Integrated Sensing, Dynamics and Control of a Mobile Gantry Crane

Kuan-chun Huang

Ph.D. Thesis

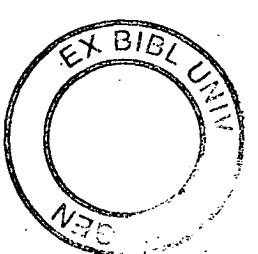
The University of Edinburgh

1997

Copyright

1997

Kuan-chun Huang



Dedicated to my father Yang-ching, and
to the memory of my grandfather Tsuang-shay

Abstract

This thesis investigates the dynamics and control of a Rubber Tyred Gantry (RTG) crane which is commonly used in container handling operations. Both theoretical and experimental work has been undertaken to ensure the balance of this research.

The concept of a Global Sensing System (GSS) is outlined, this being a closed loop automatic sensing system capable of guiding the lifting gear (spreader) to the location of the target container by using feedback signals from the crane's degrees of freedom. To acquire the crucial data for the coordinates and orientation of the swinging spreader a novel visual sensing system (VSS) is proposed. In addition algorithms used in the VSS for seeking the central coordinates of the clustered pixels from the digitised images are also developed.

In order to investigate the feasibility of different control strategies in practice, a scaled-down, 1/8 full size, experimental crane rig has been constructed with a new level of functionality in that the spreader in this rig is equipped with multiple cables to emulate the characteristics of a full-size RTG crane. A Crane Application Programming Interface (CAPI) is proposed to reduce the complexity and difficulty in integrating the control software and hardware. It provides a relatively user-friendly environment in which the end-user can focus on implementing the more fundamental issues of control strategies, rather than spending significant amounts of time in low-level device-dependent programming.

A control strategy using Feedback Linearization Control (FLC) is investigated. This can handle significant non-linearity in the dynamics of the RTG crane. Simulation results are provided, and so by means of the CAPI this controller is available for direct control of the experimental crane rig. The final part of the thesis is an integration of the analyses of the different subjects, and shows the feasibility of real-time implementation.

Declaration

This thesis is submitted in fulfillment of the requirements for the degree of Doctor of Philosophy at the University of Edinburgh. Unless otherwise stated, the work described is original and has not been previously submitted in whole, or in part, for any degree at this, or at any other, University.

Kuan-chun Huang

University of Edinburgh

November 1997

Acknowledgment

First and foremost I would like to thank my supervisor, Dr. M.P. Cartmell, without whom this work would not have been possible. He has given me a great amount of help and encouragement in each aspect of the work. Dr. Cartmell has been a constant source of support, strength and understanding, and has always provided excellent suggestion and guidance.

My thanks also to Dr. Jenq-shyang Pan and Chanin Bunlaksananusorn, who never refused to provide significant technical knowledge and advice, Mr. Alan Thomson, Mr. Laird Parker, Mr. Harry Mckeating and other technicians in the workshop who contributed a tremendous amount of effort to help me complete the experimental crane rig.

I would finally like extend my thanks to my father Yang-ching, my mother Whay-wha, my grandmother Yu-shing, my sisters Joy, Jung-ian and brother-in-law Michael. Without their consistent financial and mental support through these years I would not have been able to have gone this far. Finally, I would like to dedicate this thesis to the memory of my grandfather Tsuang-shay who passed away during my studying in Britain.

Contents

Chapter 1

Introduction

1.1 Introduction

Overhead cranes are commonly used to transport objects in many factories and work spaces. The fundamental motions are longitudinal travel, transverse trolley motion, object hoisting and object lowering. Cranes of this type normally consist of a gantry beam running on rails attached to factory walls, a motorised trolley which is free to move across the beam and a hook or electromagnetic clamp used as lifting gear. This lift gear is connected to the crane by means of cables, or ropes, and can be raised or lowered from the hoist drum attached to the trolley. Usually such cranes provide a facility for lifting/lowering of the payload and crossing the lateral span of the gantry at the same time. Many factory cranes also allow a longitudinal movement of the gantry beam by virtue of a pair of parallel rails fitted onto the walls to guide the gantry translation.

