from a single impulse of unity magnitude. The resulting expression gives the ratio of vibration with input shaping to that without input shaping. This percentage residual vibration (PRV) is given by [16]:

$$PRV = V(\omega,\zeta) = e^{-\zeta\omega t_n} \sqrt{[C(\omega,\zeta)]^2 + [S(\omega,\zeta)]^2}, \tag{49}$$

Equation (49) represents the level of vibration induced by an impulse sequence given any value of frequency and any damping ratio less than one. A constraint on

residual vibration amplitude can be formed by setting (49) less than or equal to a tolerable level of residual vibration at the modeled natural frequency and damping ratio. For the simplest Zero Vibration (ZV) shaper, the tolerable amount of vibration is set to zero. Additionally, the magnitudes of all impulses are required to be positive and the number of impulses minimized. This results in a shaper of the form [37, 35]:

$$ZV = \begin{bmatrix} A_i \\ t_i \end{bmatrix} = \begin{bmatrix} \frac{1}{1+K} & \frac{K}{1+K} \\ 0 & \frac{\pi}{\omega\sqrt{1-\zeta^2}} \end{bmatrix}, i = 1, 2 \tag{50}$$

where,

$$K = e^{\frac{-\zeta\pi}{\sqrt{1-\zeta^2}}}. \tag{51}$$

To gain more insight, one can analyze the ZV shaper's performance with the use of a sensitivity curve, as shown in Figure 50. The sensitivity curve for a ZV shaper is shown by the solid line. The sensitivity curves for the Zero-Vibration-Derivative (ZVD) and the Extra-Insensitive (EI) shapers are also shown. The vertical axis is the Percent Residual Vibration (PRV) and the horizontal axis is the actual natural frequency, ω, normalized by the modeled frequency, ω_m, which is used to design the input shaper. The curve indicates how residual vibration amplitude changes as a function of modeling errors in frequency. While a sensitivity curve itself is not a measure of robustness, a qualitative picture of the robustness of a command can be obtained from it and quantitative measures can be extracted from it. One key quantitative measure of robustness derived from the sensitivity curve is Insensitvity. Insensitivity is the width of the sensitivity curve at a tolerable vibration level, V_{tol}, with respect to the parameter of interest. For example, Figure 50 shows the ZV shaper has an Insensitivity at $V_{tol} = 5\%$, $I(5\%)$, of 0.06.

Figure 35 showed that the magnitude of residual oscillations to a standard pendent control (without any form of oscillation-compensation control) is dependent on the

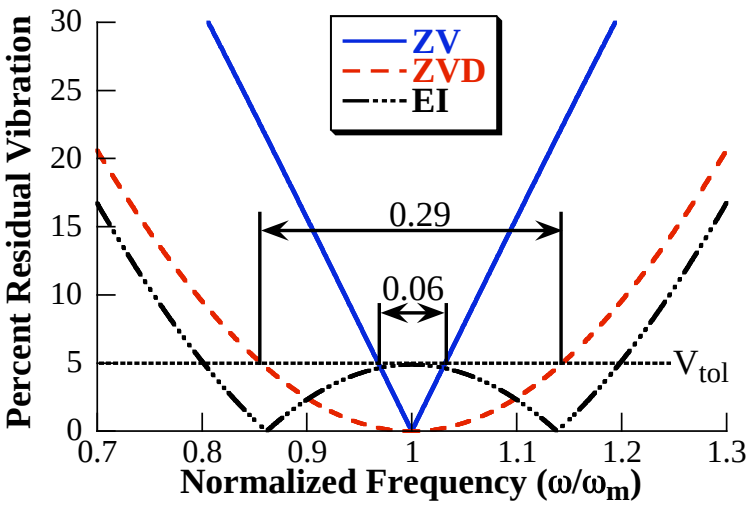


Fig 50. Sensitivity Curve for the ZV, ZVD, and EI Shapers

move distance. With a correctly designed input shaper however, reduced residual oscillations are guaranteed for all move distances.

3.3.2 Simulation Results of PD, and PD with Input-Shaping Controllers

We now discuss an empirical approach based on engineering rules of thumb to design the PD hand-motion controller and the PD with input-shaping controller. We begin with the PD controller, discuss its limitations, and proceed to PD with input shaping. The results presented used the glove control interface.

A model was constructed using Matlab Simulink to simulate the control architectures in Figure 36 and 46. Here, we assume the operator uses the glove control interface, shown in Figure 4. The crane model used a hook cable length of 5 meters. Glove trajectories were specified as ramps with gradients equivalent to the maximum velocity of the Georgia Tech 10-ton crane (0.3577 m/s) and ending at a distance of 2 meters.

Figure 51 shows the simulation results for the PD hand-motion controller with low gains (P=2, D=0). The response is sluggish, with a 10% to 90% rise time of nearly 9.3 seconds and a 2% settling time of around 55 seconds. The maximum percentage overshoot is approximately 2%. The amplitude of residual oscillations is also low.

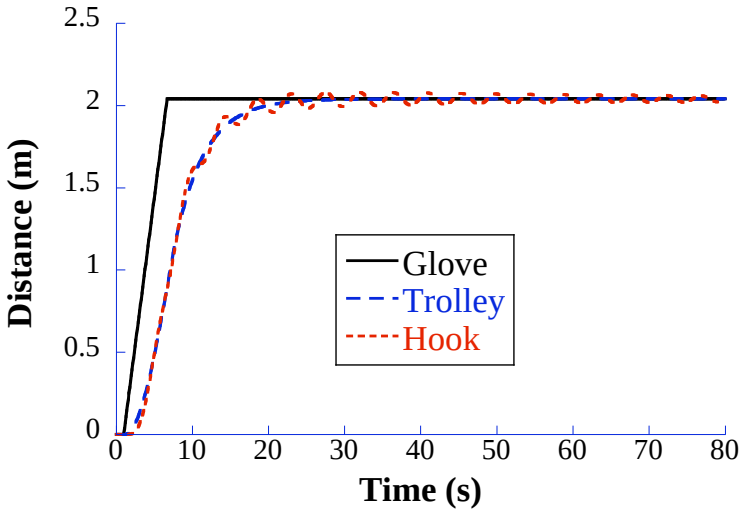


Fig 51. Simulation Result of the PD Controller with Low Gains

To increase the responsiveness of the hand-motion controller, the gains were increased so that the motors saturated and the trolley was able to move at its maximum velocity. Figure 52 shows the response for the hand-motion control with increased gains (P=15, D=10). Under this set of gains, the crane is more responsive; as the 10% to 90% rise time is reduced by 53% to only 4.4 seconds. However, hook oscillations are now more significant, as the maximum percentage overshoot has increased by more than eight-fold to 16.6%. Due to the large overshoot and lightly-damped nature of the hook dynamics, the 2% settling time has increased by more than 100% to 119 seconds.

It can be seen that there is an inherent trade-off with the PD hand-motion controller. Low gains do not tend to induce large hook oscillation amplitudes, but the response is sluggish. However, high gains have a faster response at the cost of large amplitude hook oscillations. The goal of combining PD feedback with a ZV input shaper is to create a controller with the desirable features of fast response and low residual oscillations. The ZV input shaper was designed to cancel the oscillatory nature of the suspended hook at a suspension length of 5 meters.

Figure 53 shows the improved response of the combined controller. The 10% to

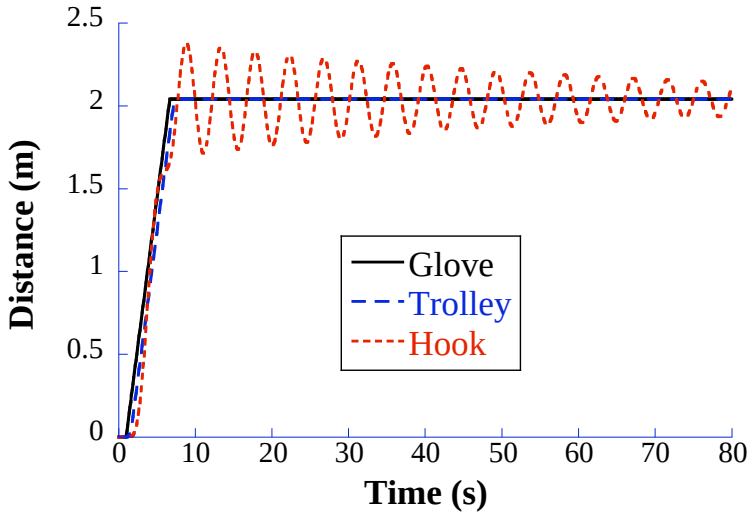


Fig 52. Simulation Result of the PD Controller with High Gains

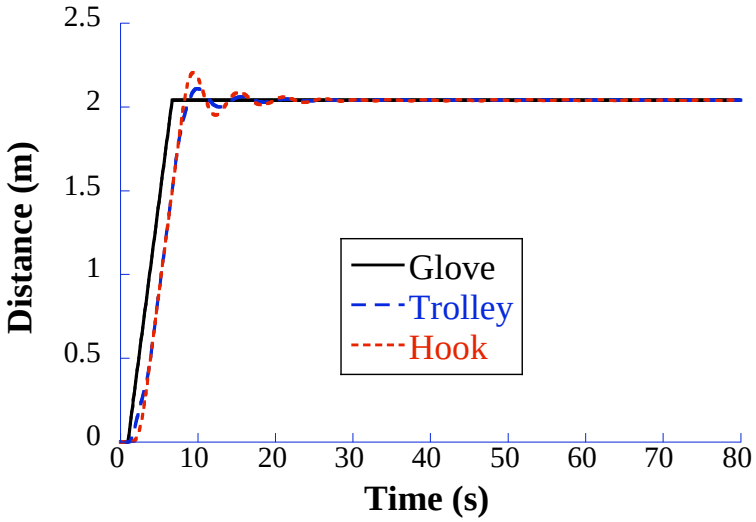


Fig 53. Simulation Result of the High-Gain PD with ZV Shaper Controller

90% rise time remains largely unchanged at 4.6 seconds, while the 2% settling time is reduced by ten-fold to only 18 seconds, which is much faster than hand-motion control with low PD gains. The maximum percentage overshoot is only 7.8%.

3.3.3 Experimental Results of the PD, and the PD with Input-Shaping Controllers Using the Glove Interface

The hand-motion control system was implemented on the Georgia Tech 10-ton bridge crane. The goal was to control the movement of the hook, with no payload attached. This single pendulum system is representative of an important subset of crane applications. The glove trajectories produced by the human operator were similar to those used in the simulations. The ramp gradient was roughly equivalent to the maximum velocity of the crane and the move distance was approximately 2 meters.

Note that when the crane approaches the desired stopping location, the crane operator drops his or her hand to stop the crane. When the hand is dropped, the camera can no longer detect the presence of the glove. As a result, the control system reacts by setting $e1$ to zero. This action will manifest itself as a spurious artifact whereby the glove position discontinuously becomes equivalent to the trolley position at the time at which the glove is dropped. Also, to accelerate the crane, the operator does not need to be at zero distance (right next to the hook). He or she can expose the glove to the camera at some distance away to initiate movement. Due to these operational effects, the glove position contain discontinuous artifacts at the starting and stopping stages of the motion.