With functions similar to the factory overhead crane, *Rubber Tyred Gantry*, or RTG cranes are widely used to handle heavy ISO (International Standard Organization) freight containers in harbours and freight depots all over the world. This type of mobile crane allows the trolley to move along the top beam of a rail guided, or wheeled, Π-shape, portal gantry frame (Figures 1.1 and 1.2). The payload (container) can be hoisted or lowered, moved laterally and longitudinally, and in the case of the tyre mounted gantry a slow rotation of the whole crane is even possible.

Of particular theoretical and practical interest to researchers and operators alike is the capability and potential for *automation* of the mobile RTG Crane. A study of automation of this type of crane is presented in this thesis.

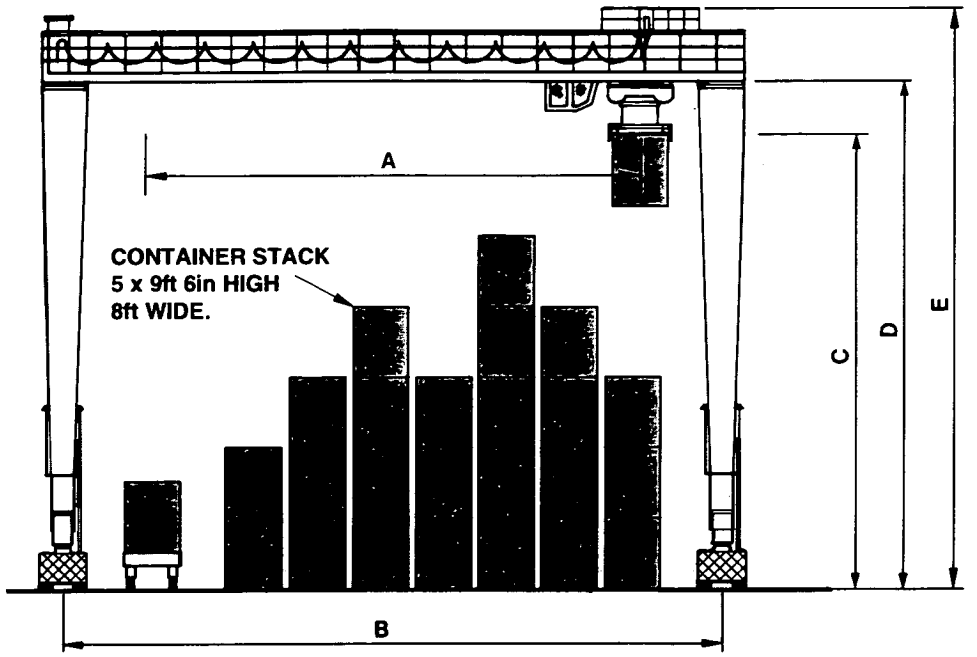
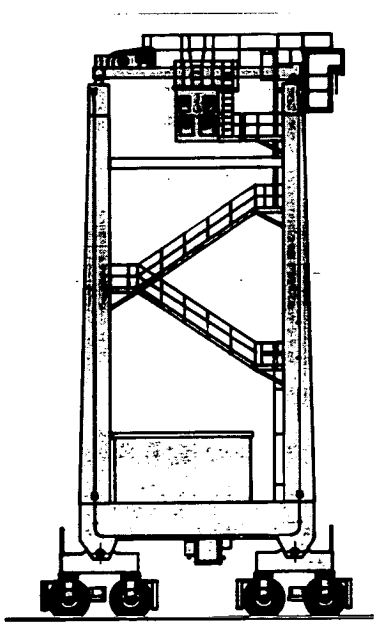


Figure 1.1 The front view of a full-size RTG crane (ref FEL International Ltd. [30])