Figure 54 shows the experimental results for the PD hand-motion controller with a low proportional gain (P=2, D=0). The 10% to 90% rise time is around 10 seconds, the maximum percentage overshoot is about 5%, and the 2% settling time is 50 seconds. Note that the glove was not moved through a perfect ramp in position, as in the simulations. The operator was asked to move the crane in a point-to-point motion, while moving the glove at a relatively constant speed. The operator did move the glove at a roughly constant velocity during most of the test, but prematurely stopped the glove before moving it rapidly to the final target location. This type of motion is to be expected from real human operators and it occurred in all of the tests. However,

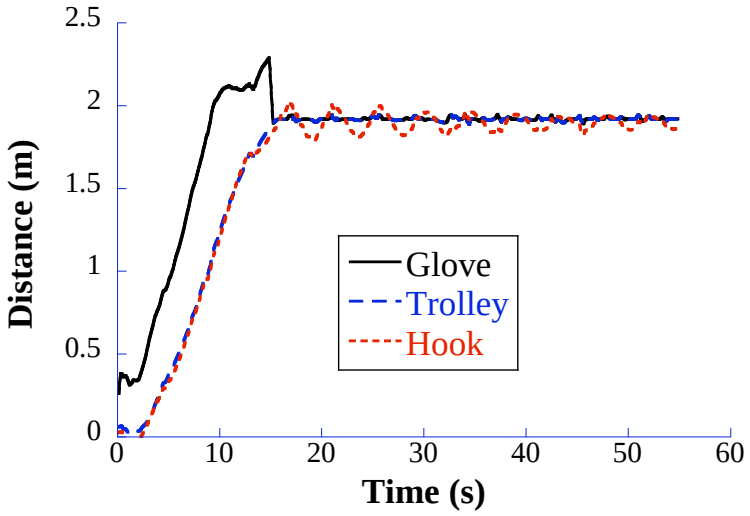


Fig 54. Experimental Result of the PD Controller with Low Gains

this uneven command profile has very little effect on the performance of the control system.

Figure 55 shows the experimental response of PD hand-motion control with high gains (P=15, D=10). The 10% to 90% rise time was reduced by over 40% to 5.8 seconds, but the maximum percentage overshoot doubled to 10%. The hook took longer than 53 seconds to settle within 2% of the desired stop location. Note that with the higher gains, the hook tracks the glove position much more closely than with the low gains shown in Figure 54.

The experimental results using the PD hand-motion controllers clearly demonstrate the expected trade-off between using high gains and low gains. The use of a PD controller with high gains will reduce rise time at the expense of increased overshoot and settling time.

Figure 56 shows the experimental response of the combined PD and ZV-shaping controller. The 10% to 90% rise time is 4.8 seconds, and there is virtually no overshoot or residual vibration. For this reason, the 2% settling time is approximately 8 seconds (an 84% improvement over PD with low gains).

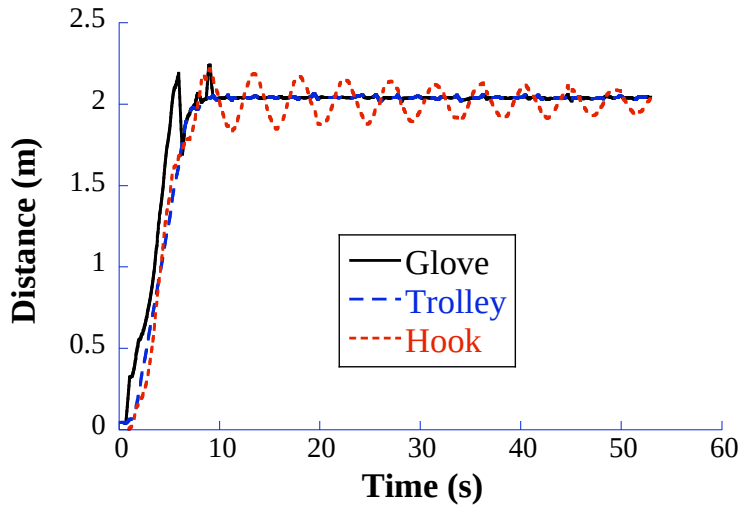


Fig 55. Experimental Result of the PD Controller with High Gains

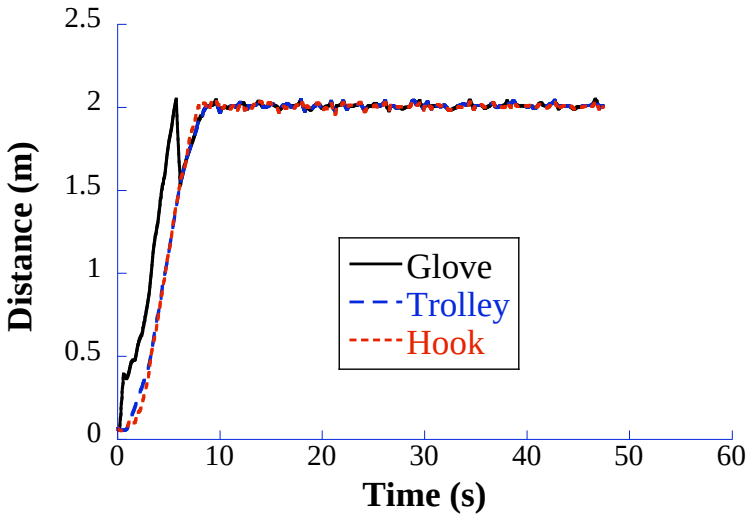


Fig 56. Experimental Result of the High-Gain PD with ZV Shaper Controller

Clearly, the experimental and simulation results demonstrate that the hand-motion controller with high PD gains and ZV shaper is able to produce a hook response that is fast and without significant overshoot and residual vibration.

It is impossible to experimentally reproduce the perfect ramp input that was used in simulations, and there is no reason to do so as real operators do not use such perfectly formed commands. Therefore, the glove trajectories from the experiments

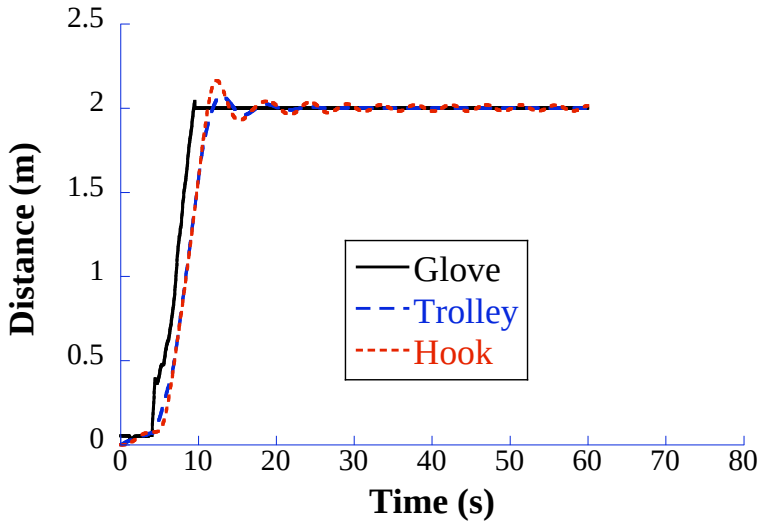


Fig 57. Simulation Result Using the Glove Trajectory from Experiment

were used to drive the crane in simulation to provide further comparison between theoretical and experimental results. Figure 57 shows the simulation results when the glove position of Figure 56 is used to drive the crane. Note that the discontinuous artifact caused by the operator dropping his hand was removed. As expected, there is virtually no residual oscillations. It should be noted that the goal of this research is to design a controller centered around the human operator. Therefore, to assess and validate controller performance, simulation results were used as a guiding reference, while actual experimental data were more highly valued.

To investigate the robustness of the controller on the real crane, similar movements were repeated for hook cable lengths of 4 meters and 6 meters, without redesigning the input shaper. This corresponds to an increase and decrease of cable length by 1 meter from the original 5 meter cable length, for which the input shaper was designed. The experimental results for the 4m and 6m cable lengths are shown in Figures 58 and 59, respectively.

Clearly, the controller still displays effective suppression of residual oscillations even when the cable lengths are changed. To gain more insight, one can analyze the ZV shaper's performance with the use of a sensitivity curve, which was shown in

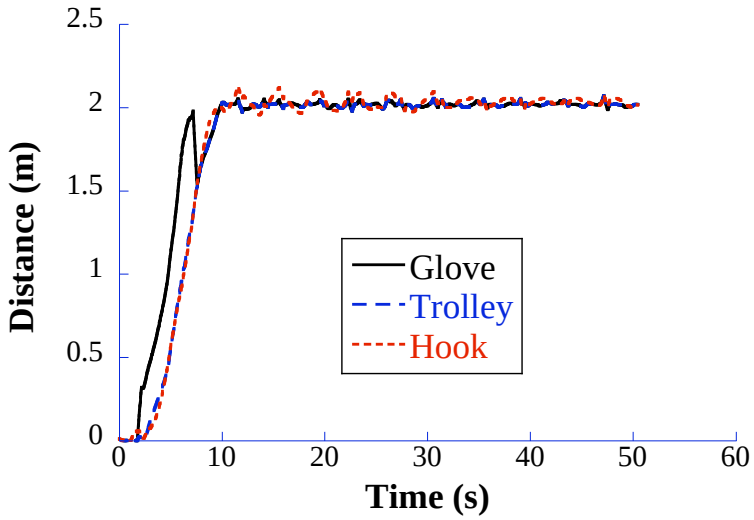


Fig 58. PD with ZV Shaper Controller with 4 Meters Hook Cable Length

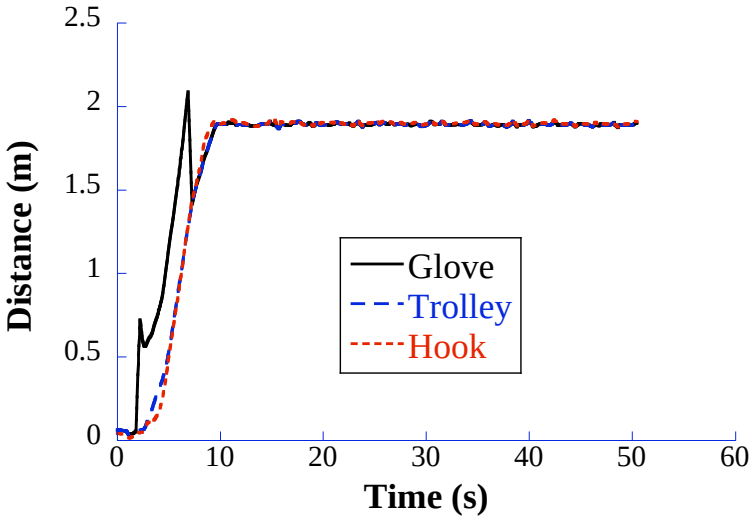


Fig 59. PD with ZV Shaper Controller with 6 Meters Hook Cable Length

Figure 50.

For this specific application, the ZV shaper designed for the 5 meters cable length uses a modeled frequency, ω_m, of $1.4 rad/s$. When the cable length is changed to 4 meters and 6 meters, the actual natural frequency, ω, changes to $1.57 rad/s$ and $1.28 rad/s$, which corresponds to a normalized frequency of 1.12 and 0.91, respectively. Referring to the ZV sensitivity curve in Figure 50, it can be seen that this is still

reasonably good performance, as the PRV for both cases are under 20%. The case of the 4 meter cable length has a slightly higher PRV, which explains the presence of the visible, but nevertheless small amount, of residual oscillations in Figure 58. If the crane was expected to undergo large changes in cable length, then more robust shapers such as the ones (ZVD, EI) shown in Figure 50 should be used in the controller[35, 36].

3.3.4 Experimental Results of the PD with Input-Shaping Controller Using the RF-based Interface

Using the same PD with input-shaper controller, we now present experimental results using the RF-based interface to control the crane. The experimental results presented thus far were based on hand-motion input trajectories that were short in move distance and traversed in only one direction. It is of interest to examine the crane response to a hand-motion trajectory that is more complex - one that lasts longer and traverses through the horizontal plane.