DIMENSIONS (mm) - typical

	7+1 Wide	6+1 Wide	
A	19354	16612	Trolley Travel
B	23977	21234	Inside Clear Width

	1 over 6	1 over 5	1 over 4	
C	20600	17704	14808	Under Spreader
D	22429	19533	16637	Underside Top Beam
E	25553	22657	19761	Overall Height

Figure 1.2 The side view (ref FEL International Ltd. [30])

1.1.1 Rubber Tyred Gantry (RTG) Cranes

As shown in Figure 1.1 and Figure 1.2 the structure of the crane consists of the following parts:

1. The portal *gantry* frame

The portal frame is really two portal frames tied together one in front of the other, each comprising a top beam and two side beams. There are engine bays fitted onto the bottom end of the side beam pair with the wheels fitted to transmission equipment underneath the engine bays. These wheels are arranged so that they can be separately motored, thus providing a capability for drive in fore-aft, lateral, and even steered modes.

Thus the two pairs of side beams and top beams form a rigid, doubled, Π- shape structure capable of carrying payload containers of up to 40 tonnes.

2. The overhead *trolley*

The trolley is a fabricated structure which runs on wheels guided by rails on the top beams of the gantry. The trolley contains hoist hydraulics, motors, and a winching hoist drum as shown in Figure 1.3. An operator cab is frequently attached to the trolley (slung underneath and offset to the side), to provide the driver with clear views during manoeuvres.

3. The *spreader* (or lifting frame)

The spreader is a rectangular fabricated structure, also shown in Figure 1.3, often with a telescopic subassembly fitted, to provide for containers of different sizes.

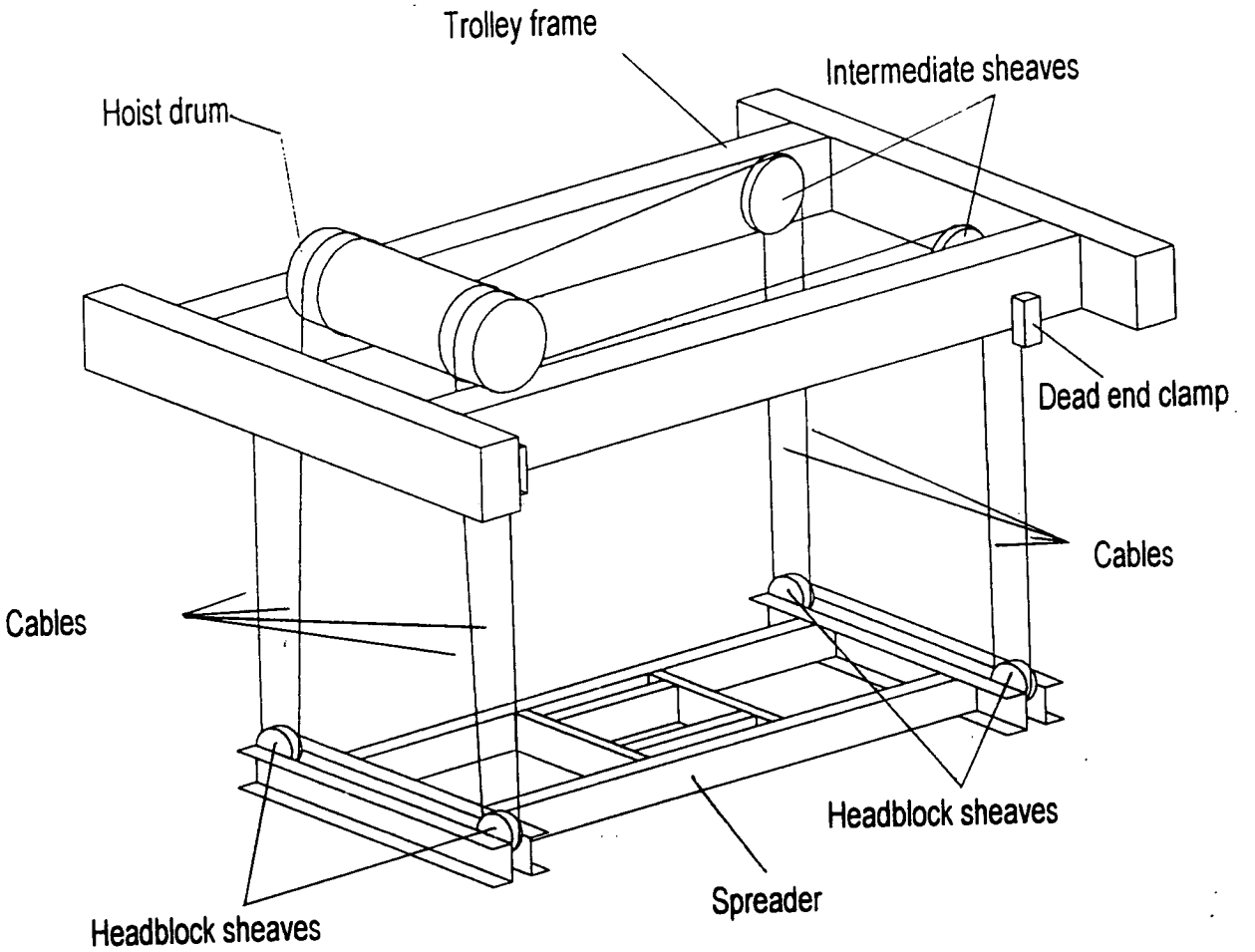


Figure 1.3 The trolley and spreader (after Cartmell et.al. [14])

4. Hoist ropes (or *cables*)

The cables are used to attach the spreader to the trolley. Figure 1.3 shows a typical cable suspension arrangement. Four pairs of parallel, closely running cables connect four corresponding corners of the trolley and the spreader, running over the set of headblock and intermediate sheaves and the hoist drum to allow the spreader to be lowered or hoisted. Thus, due to this configuration, the spreader is potentially free to swing below the trolley and also to rotate about its own local central axis.

1.2 Problem Overview

There has been a great deal of effort expended since containerisation started in the late 1960s to improve the productivity of cargo handling. Most of the large container terminals now use state of the art technology, such as two-way RDT (radio data transmission) systems, which are employed for the transfer of data in real time to/from automated management systems. Full use is now also being made of APD (automatic positioning determination) systems for both park equipment and container location. This trend towards automation will also ultimately extend to automation of the machinery. At Rotterdam container terminal, which is Europe's largest, AGV (automatic guided vehicle) trailer trains are used to transfer containers between the ship-to-shore crane (so-called quay crane) and the park cranes (such as RTG cranes), eliminating the need for labour intensive driver-operated park vehicles (Verschoof [89]). In Britain, the port of Thamsport is also configured to handle container transportation between storage areas and the shipside by means of an IMV (Internal Movement Vehicle) (Macleod [48]). Some companies plan even further for a fully automatic container management system, which means no human operators would be needed during the operation from the ship to the intended stacking site and vice versa. For example, the GRAIL system by Bohlman [8] proposed an overhead rail system which covers the entire working area. Therefore the complexity of the inter-task routines (and the crane-waiting time) can be reduced. These industrial developments have shown that a semi- or fully-automatic crane control system capable of dealing with different situations is needed more than ever.

Many terminal operators, especially those with specific container stack configurations, would suggest that emphasis should be placed on improving *park* (or stacking) productivity. This is because most of the bottlenecks in terminal operations occur before the ship-to-shore cranes (Bryfors [9]). As described in the last section, the cargo (or container) is transferred by means of the spreader (or lifting frame), which is suspended by non-rigid cables on a pulley mounted at each of its four corners. Because of the nature of the load suspension there is a tendency for load *sway* to occur during transport of the container. It is this sway which has been found to most adversely affect a crane's cycle time, and thus ultimately productivity and service. An expert

operator, through his sensory perception and fast reaction time, can do an excellent job at minimising the sway motion. Unfortunately, the kind of manoeuvres undertaken by a human operator are not repeatable time after time. A bad operation might be in the form of an exaggerated response, which could lead to an uncontrollable load sway and to long idle times before picking and placement of the payload. When the direction of motion is altered, residual vibration from earlier motion can compound the problem. Given concerns for increased productivity and safety, and ultimately with an eye on complete automation in the future, an automatic crane system capable of controlling load sway has now become a topic of considerable interest to the industry.