The experiment was conducted using the unfiltered RFID tag location data. The tag was moved at roughly walking speed in the horizontal plane around the edges of a table with dimensions of 0.77m by 2.44m. The tag was held at the corners of the table for short periods of time to allow the crane to catch up. The height of the tag in the vertical direction was not considered as the tag was assumed to move only in the horizontal plane.

Figure 60 shows the overhead view for one lap around the table. The graph shows the actual tag/hand positions (corresponding to the table edge), the measured tag/hand positions, and the crane hook response. The sensor noise is fairly large as the average and standard deviation of the distance error (obtained by taking the difference between the measured tag location and the nearest point on the table) were 10.39 cm and 10.41 cm, respectively. Note that despite the highly-fluctuating tag-location measurements, the crane does not respond in a similar manner. This is because the crane trolley can be approximated as a second-order mass-friction

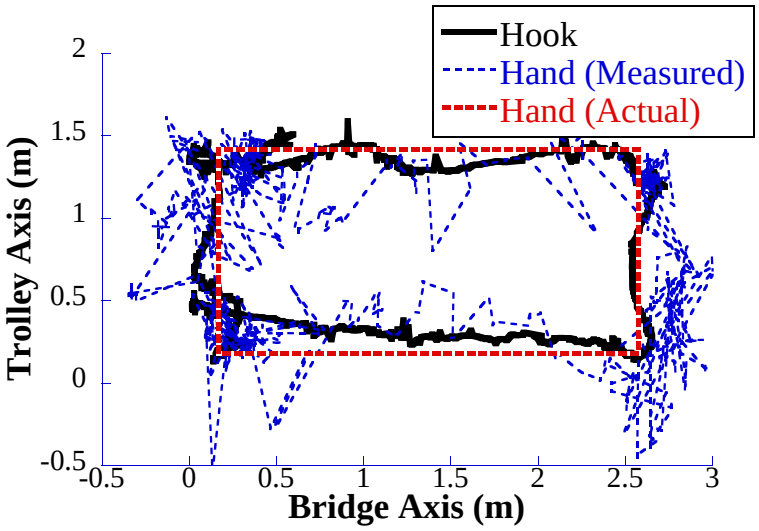


Fig 60. Overhead View of the RFID Tag Trajectory and Hook Response

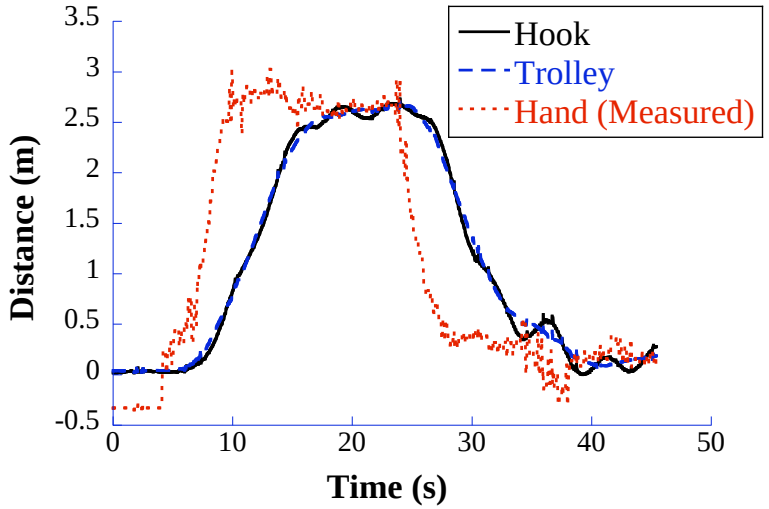


Fig 61. RF-Based Control Crane Response along the Bridge Axis

system that has low-pass filtering properties. Nevertheless, achieving more accurate and consistent tag location data would be desirable, especially when the operator desires small or precise crane movements.

Figure 61 shows the trolley position, hook position, and the measured hand/tag position along the bridge axis. Inevitably, the hook and trolley lag behind the hand, as a non-zero value of e1 is needed to drive the system. However, the controller ensures smooth tag-following crane movements with suppressed residual hook oscillations. It

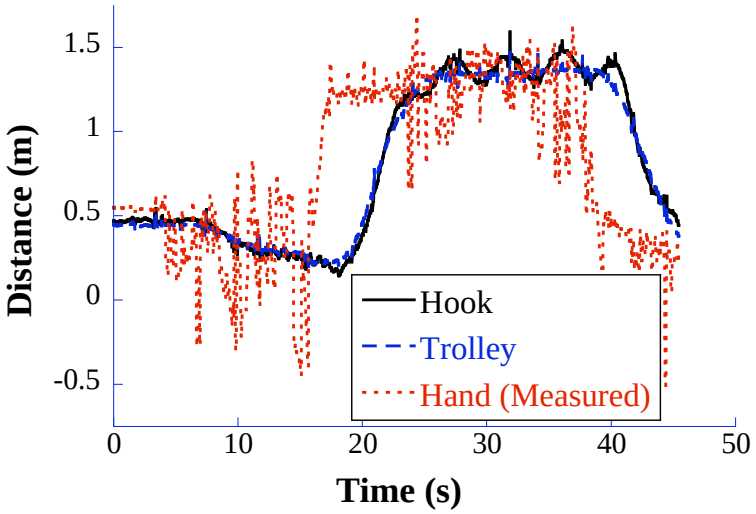


Fig 62. RF-Based Control Crane Response along the Trolley Axis

can be seen that the shape of the measured hand position is similar to the path of the hook and trolley. Figure 62 shows similar information for the trolley axis.

The average and standard deviation of the absolute distance error between the hook and the measured hand positions are 0.54m and 0.57m, respectively, for the bridge axis. For the trolley axis, these are 0.32m and 0.31m, respectively. However, if the spatial error is considered (calculated by shifting the hook response forward in time to overlay the hand position, and then forming the absolute difference), then the average and standard deviation are only 0.25m and 0.21m, respectively, for the bridge axis. For the trolley axis these values are 0.17m and 0.16m, respectively. Spatial error disregards the lag in hook response that is necessary for the controller, and is an indicator of the crane's ability to follow the *shape* of the desired trajectory.

The experimental results demonstrate good performance, as the crane hook is able to follow the desired trajectory around the edges of the table. However, there are issues that need to be addressed, such as the sensor noise. Work is in progress to design appropriate filters for the raw tag-location data.

3.4 Root Locus for Closed Loop Input Shaping

In section 3.3, we noted that the design process of the PD with input-shaping controller was based on empirical means. The controller in question was shown in Figure 46. Although the input shaper was designed analytically to reduce the oscillation of the hook; the PD gains were chosen based on engineering rules of thumb. What is lacking is a rigorous, theoretical-based design methodology for the selection of appropriate PD gains.

Furthermore, little consideration was given to the stability of the feedback control with input shaping in the loop. Stability is important because input shapers utilize partial time delays. That is, input shapers delay a portion of the signal they are shaping. The remainder of the signal is not delayed, although it may be scaled. Considering how full time delays affect closed-loop stability, the use of partial delays certainly presents a stability question.

This section will describe a method of using the root locus to analyze the stability of systems containing closed loop input shapers. Furthermore, the root locus can be used as an approximate guideline for the selection of stable PD gains. This work utilizes the developments in John Huey's PhD thesis [7]. The numerical method used for drawing the root locus of systems with time delays is based on the work of Nishioka et al. [28].

3.4.1 Pivoting Algorithm for the Root Locus of Linear Systems with Time Delay

In the Laplace domain, a generic input shaper may be represented by the following:

$$I(s) = A_1 + \sum_{i=2}^{n} A_i e^{-st_i} \tag{52}$$

where A_i and T_i are the amplitude and time of the i^{th} impulse. Given that the transfer function of a system with an input shaper inside the feedback loop contains terms with e^{-st_i}, it follows that a numerical method to solve for the root locus is preferred

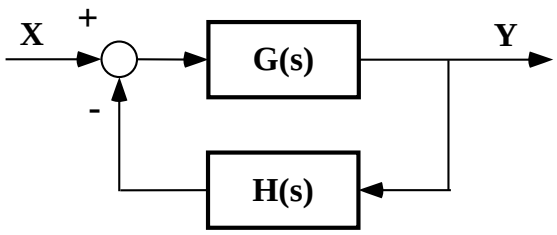


Fig 63. Feedback control system

over one that relies on a closed form solution.

Nishioka et al. described a simple algorithm for deriving the root locus for linear systems with time delays [28]. The general procedure will be briefly explained here. Consider the linear feedback control system shown in Figure 63. The closed-loop transfer function of the system is:

$$Y(s)/X(s) = G(s)/(1 + G(s)H(s)). \tag{53}$$

Suppose that the open-loop transfer function contains a time delay similar to an input shaper and can be written as:

$$G(s)H(s) = k\frac{Q(s)}{P(s)}exp(-\mu s) \tag{54}$$

where $s \in \mathbb{C}$ (the set of complex numbers), k is the controller gain, μ is the delay time, and $Q(s)$ and $P(s)$ are polynomials of s of degree n and r, respectively, where $n > r$. The poles of the closed loop system are given by the roots of the characteristic equation, which can be written as:

$$1 + G(s)H(s) = 0 \tag{55}$$

$$P(s) + kQ(s)exp(-\mu s) = 0. \tag{56}$$

Rearranging (56) for k:

$$k = -\frac{P(s)}{Q(s)}exp(\mu s). \tag{57}$$

Here, we introduce the complex function $K(s)$ that is equivalent to the right hand side of (57):

$$K(s) = -\frac{P(s)}{Q(s)} exp(\mu s). \tag{58}$$

The root locus is the trace of the roots of (56) when k is increased from zero to infinity. Put differently, the root locus is the union of the following two sets:

1. The set $s \in \mathbb{C} - Q^{-1}(0)$ such that equation (58) is satisfied, and

2. $Q^{-1}(0)$, where $Q^{-1}(0) = \{s : Q(s) = 0\}$.

It is necessary to break the root locus into two sets because the function $K(s)$ is not analytic on the region $Q^{-1}(0)$. It should be noted that the second set, $Q^{-1}(0)$, are the roots of (56) when $k = \infty$. It is also equivalent to the set of open-loop zeros. Expressed mathematically, the root locus is the set:

$$\Gamma = \{s \in \mathbb{C} - Q^{-1}(0)\} \cup \{Q^{-1}(0)\}. \tag{59}$$

However, for real-world controller gains, we assert the condition that $K(s)$ must be real and greater than or equal to zero. That is, $k \geq 0; k \in \mathbb{R}$. Therefore the real-world root locus is expressed as:

$$\Gamma_{real} = \{s \in \mathbb{C} - Q^{-1}(0); Im(K(s)) = 0, Re(K(s)) \geq 0\} \cup \{Q^{-1}(0)\}. \tag{60}$$

A corollary from (60) is that for any two points on the s-plane, $s1 \in \mathbb{C}$ and $s2 \in \mathbb{C}$, if $(Im(K(s1)) \geq 0, Re(K(s1)) \geq 0)$, and $(Im(K(s2)) < 0, Re(K(s2)) \geq 0)$, then the line segment connecting points $s1$ and $s2$ intersects Γ_{real}. This idea is illustrated in Figure 64 where the s-plane is mapped to the K-plane. The root locus that lies between $s1$ and $s2$ in the s-plane is mapped to the positive $Re(K(s))$ axis on the K-plane. It follows that if $s1$ and $s2$ are sufficiently close, the root locus can be approximated by the midpoint of the segment connecting the two points.