1.3 Review of existing research on the automation of gantry cranes

A number of previous researchers have tried to tackle this subject and different methods of control have been investigated in theory in the expectation of high speed transfers and high accuracy positioning of the payload. For example, *time optimal controls* have been developed by Auernig [4], Hamalainen [35], Mason [54], and Sakawa [67]. These papers have investigated an open-loop control method based on time-optimal solution. Starr [72] also shows a simple technique for guaranteeing that the load swing will be canceled at a desired set point. However, all the above papers employ open-loop control and cannot correct for unknown disturbances e.g. wind, or uncertain initial conditions. Controllers based on feedback will be more robust in the case of such disturbances. Moustafa [58], and Bulter [10] have looked at feedback controllers for overhead cranes but these were based on linearisations of the system around suitable operating points, and did not take into account the various inertial nonlinearities, particularly those imposed by the variation of the length of the cable that the load is suspended from.

Nonlinear control methods such as fuzzy control (Benhidjeb [5], Cartmell [16], Kijima [45], Suzuki [74], Yamada [94], Zadeh [97]) are also applicable in this application, realising the skill of human operators to produce a set of fuzzy rules for the control strategy. In the paper presented by Woods [93] a controller is proposed, based on dynamic feedback, and an extended state-coordinate change, which transforms the

system into a linear, controllable, one. Nevertheless the swing angle in this paper is assumed to be very small and only 2-D planar motion is simulated. *Artificial neural networks* (Javed [41]) is another modern control method introduced to this area, invoking multi-layered perceptron networks (Fukuda [33], Wang [90]) capable of producing the system parameters by learning (or being trained) from a preset knowledge base. However the lack of definitive theories for choosing the number of layers and perceptrons implies a difficulty in applying this technique. Some other, more unconventional control methods such as energy dissipation control (Chung [20], Hashimoto [37]) have also been investigated. The control object of the system is to manipulate the trolley position, suppressing the load swing only by the trolley driving force. Although the control law derived from the energy dissipation analysis of the system has the advantages of ease of implementation and high robustness against parametric errors, the swing angle of the load has not been utilized for control because of a generally encountered physical difficulty in measuring it. In the papers presented by Joshi [42] the cable of the payload is considered flexible, and a control law based on the root-locus approach is developed.

Therefore, the shortcoming of most of the above approaches are that they are:

1. based on linearized crane models, leaving unanswered questions of the effect of the resulting controller on the nonlinear crane dynamics.
2. only dealing with two degrees of freedom (the actuating degree for trolley movement and the swing angle of the payload), whereas in a practical working environment, the other two actuated degrees of freedom (transverse movement of the gantry and hoist/lowering action of the spreader) have to be taken into account to get a more thorough three dimensional representation of the problem.
3. merely focusing on numerical simulation rather than experimental tests.

From the arguments above it was reasonable to start considering the design of a dedicated experimental rig with the characteristics of a full-sized RTG crane, so that appropriate control theories can be tested, observed, and refined. Moreover, in order to comply with and explore the dynamics and control of a RTG crane more precisely, this rig should be designed to be capable of handling the movement of the trolley, the

gantry, the hoisting/lowering of the spreader, and also a rotation of the trolley. In addition to these described abilities the spreader is equipped with multiple cables to emulate the functionality of a full-size RTG crane. This rig now forms the basis of an excellent test-bed for future research.

Nevertheless, one of the most difficult, yet essential, aspects of any relevant experimental crane system has to be the practical arrangement configured for measuring the swing angle of the cable. A method provided in Marttinen [53] to measure the rope angle is to use the components of the rope tension in the pivot of the rope wheel. The momentary rope angle can be calculated from these force components measured from the force sensors on the ropes by trigonometry. However, hoisting causes disturbances to the measurement in this case and the relative position of the load cannot be calculated reliably from the rope angle measurement. The rope angle is therefore a function of the relative load position and acceleration of the trolley. However elastic waves in the rope become apparent in high frequency operations, if the load is not heavy enough and the rope is long. Therefore, the main problem lies in locating the position of the payload without interfering with it. There are a couple of non-contacting sensor systems which may be capable of resolving this problem. An acoustic sensor system was used by Hamalainen [35]. However, it is not feasible in this case here because not only does the position of the trolley need to be determined but the orientation of the spreader is also an essential part of the subject. Also the layout of multi-acoustic sensor systems poses potential problems in terms of mutual interference. A *visual* sensing system does not suffer from such problems. With the images taken from a video camera it is possible to retrieve the relevant information from the payload. Furthermore, as the spreader in the experimental rig is designed as a multi-cable-model (MCM) (Cartmell [14], Morrish [57]), complex dynamics of the spreader can occur during the manoeuvre. It is considered reasonable not to allocate sensors to each cable for individual reading, but to retrieve the required information directly from the spreader itself. Therefore, the *visual sensory system* (VSS) is worth further investigation.

Given the complexity of modern control hardware circuitry such as Digital I/O and Digital-Analogue-converters (DACs) typically embedded in modern experimental rigs,

any end-user must have some knowledge of low-level programming to deal with any potential conflicts between these devices. In addition to this it is also fundamentally difficult to transfer the simulated control strategies to lines of native programming languages. Thus it would be ideal to find a way to get around this laborious work so that the true effort can be concentrated on more important tasks such as control strategy planning etc. In this thesis an efficient and user-friendly programming environment, CAPI, (Crane Application Programming Interface) is developed for this purpose, in which the simulation of a control strategy can be transferred to control the experimental rig directly.

1.4 Thesis structure

The structure of the thesis is shown in Figure 1.4 at the end of this chapter. There are six main chapters:

1. In Chapter 2 a concept for a Global Sensing System (GSS) is proposed, which outlines a closed-loop automatic system capable of guiding the spreader to the location of the target container by using the feedback signals from the actuated degrees of freedom, as well as the position and orientation of the swinging spreader by means of a visual sensing technology.

In the second part of Chapter 2 a concept for a Local sensing System (LSS) is proposed. This system is intended to provide final positional, as well as orientational, misalignment corrections between the axes of the spreader and the target container after the GSS has directed the system within the required tolerance.

2. In Chapter 3 a control strategy in the form of Feedback Linearization Control (FLC) is chosen to contain the dynamics of the GSS. Simulation results using *SIMULINK* [83] and *MATLAB* [82] are also presented.
3. The design of a novel, scaled-down, 1/8 size experimental rig with a new level of crane automation is described in Chapter 4. It offers a platform for investigating different control strategies in practice.
4. In Chapter 5 the algorithms developed for the visual sensing system (VSS) are presented.

5. In Chapter 6 a novel Crane Application Programming Interface (CAPI) is proposed. This interface is designed to use the metaphor of the 'block diagram' to integrate software (control strategies) and hardware (VSS, shaft encodes, DACs etc.) without directly involving complex I/O routines between physical devices. By means of this interface the control strategies studied in Chapter 3 can be transferred to the experimental rig easily.
6. In Chapter 7 the analyses of the previous chapters are applied to establish the real-time implementation of an integrated FLC-VSS system.
7. Finally Chapter 8 presents some conclusions and also some recommendations for future work.