When mapped to the K-plane, Nishioka's method of numerically drawing the root locus is a "pivoting" process of determining the values of s where $Re(K(s))$ is

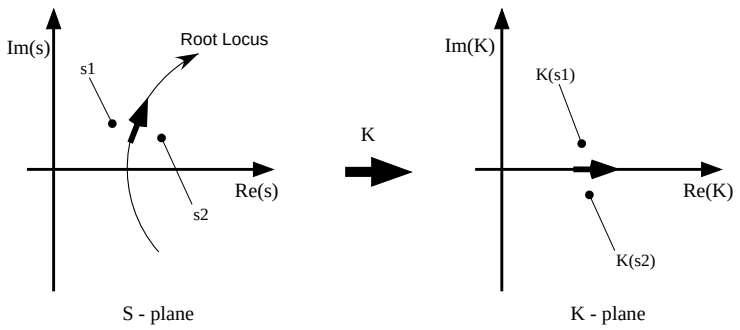


Fig 64. Mapping from the s-plane to the K-plane

crossed. The algorithm for plotting the approximated root locus is summarized in the following:

1. Construct a closely-spaced uniform grid of points in the s-plane
2. For each pair of neighboring points, $s1$ and $s2$, evaluate equation (58)
3. If $Im(K(s1)) \geq 0, Re(K(s1)) \geq 0$ and $Im(K(s2)) < 0, Re(K(s2)) \geq 0$, then s_{mid}, the midpoint that lies on the segment connecting $s1$ and $s2$, is a point on the approximate root locus, Γ_{apprx}
4. Plot the closed-loop poles and open-loop zeros

3.4.2 ZV Input Shaper in the Laplace Domain

Now that a tool is available for generating the root locus of systems with time delays, we focus on analyzing the input shaper in the Laplace domain.

An input shaper in the time domain is a series of i impulses with amplitudes given by A_i. The first impulse has amplitude A_1 and occurs at $t = 0$. The times at which subsequent impulses occur after the initial impulse is given by t_i. Thus, the input shaper can be thought of as a sequence of time-delayed impulses. A ZV input shaper, such as the one in (50), has two impulses, and is written in the Laplace domain as:

$$IS_{ZV}(s) = A_1 + A_2 e^{-st_2}. \tag{61}$$

Using $s = \sigma + j\omega$, the zeros of IS_{ZV} can be determined by setting (61) to zero and separating the second term on the right hand side into real and imaginary parts:

$$IS_{ZV}(s) = 0 = A_1 + A_2 e^{-\sigma t_2} e^{-j\omega t_2}. \tag{62}$$

By collecting the real and imaginary parts, and noting that $e^{-\sigma t_2}$ must be positive and real, the following conditions must be satisfied:

$$A_1 = A_2 e^{-\sigma t_2} \tag{63}$$

$$-1 = e^{-j\omega t_2}. \tag{64}$$

Solving for σ and ω, the locations of the shaper zeros are:

$$\omega = \frac{\pm n\pi}{t_2} n = 1, 3, 5... \tag{65}$$

$$\sigma = -\frac{1}{t_2} ln\left(\frac{A_1}{A_2}\right) \tag{66}$$

where σ and ω correspond to the real and imaginary coordinates in the Laplace domain.

A ZV shaper will cancel a system's flexible poles if the location of its zeros are the same as that of the poles. In another words, if the system has natural frequency ω_{sys} and damping ratio ζ_{sys}, then A_1, A_2, and t_2 must be set such that they satisfy (65) and (66), with ω set to $\omega_{sys}\sqrt{1-\zeta^2}$, and σ set to $-\zeta_{sys}\omega_{sys}$.

It is important to note from (65) that a ZV input shaper has an infinite number of zeros, as illustrated in Figure 65. This means that the shaper produces an infinite column of zeros when plotted in the s-plane, with the first set of zeros ($n = 1$) usually set to cancel the poles of the flexible system. It is also important to note that a ZV shaper has no finite, open-loop poles, because from (61), $IS_{ZV} = \infty$ only if the real part of s equals $-\infty$.

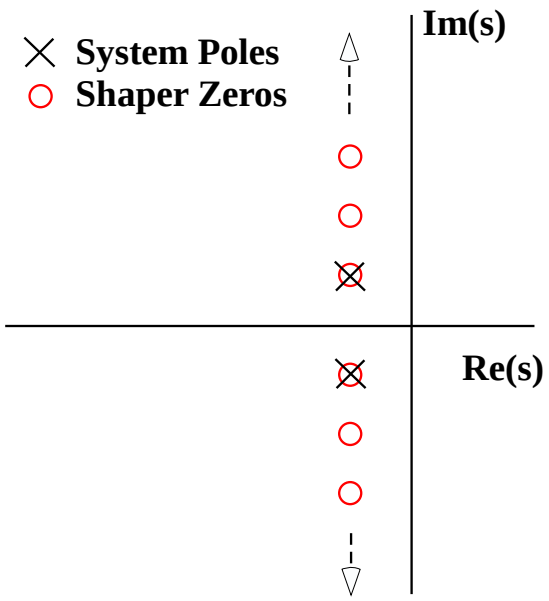


Fig 65. Shaper Zeros Canceling Flexible Poles

3.4.3 Root Locus of the PD with ZV Input-Shaping Hand-Motion Controller

Before plotting the root locus of the PD with ZV input-shaping controller from Figure 46, it must first be modified. The resultant control block diagram is shown in Figure 66.

Firstly, the feedback loop in question is indicated by the dashed box, i.e. we are interested in the closed-loop system given by $\frac{V_T(s)}{Hand-HeldDevicePosition(s)}$. The "Hook Dynamics" block is disregarded since it is outside of the feedback loop. Although it may seem unwise to ignore the hook dynamics (consider the case where the trolley enters a limit cycle that has a frequency which resonates with the hook swing), the root locus analysis still holds value because the stability of the trolley motion is taken into account.

As will be explained later in this section, the value of this analysis using root locus is not to determine the precise values of PD gains that will yield the best crane response. Rather, the intent is to determine the approximate gains that will provide stable trolley motion. Using these approximate gains as starting points, other gains

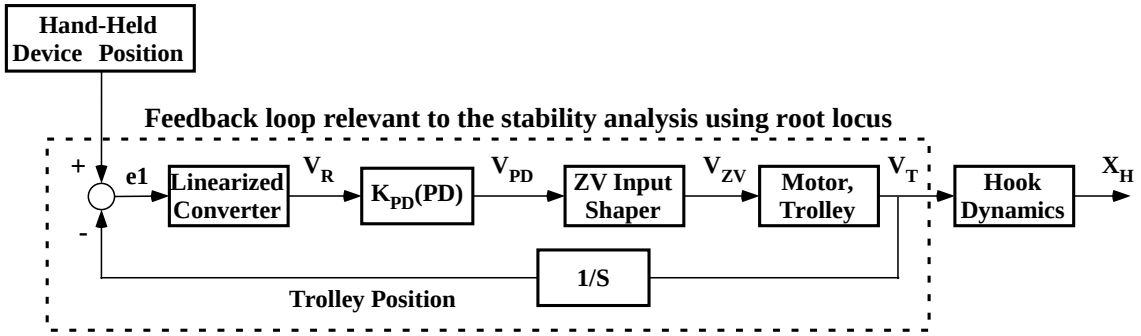


Fig 66. Linearized PD with ZV Input Shaper Hand-Motion Control Block Diagram

that yield a more desirable hook response can then be determined using iterative methods. Furthermore, the starting points will not be chosen near the boundaries of trolley instability. Coupled with the fact that the hook generally comes to rest in its equilibrium position beneath the trolley, a stable trolley is enough to ensure stability of the hook.

The second modification to Figure 46 is that the original "PD" controller is replaced by the "$K_{PD}(PD)$" block. This is necessary as the root locus traces the poles of a system by varying *one* gain parameter. Therefore, the transfer function of "$K_{PD}(PD)$" is $K_{PD}(P + Ds)$, where K_{PD} is the variable gain parameter. The P and D gains are fixed at preset values.

Thirdly, because the described root-locus plotting algorithm only works for linear systems, all non-linear elements must be either linearized, or removed completely. One such non-linear element that must be removed from Figure 46 is the "Saturator" block. Also, the "Converter" block from Figure 46 must be linearized. Thus, the piecewise equation that described the original converter, (39), was modified to the following:

$$\frac{V_R(s)}{e1(s)} = \frac{100}{e_{max}}, \tag{67}$$

where V_R is the reference velocity signal, $e1$ the error distance, and e_{max} is as before in (39).

The "Motor,Trolley" block is modified slightly from the original equation given in (37). The implicit conversion from the the PLC's digital representation of velocity (a value ranging from -100 to 100), to the actual velocity (in m/s) must now be made explicit in the transfer function, i.e.:

$$\frac{V_T(s)}{V_{ZV}(s)} = \frac{V_{max}}{100} \frac{\omega_{n,MT}^2}{s^2 + 2\zeta_{n,MT}\omega_{n,MT}s + \omega_{n,MT}^2} \tag{68}$$

where V_{max} corresponds to the maximum speed of the crane. Note that "Motor,Trolley" already ignores non-linearities such as motor saturation, rate limiting, switching element, and dead-zone. "Hook Dynamics" has the same transfer functions as before, given by (38). The ZV input shaper is given by (61).

With the linearized system in Figure 66, the root locus can be plotted. The open-loop transfer function of the closed-loop system we wish to analyze is given by:

$$G_{OL} = (K_{PD}(P + Ds)) \left(A_1 + A_2 e^{-st_2}\right) \left(\frac{1}{s}\frac{V_{max}}{e_{max}}\right) \left(\frac{\omega_{n,MT}^2}{s^2 + 2\zeta_{n,MT}\omega_{n,MT}s + \omega_{n,MT}^2}\right). \tag{69}$$

Figure 67 shows the root locus using $P = 1$ and $D = 1$, and varying K_{PD} from 0 to 5000. Only the positive imaginary half-plane is shown. The closed-loop poles and zeros are shown by crosses and circles, respectively. The imaginary axis is represented by the dashed line.

As the input shaper generates an infinite column of zeros parallel to the imaginary axis, the root locus contains an infinite number of branches (the branches are the trace of closed-loop poles) that end at those zeros. This presents a difficulty, as it is not obvious what axes range should be drawn to include all significant branches and zeros. However, it is explained in [7] that below a sufficiently large radial distance from the origin, the closed-loop poles arising from the input shaper move left (in the negative direction on the real axis) as they get further from the real axis. This means that the most significant closed-loop poles arising from an input shaper are those closest to the real axis. Typically in a real system, the most significant poles are those that

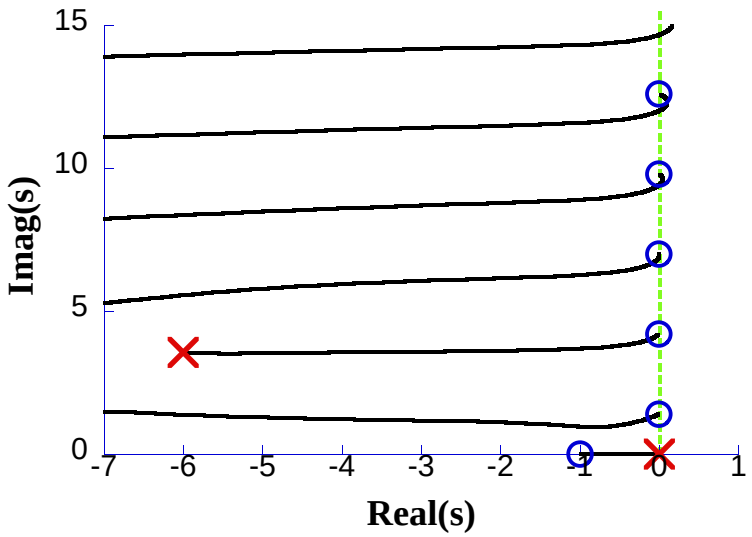


Fig 67. Root Locus for the Linearized Closed Loop PD with ZV Input Shaper Hand-Motion Controller

have lower frequencies (i.e. closer to the real axis); therefore with some foresight and judgement, the axes range of the root locus should be chosen to include only a finite number of these branches.

Figure 68 shows a close-up view on the branches near the imaginary axis. The branches curve from beneath the zeros out towards the unstable region, before curving back and terminating at the zeros. The system becomes marginally stable when $K_{PD} = 50$. This corresponds to the closed-loop pole that crosses the imaginary axis at around $imag(s) = 12$. At this gain, the closed-loop poles for the branches near $imag(s) = 9$ and $imag(s) = 7$ are also close to the imaginary axis, but do not cross into the unstable region.

Figure 69 shows the computer-simulated trolley response when the linearized system of Figure 66 is subjected to a step input, with $P = 1$, $D = 1$, and $K_{PD} = 50$. Clearly the trolley response shows growing instability. Figure 70 shows the close-up view of the trolley response for a portion of the simulation. Two distinct frequencies of oscillations are visible, one with a period of 0.5 seconds, which corresponds to the unstable poles that occur at $imag(s) = 12$ in the root locus, and the other with a

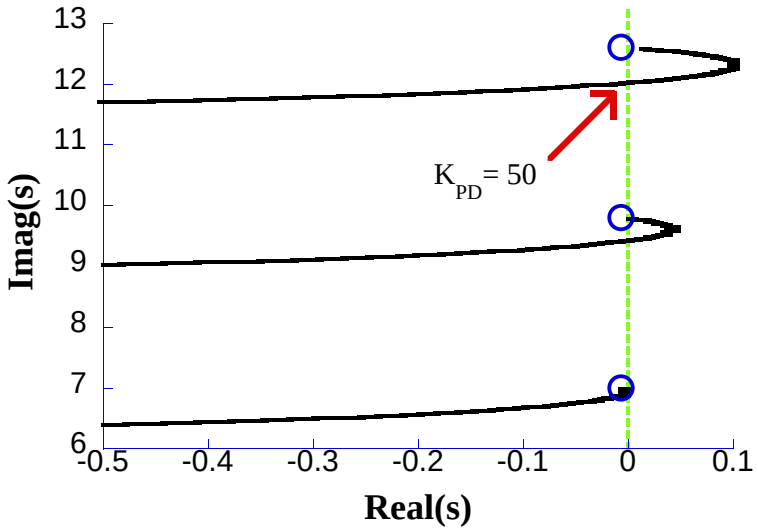


Fig 68. The Unstable Poles of the Linearized Closed Loop PD with ZV Input Shaper Hand-Motion Controller

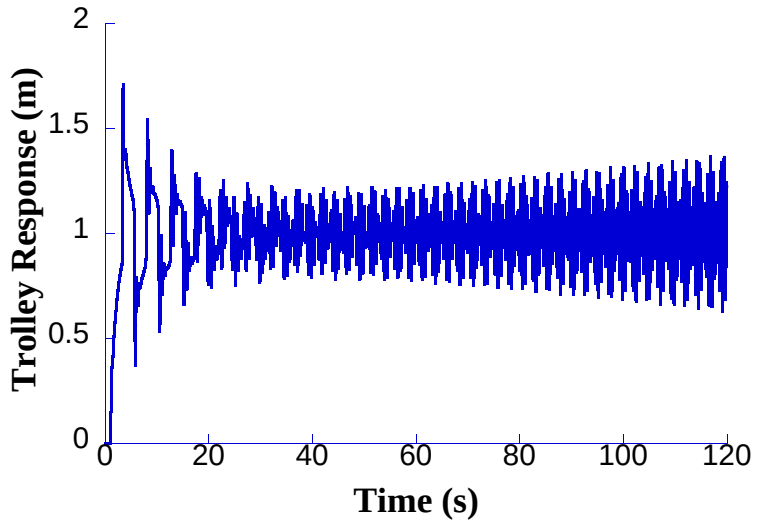


Fig 69. Linearized System Simulated Trolley Response for $K_{PD} = 50$

period of 2.0 seconds.

It is of interest to examine the simulated step response of the non-linear system from Figure 46 while using similar gains as the PD controller above. That is, we set $P = 50$ and $D = 50$. The simulated response is shown in Figure 71. The response is stable, however, as the close-up view of Figure 72 shows, there are minute oscillations (peak to peak amplitude of less than 3mm) around the set point with a period of

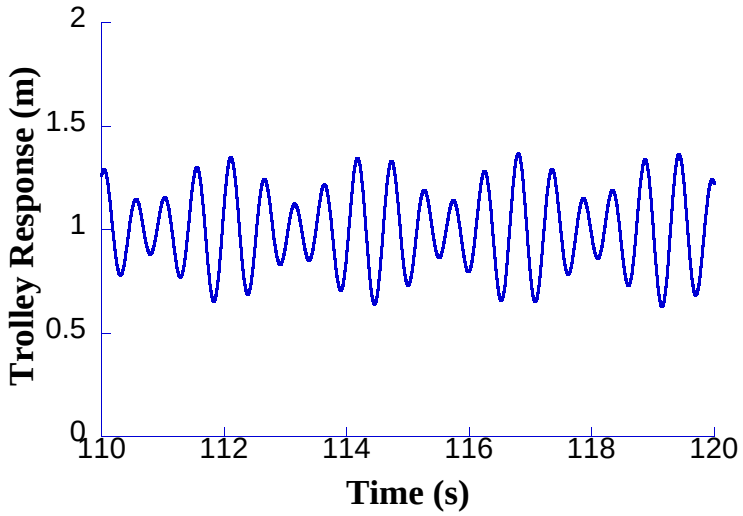


Fig 70. Linearized System Simulated Trolley Response for $K_{PD} = 50$, Close-Up View

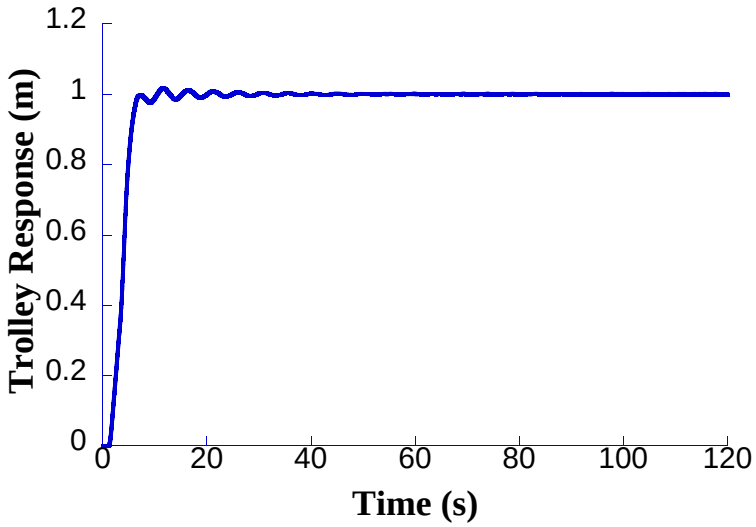


Fig 71. Non-Linear System Simulated Trolley Response for $P = 50$, $D = 50$

approximately 1 second.

Figures 71 and 72 demonstrate the limitations of the root locus. As root locus can only be plotted for a linearized system, the stabilizing properties of non-linear elements, such as the saturator, are ignored. The saturator has a tendency to stabilize the system because it prevents excessive actuator effort. Furthermore, the two degrees of freedom available when choosing the P and D gains separately is reduced to one

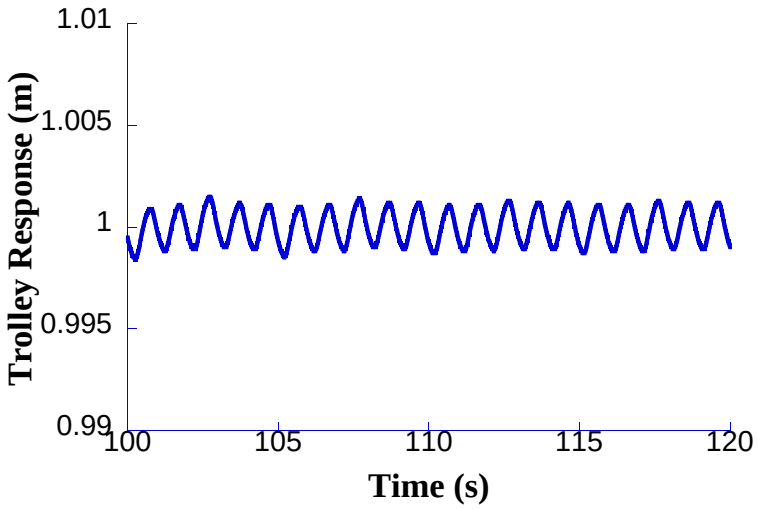


Fig 72. Non-Linear System Simulated Trolley Response for $P = 50$, $D = 50$, Close-Up View

when using the root locus, as the gains must be lumped into one variable, K_{PD}. Despite these limitations, stability analysis using the root locus is still valuable as it provides an approximate upper limit on the range of stable gains. It is also useful as it provides a starting point in the search for the PD gains that will yield desirable crane responses using more detailed methods.

3.5 Numerical Methods for the Selection of PD Gains Using Matlab and Simulink

As seen in section 3.4, the utility of the root locus analysis is to provide approximate values for the gains that result in unstable trolley behavior. In this section, we present two numerical, iterative methods implemented in Matlab and Simulink. These methods use the information from the root locus analysis to search for P and D gains that will yield the most desirable crane response.

A model was constructed in Simulink that implements the block diagram shown in Figure 46. This was the same model that was used to generate the simulation results discussed in section 3.3.2. To search for the optimized PD gains, a cost function must first be constructed to quantize the performance of the Simulink model. After

a simulation is completed, the cost function is used to analyze the simulation results and to return a scalar value. Numerical and iterative methods are then used to find the optimal set(s) of PD gains that minimize the cost.

3.5.1 Cost Function for Quantifying the Controller Performance

The crux of numerical methods lies in the design of the cost function. Its design is important, because it needs to capture all the characteristics of the crane response that are considered "desirable" for an intuitive crane controller. However, as mentioned in the beginning of this chapter, the desirable response is subjective to individual preferences. This presents a difficulty, because there is not *one*, but *many* sets of PD gains that produce "desirable" crane responses.

Presented here is an example of a simple cost function. We assume the input to the hand-motion controller, i.e. the position of the hand-held device, is a ramp with a gradient equivalent to the maximum velocity of the Georgia Tech 10-ton crane (0.3577m/s). The ramp begins at $t = t_1 = 1s$ and ends when the position of the hand-held device has traveled a distance of 2 meters, i.e. at $t = t_2 = 6.59s$. This is the same input trajectory that was used to generate the simulation results of section 3.3.2, and was seen (labeled as "Glove") in Figures 51, 52, and 53.

The total cost function is a piece-wise function consisting of two parts, the ramp-up cost and the settling cost. It compares the position of the hand-held device, $pos_{hand-held\,device}$ with the position of the hook, pos_{hook}:

$$Ramp\,Up\,Cost = \Sigma|pos_{hand-held\,device}(t) - pos_{hook}(t)| \quad : t_1 \leq t < t_2 \tag{70}$$

$$Settling\,Cost = \Sigma|pos_{hand-held\,device}(t) - pos_{hook}(t)| \quad : t_2 \leq t < t_{end} \tag{71}$$

$$Total\,Cost = Ramp\,Up\,Cost + Settling\,Cost, \tag{72}$$

where t_{end} is the simulation end time, which was set to 30 seconds. The ramp-up cost is the summation of the absolute difference between the position of the hand-held device and the hook, during the ramp-up stage of the input. This was designed to

penalize gains that produced a slow response to operator commands. The settling cost is the summation of the absolute difference between the position of the hand-held device and the hook, between the ramp end time and the simulation end time. This was designed to penalize gains that produced long settling times. Gains that have high settling costs typically produced responses that were oscillatory around the settling point.

3.5.2 Numerical Methods for the Selection of PD Gains that Minimized the Cost Function

Using the cost function in (72), we present two numerical methods that find the optimal PD gains.

The first method uses `fminsearch`, a built-in Matlab function. This is an unconstrained minimum search method that searches over the two dimensional space of P and D gains to find the combination that yields the lowest cost. It requires the user to enter initial guesses. The choice for the initial guesses was provided by the judicial use of the results from the root locus analysis, which was discussed in section 3.4.3.

As mentioned earlier, the root locus analysis provides an approximate upper limit on the range of stable PD gains. The marginally stable gains found in section 3.4.3 were $P = 50$ and $D = 50$. We set our allowable search range of gains to: $0 < P < 50$, and $0 < D < 50$. The algorithm for `fminsearch` stops after it reaches convergence to a solution within the specified tolerance. Table 1 summarizes the results for various start points. The processing time required to find the solution is listed for each case. The platform used was a 2GHz Intel Core 2 Duo PC running Windows Vista.

It is interesting to note that the initial guess has little effect on the optimal PD values, which are approximately 7.33 for P, and 8.32 for D. However, the processing time is longer if the initial guess is further from the final solution. Also, it appears that there is only one minimum over the range of PD gains investigated. This may be due to the simple design of our cost function. The benefit of using `fminsearch` is

Table 1. Optimal PD Gains Found Using the `fminsearch` Method

Initial Guess	Optimal P	Optimal D	Processing Time (s)
$P = 10$ $D = 10$	7.331	8.3189	86
$P = 20$ $D = 20$	7.331	8.3187	117
$P = 30$ $D = 30$	7.3308	8.3189	127
$P = 40$ $D = 40$	7.3305	8.3185	139
$P = 49$ $D = 49$	7.3309	8.3186	230

that it does not rely on numerical or analytic gradients and can handle discontinuities. However, the algorithm is not guaranteed to find the global optimum.

Suppose that cost function and the dynamics of the entire hand-motion control system were more complex; and this complexity produced multiple local minima over the space of PD gains and costs. In this case, an alternative to `fminsearch` is to use a method that relies on brute force. First, construct a grid over the allowable range of P and D gains. The optimal PD gains are then selected by examining the costs resulting from every P and D combination. The advantages of this method is that it is guaranteed to find the "global" (global within the range of allowable PD gains) minimum.

The brute force method was executed in the allowable range of $0 < P < 50$, and $0 < D < 50; P \in \mathbb{I}, D \in \mathbb{I}$. In total, approximately 2500 simulations were made, taking around 40 minutes to complete on a 2GHz Intel Core 2 Quad PC running Windows Vista. Figure 73 shows two views of the surface plot of the cost versus the P and D gains. The vertical axis represents the logarithmic of the total cost, and the horizontal axes represent the P and D gains ranging from 1 to 50. It is clear that there is one minimum over this range (darkest color denote the lowest cost), and lies in the region of P = 7, D = 8. The result from this brute force method confirms the single minimum found using the `fminsearch` method.

The `fminsearch` method has the advantage of speed, but is not guaranteed to find the global minimum over the allowable range of PD gains. However, this can be

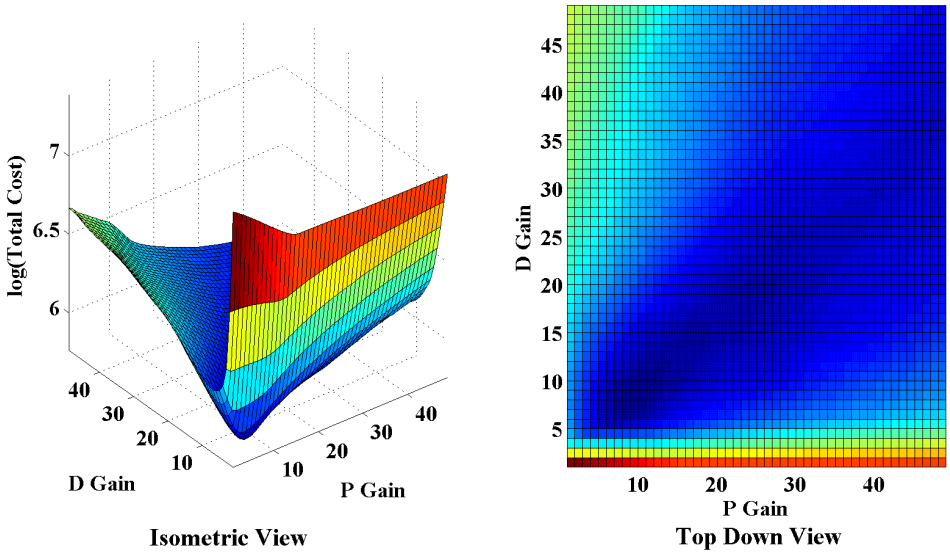


Fig 73. Cost vs. PD Gains Using the Brute Force Method

supplemented by the brute force method, which can be used to provide an overview of the cost over the entire range of PD gains. However, the brute force method is costly and inefficient. Also, the degree of accuracy of the brute force method is determined by the grid spacing between successive PD gains. If there are multiple minima over the allowable range, then the brute force method can be used to complement the `fminsearch` method by providing appropriate initial guesses. In this way, accurate solutions can be found for each local minimum.

Figure 74 shows the computer simulated hand-motion controller response using the optimal gains from Table 1: 7.33 for P, and 8.32 for D. The 10% to 90% rise time is approximately 4.87 seconds, the maximum percentage overshoot is about 2%, and the 2% settling time is around 10 seconds. Figure 75 shows the experimental response on the 10-ton bridge crane using the same optimal gains. The 10% to 90% rise is approximately 5 seconds and there is virtually no overshoot. The 2% settling time is around 10 seconds. The performance of the hand-motion controller using the optimized PD gains is similar to that of using the PD gains determined by empirical means, which were shown in Figures 53 and 56 of section 3.3.

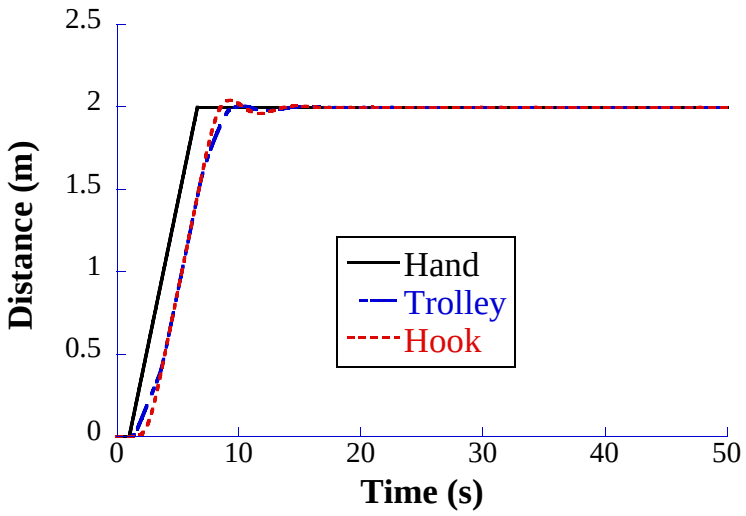


Fig 74. Simulated Results of the Hand-Motion Controller with Optimized PD Gains

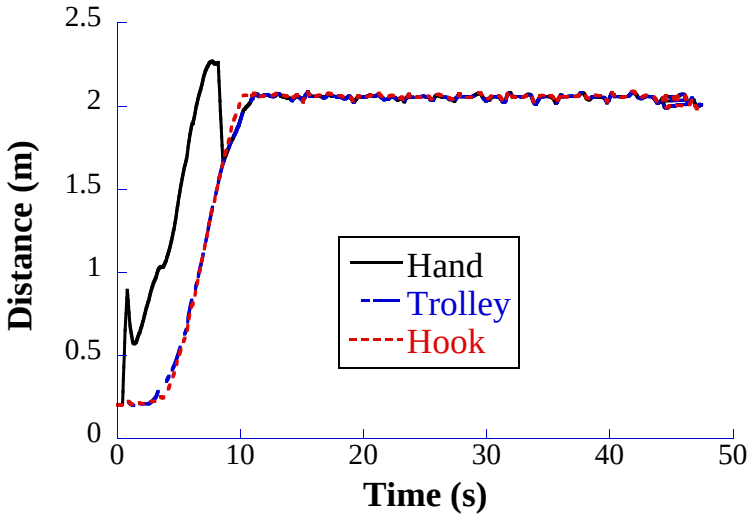


Fig 75. Experimental Results of the Hand-Motion Controller with Optimized PD Gains

3.5.3 Numerical Methods Using Alternative Cost Functions

To investigate the robustness of the numerical methods to different cost functions, we present an illustrative example using a cost function that penalizes/rewards different attributes of the crane response.

We set the hand-held device input trajectory to the same ramp-plateau as before.

The trajectory stops at a distance of $d_{stop} = 2m$. The ramp-up and settling costs are calculated in a similar way to (70) and (71):

$$Ramp\,Up\,Cost = \Sigma|pos_{hand-held\,device}(t) - pos_{hook}(t)| \quad : t_1 \leq t < t_{stop} \tag{73}$$

$$Settling\,Cost = \Sigma|pos_{hand-held\,device}(t) - pos_{hook}(t)| \quad : t_{stop} \leq t < t_{end} \tag{74}$$

where t_{end} is the simulation end time, which was set to 30 seconds. The variable t_1 is the time at which the ramp started (set to 1 sec), and t_{stop} is the time at which the hook position first reaches the stop distance, d_{stop}.

Additionally, we include a cost that penalizes excessive actuator effort or high energy requirement. As there is friction in the system, the total work done by the actuators can be approximated by the total trolley move distance. This allow us to form a simple energy cost function:

$$Energy\,Cost = \Sigma|pos_{trolley}(t^+) - pos_{trolley}(t)| \quad : 0 \leq t < t_{end} \tag{75}$$

We also define three user-adjustable weighting terms to tune the contribution each type of cost makes to the total cost function. The weighting for the ramp-up cost, α, is calculated by normalizing the ramp-up duration of the hook response against the total simulation time. The weighting for the settling cost, β, is calculated by similar means using the settling duration. The weighting term, γ, scales the energy cost according to the total simulation time. The three weighting variables and the total cost are given by:

$$\alpha = 1 \times \frac{(t_{stop} - t_1)}{t_{end}} \tag{76}$$

$$\beta = 1 \times \frac{(t_{end} - t_{stop})}{t_{end}} \tag{77}$$

$$\gamma = 1 \times t_{end} \tag{78}$$

$$Total\,Cost = \alpha(Ramp\,Up\,Cost) + \beta(Settling\,Cost) + \gamma(Energy\,Cost). \tag{79}$$

Figure 76 shows two views of the cost surface over the range of $0 < P < 50$, and $0 < D < 50; P \in \mathbb{I}, D \in \mathbb{I}$. This was generated using the brute force method.

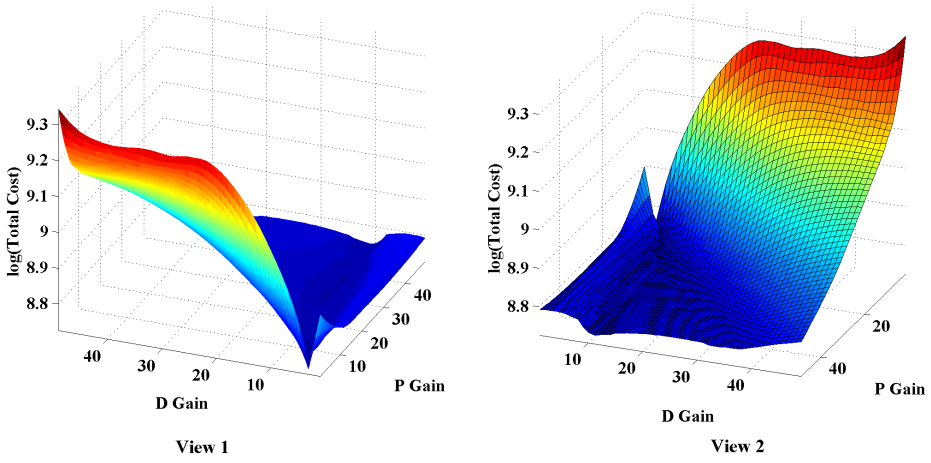


Fig 76. Two Views of the Cost Surface vs. PD Gains Using the Alternate Cost Function

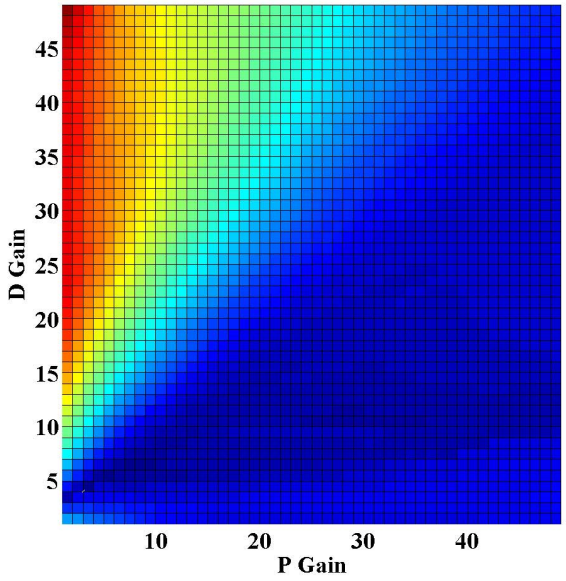


Fig 77. Top View of the Cost Surface vs. PD Gains Using the Alternate Cost Function

The shape of this cost surface is significantly different than the cost surface shown in Figure 73. Figure 77 shows the top view of the cost surface. The darkest regions denote the lowest cost. Over this range, the lowest cost is given by the set of gains $P = 3$ and $D = 4$.

Table 2 summarizes the results using the `fminsearch` method for various initial guesses. The first four columns have the same meaning as in Table 1. Highlighted in

Table 2. Optimal PD Gains Found Using the `fminsearch` Method for the Alternate Cost Function

Initial Guess	Optimal P	Optimal D	Process Time (s)	Actual Cost
P=3 D=3	4.2011	3.5462	139	6143
P=3 D=4	**4.6230**	**4.5067**	**92**	**6149**
P=4 D=3	4.0078	3.1478	89	6151
P=4 D=4	4.2166	3.5817	94	6143
P=10 D=10	4.2136	3.5746	185	6143
P=20 D=20	9.2404	27.6598	126	6259
P=30 D=30	9.2627	27.6267	162	6258
P=40 D=40	4.2143	3.5713	368	6143
P=49 D=49	31.9575	46.8810	143	6425

bold are the results when the gains from the brute force method ($P = 3$ and $D = 4$) , are used as the initial guess for the `fminsearch` method. It can be seen that due to the increased complexity of the cost function, multiple local minima exist. This is also reflected by the irregular pattern in the total process time. The results suggest there are local minima around $P \approx 4, D \approx 3.5$; $P = 9.2, D = 27.6$; and $P = 32, D = 46.9$. Despite this, initial guesses that were close to $P = 3$ and $D = 4$ yielded similar solutions.

The last column in Table 2 shows the actual cost from using the optimized PD gains. Gains of around $P \approx 4, D \approx 3.5$ yielded the lowest costs of around 6140, while the two local minimum near $P = 9.2, D = 27.6$ had a slightly higher cost of approximately 6260. Interestingly, the local minimum at $P = 32, D = 46.9$ also yielded a low cost of 6425.

It is important to note that the form of the cost functions we presented were chosen according to our commonly desired performance requirements. Should there be a need to penalize or reward other characteristics of the crane response, the control engineer simply needs to choose another cost function that better captures the important performance characteristics necessary for the intended application.

For example, high energy usage may also occur when the motors undergo high

acceleration or deceleration. Therefore, terms related to trolley acceleration can be used in the energy cost function. Cost functions that assume other input trajectories, such as a step or more complex 2-D motions, may also be utilized. For example, the ramp of the gradient can be modified to reflect a human operator's walking speed, or the stopping distance can be increased.

Alternatively, the engineer may choose to penalize only the spatial error (discussed in section 3.3.4) between the positions of the hand-held device and the hook. Spatial error disregards the lag in the hook response that is necessary for the controller, and is an indicator of the crane's ability to follow the shape of the desired trajectory.

3.6 Summary

This chapter presented hand-motion controllers for intuitive crane control. The chapter began with a discussion on the performance of the standard pendent controlled crane. Following this, the structure of and justification for the empirically designed controller was presented. The first controller investigated was a PD controller with low gains. Given the sluggish response, the controller was changed to use high gains. The high-gain controller resulted in large oscillations, so a PD with high-gains and ZV-input-shaper controller was developed. These controllers were evaluated by simulation and experimental results.

Noting that the empirically designed controller chose the PD gains based on engineering rules of thumb, a structured methodology for determining the optimal PD gains was then presented. The discussion began by using root locus to analyze systems with closed-loop input-shaping. The results of this analysis can be used to generate a stable space of possible gains for two numerical methods that search for the optimized PD gains. The methodology is summarized below:

1. Plot the root locus of the system. Linearize the system if necessary.

2. Obtain the set of marginally stable PD gains, PD_{RL}, from the root locus.

3. Design a suitable cost function, $f_{cost}()$, that penalizes undesired crane responses.

4. Using PD_{RL} as an approximate guideline, and the cost function $f_{cost}()$, determine the optimized PD gains $PD_{optimized}$ using one (or a combination) of the following two numerical methods:

   (a) `fminsearch` method - Searches for the set of PD gains that yield the local minimum cost. This method is fast and efficient, but can only find the local minimum.

   (b) Brute force method - Determines the cost for every combination of PD gains over a prescribed range. This method provides a general picture of the cost over the range of PD gains. It is inefficient, but can indicate the locations of local minima.

CHAPTER IV

CONCLUSIONS AND FUTURE WORK

Cranes are crucially important elements in the industrial complex. They are used in many areas such as shipping yards, construction sites, and warehouses, just to name a few. However, payload oscillation inherent to all cranes makes it challenging for human operators to manipulate payloads quickly, accurately, and safely. Manipulation difficulty is also increased by non-intuitive crane control interfaces. Intuitiveness is characterized by ease of learning, simplicity, and predictability. This thesis addressed the issue of intuitive crane control in two parts: the design of the interface, and the design of the controller. Simulation and experimental results were presented.

Three novel types of crane control interfaces were presented in chapter 2 - wand, glove, and RF-based control. These interfaces allow an operator to drive a crane by moving his or her hand freely in space. The wand and glove control interfaces were heavily dependent on machine vision. This thesis contributes a number of machine vision algorithms in order to facilitate those interfaces. These algorithms include: i) the use of two camera acquisition windows for the independent real-time tracking of the hook and the wand/glove; ii) the K-means algorithm used to distinguish between hook and wand/glove blobs; and iii) using software predictors to aid hook tracking.

The implementation of the RF-based control using an off-the-shelf product from Ubisense was also described. The potential for RF-based control exceeds machine-vision-based methods because of the added robustness and reliability in using RF sensors and RF tags. Tracking is still possible when the direct sensor-to-tag line of sight is broken. Furthermore, extra functionalities can be added using the two-way communication channel between the sensors and the RF tag.

The design of the hand-motion controller was discussed in chapter 3. The process of empirically designing the controller was presented, and its performance was documented with simulation and experimental results. The design process started with a PD controller with low gains, followed by a PD controller with high gains, and ended with a PD with ZV-input-shaper controller. The combination of aggressive PD gains and ZV input shaping produced the desired characteristics of fast response, short settling time, small amplitude overshoot, and low residual oscillations. We also presented the results from an operator study that indicated crane control using an intuitive interface was more effective than using a standard pendent interface.

A more structured methodology for determining the optimal PD gains was also discussed. The discussion began with an analysis of systems with closed-loop input shaping using root locus. This analysis has some caveats, among which is the inability to account for non-linearities (such as saturators). For this reason, the results from the root locus analysis should be used as guidelines for other methods that search for optimized PD gains. Two such methods were discussed, both of which are numerically based.

Crucial to the numerically-based methods was the design of the cost function. The cost function returns a scalar value by analyzing the simulated response of a computer model of the crane. Undesirable characteristics such as lag and large-amplitude oscillations are assigned a higher cost. The goal of the numerical methods is to find the set of PD gains that minimized the cost. The results of the root-locus analysis are used as guidelines to define the suitable search space for these numerical methods.

The first numerical method is derived from the Matlab function `fminsearch`, which is fast, accurate, and capable of finding the local minimum. The second method relies on brute force and calculates the cost for a large combination of P and D gains. The two methods can be used in conjunction and each complements the deficiencies

of the other. For example, the brute force method can be used gain an overview of the search space and provide the approximate locations of multiple local minima. The `fminsearch` can then be used to find accurate locations of those minima.

4.1 Future Work

The RF-based control interface has many interesting features and possible extensions that have not been explored and optimized. As was discussed in section 3.3.4, there is noticeable amount of noise in the measured tag locations. Although this does not have a significant impact on the crane performance for the controllers presented in this thesis, it is still desirable to have the best measurements available. Ongoing work is addressing this issue by designing suitable filters.

The removal of RF-sensor noise will be important for additional functionalities that exploit the RF-based technology. One such area is to expand the movement capabilities of intuitive crane control by enabling movement in the vertical (hoisting/lowering) direction, which the operator signals by moving the hand-held device vertically up or down. There is an implemented version of the control system that is capable of vertical movement, but is based on machine vision. It is dependent on the fact that reflective markers on the wand or glove appear larger or smaller to the camera as it moves up or down. This essentially requires the camera to detect movement in the direction parallel to its focal axis, which is not ideal because a single camera setup does not have true depth perception. The RF-based interface circumvents this problem because the tag can be located in 3-D space.

The current controllers use a PD controller to drive the error between the hook and the hand-held device to zero. The effect of this is that the hand-held device acts as a crane-puller; i.e. the hook follows the operator. Further research could explore the inverse of the puller, i.e. a crane-pusher, where the hand-held device exerts a "repulsive force" on the crane. Additionally, a hybrid between the puller and pusher

can be investigated. The crane will act as a puller when the hand-held device is beyond a preset distance, $D1$, from the crane, and act as a pusher when the device is within another preset distance, $D2$, from the crane. Effectively, the crane will move to maintain its position in the dead-zone between $D1$ and $D2$, around the operator. This feature will be useful for "hands-free" operation, where the operator can safely have both hands manipulating the payload, while the crane automatically maneuver itself in the dead space around the operator.

The simulation and experimental results of this thesis were all based on single pendulum dynamics. That is, cranes with a suspended hook and no payload attached, or a payload that behaves sufficiently like a single pendulum when attached to the hook. Further research would analyze the likely double-pendulum dynamics that can result from a crane with payload attachments. The research would also focus on designing appropriate counter-measures (such as multi-mode or robust input shapers) to address residual oscillations resulting from the new dynamics.

A final area of research pertains to the design of the cost function, which was used for the numerical methods that finds optimized PD gains. As discussed in section 3.5.1, the cost function is crucial as it essentially defines the "desirable" characteristics of the crane response. This thesis presented two simple cost functions with the aim of demonstrating the use of the numerical methods. However, further research would develop cost functions that capture "desirable" crane responses for a wide range of applications. One example that was mentioned in section 3.5.1 is to penalize only the spatial error, rather than the inevitable temporal lag in the crane response. The spatial error and temporal lag can also be combined to produce a hybrid cost function.

REFERENCES

[1] Abdel-Rahman, E., Nayfeh, A., and Masoud, Z., "Dynamics and Control of Cranes: A Review," *Journal of Vibration and Control*, vol. 9, no. 7, p. 863, 2003.

[2] Colgate, J., Peshkin, M., and Klostermeyer, S., "Intelligent assist devices in industrial applications: a review," *Intelligent Robots and Systems, 2003. (IROS 2003). Proceedings. 2003 IEEE/RSJ International Conference on*, vol. 3, pp. 2516–2521 vol.3, Oct. 2003.

[3] Dobson, B. and Rider, E., "Review of the indirect calculation of excitation forces from measured structural response data," *Proceedings of the Institution of Mechanical Engineers. Pt. C. Journal of Mechanical Engineering Science*, vol. 204, pp. 69–75, 1990.

[4] Fisch, J., "Machine vision for an advanced crane control system," tech. rep., Georgia Institute of Technology, 2007.

[5] Han, J. Y., "Low-cost multi-touch sensing through frustrated total internal reflection," in *UIST '05: Proceedings of the 18th annual ACM symposium on User interface software and technology*, (New York, NY, USA), pp. 115–118, ACM, 2005.

[6] Hong, K.-S., Kim, J. H., and Lee, K.-I., "Control of a container crane: Fast traversing, and residual sway control from the perspective of controlling an underactuated system," in *Proceedings of the 1998 American Control Conference*, vol. 2, pp. 1294–1298, 1998.

[7] Hong, K.-T. and Hong, K.-S., "Input shaping and vsc of container cranes," in *IEEE International Conference on Control Applications*, (Taipei, Taiwan), pp. 1570–1575, 2004.

[8] Huey, J., *The Intelligent Combination of Input Shaping and PID Feedback Control*. PhD thesis, Georgia Institute of Technology, 2006.

[9] Huey, J. R., Sorensen, K. L., and Singhose, W. E., "Useful applications of closed-loop signal shaping controllers," *Control Engineering Practice*, vol. 16, no. 7, pp. 836 – 846, 2008.

[10] Kazerooni, H., "The human power amplifier technology at the university of california, berkeley," *Robotics and Autonomous Systems*, vol. 19, no. 2, pp. 179 – 187, 1996. ICRAM'95.

[11] Kazerooni, H., Fairbanks, D., Chen, A., and Shin, G., "The magic glove," *Robotics and Automation, 2004. Proceedings. ICRA '04. 2004 IEEE International Conference on*, vol. 1, pp. 757–763 Vol.1, April-1 May 2004.

[12] Kazerooni, H. and Guo, J., "Human extenders," *Journal of Dynamic Systems, Measurement, and Control*, vol. 115, no. 2B, pp. 281–290, 1993.

[13] Khalid, A., Huey, J., Singhose, W., and Lawrence, J., "Human operator performance testing using an input-shaped bridge crane," *ASME J. of Dynamic Systems, Measurement, and Control*, vol. 128, no. 4, pp. 835–841, 2006.

[14] Kim, D. and Singhose, W., "Human operator learning on double-pendulum bridge cranes," in *ASME IMECE*, (Seattle, WA), 2007.

[15] Kim, D. and Singhose, W., "Studies of human operators manipulating double-pendulum bridge cranes," in *European Control Conference*, (Kos, Greece), 2007.

[16] Kim, Y.-S., Seo, H.-S., and Sul, S.-K., "A new anti-sway control scheme for trolley crane system," in *Thiry-Sixth IAS Annual Meeting. Conference Record of Industry Applications Conference*, vol. 1, pp. 548–552, 2001.

[17] Kozak, K., Huey, J., and Singhose, W., "Performance measures for input shaping," in *IEEE Conf. on Control Applications*, (Istanbul, Turkey), 2003.

[18] Lee, J., Hudson, S., and Dietz, P., "Hybrid infrared and visible light projection for location tracking," in *UIST '07: Proceedings of the 20th annual ACM symposium on User interface software and technology*, (New York, NY, USA), pp. 57–60, ACM, 2007.

[19] Lee, J. C., Dietz, P. H., Maynes-Aminzade, D., Raskar, R., and Hudson, S. E., "Automatic projector calibration with embedded light sensors," in *UIST '04: Proceedings of the 17th annual ACM symposium on User interface software and technology*, (New York, NY, USA), pp. 123–126, ACM, 2004.

[20] Lee, J. C., Hudson, S. E., Summet, J. W., and Dietz, P. H., "Moveable interactive projected displays using projector based tracking," in *UIST '05: Proceedings of the 18th annual ACM symposium on User interface software and technology*, (New York, NY, USA), pp. 63–72, ACM, 2005.

[21] Lee, J. C., Hudson, S. E., and Tse, E., "Foldable interactive displays," in *UIST '08: Proceedings of the 21st annual ACM symposium on User interface software and technology*, (New York, NY, USA), pp. 287–290, ACM, 2008.

[22] Lewis, D., Parker, G. G., Driessen, B., and Robinett, R. D., "Command shaping control of an operator-in-the-loop boom crane," in *American Control Conference*, (Philadelphia, PA), pp. 2643–7, 1998.

[23] Lloyd, S., "Least squares quantization in pcm," *Information Theory, IEEE Transactions on*, vol. 28, no. 2, pp. 129–137, 1982.

[24] Lynch, K., Liu, C., Sorensen, A., Kim, S., Peshkin, M., Colgate, J., Tickel, T., Hannon, D., and Shiles, K., "Motion Guides for Assisted Manipulation," *The International Journal of Robotics Research*, vol. 21, no. 1, p. 27, 2002.

[25] Masoud, Z. and Daqaq, M., "A graphical approach to input-shaping control design for container cranes with hoist," *IEEE Transactions on Control Systems Technology*, vol. 14, no. 6, pp. 1070–1077, 2006.

[26] Merrill, D., Kalanithi, J., and Maes, P., "Siftables: towards sensor network user interfaces," in *TEI '07: Proceedings of the 1st international conference on Tangible and embedded interaction*, (New York, NY, USA), pp. 75–78, ACM, 2007.

[27] Mistry, P., Maes, P., and Chang, L., "Wuw - wear ur world: a wearable gestural interface," in *CHI EA '09: Proceedings of the 27th international conference extended abstracts on Human factors in computing systems*, (New York, NY, USA), pp. 4111–4116, ACM, 2009.

[28] Neupert, J., Mahl, T., Sawodny, O., and Schneider, K., "A Nonlinear Control Strategy for Boom Cranes in Radial Direction," in *American Control Conference, 2007. ACC'07*, pp. 25–30, 2007.

[29] Nishioka, K., Adachi, N., and Takeuchi, K., "Simple pivoting algorithm for root-locus method of linear systems with delay," *International Journal of Control*, vol. 53, no. 4, pp. 951–966, 1991.

[30] Parker, G., Groom, K., Hurtado, J., Feddema, J., Robinett, R., and Leban, F., "Experimental verification of a command shaping boom crane control system," in *American Control Conference*, (San Diego, CA), pp. 86–90, 1999.

[31] Peshkin, M., Colgate, J., Wannasuphoprasit, W., Moore, C., Gillespie, R., and Akella, P., "Cobot architecture," *Robotics and Automation, IEEE Transactions on*, vol. 17, pp. 377–390, Aug 2001.

[32] Raskar, R., Beardsley, P., van Baar, J., Wang, Y., Dietz, P., Lee, J., Leigh, D., and Willwacher, T., "Rfig lamps: interacting with a self-describing world via photosensing wireless tags and projectors," *ACM Trans. Graph.*, vol. 23, no. 3, pp. 406–415, 2004.

[33] Sakawa, Y. and Shindo, Y., "Optimal control of container cranes," *Automatica*, vol. 18, no. 3A, pp. 257–266, 1982.

[34] Sawodny, O., Hildebrandt, A., and Schneider, K., "Control design for the rotation of crane loads for boom cranes," in *IEEE International Conference on Robotics and Automation, 2003. Proceedings. ICRA'03*, vol. 2, 2003.

[35] Singer, N., Singhose, W., and Kriikku, E., "An input shaping controller enabling cranes to move without sway," in *ANS 7th Topical Meeting on Robotics and Remote Systems*, vol. 1, (Augusta, GA), pp. 225–31, 1997.

[36] Singer, N. C. and Seering, W. P., "Preshaping command inputs to reduce system vibration," *J. of Dynamic Sys., Measurement, and Control*, vol. 112, pp. 76–82, 1990.

[37] Singhose, W., Seering, W., and Singer, N., "Residual vibration reduction using vector diagrams to generate shaped inputs," *ASME J. of Mechanical Design*, vol. 116, no. June, pp. 654–659, 1994.

[38] Smith, O. J. M., "Posicast control of damped oscillatory systems," *Proceedings of the IRE*, vol. 45, no. September, pp. 1249–1255, 1957.

[39] Sorensen, K., Singhose, W., and Dickerson, S., "A controller enabling precise positioning and sway reduction in bridge and gantry cranes," *Control Engineering Practice*, vol. 15, no. 7, pp. 825–837, 2007.

[40] Sorensen, K. L., Danielson, J., and Singhose, W. E., "Anti-sway and positioning control for an industrial bridge crane with multi-mode dynamics," in *ASME International Symposium on Flexible Automation*, 2008.

[41] Sorensen, K. L., Spiers, J. B., and Singhose, W. E., "Operational effects of crane interface devices," in *IEEE Conference on Industrial Electronics and Applications*, (Harbin, China), 2007.

[42] Sorensen, K. L., "A combined feedback and command shaping controller for improving positioning and reducing cable sway in cranes," Master's thesis, Georgia Institute of Technology, 2005.

[43] Starr, G. P., "Swing-free transport of suspended objects with a path-controlled robot manipulator," *J. of Dynamic Systems, Measurement and Control*, vol. 107, pp. 97–100, 1985.

[44] Strip, D., "Swing-free transport of suspended objects: a general treatment," *IEEE Trans. Robotics and Automation*, vol. 5, no. 2, pp. 234 – 6, 1989.

[45] Yoon, J., Singhose, W., Vaughan, J., Ramirez, G., Kim, M., and Tawde, S., "Dynamics and control of crane payloads that bounce and pitch during hoisting," in *IDETC/CIE 2009*, (San Diego, CA), 2009.