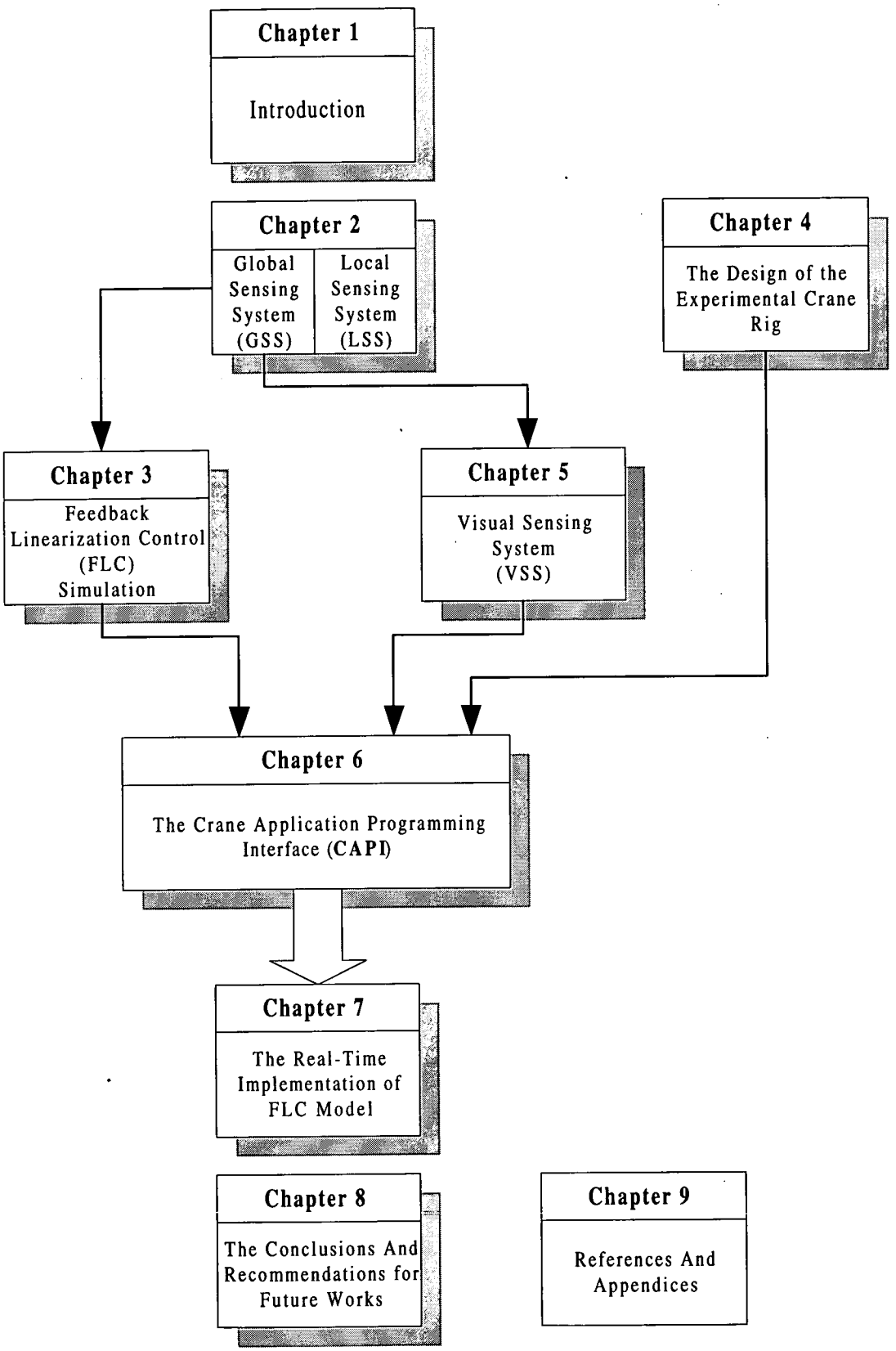


Figure 1.4 The structure of the thesis

Chapter 2

The Sensing Problem

2.1 Introduction

The technical development of crane control systems has been rapid during the past few years. However, as the size and the capacity of the cranes also keep increasing, the load control of the cranes becomes more difficult and necessary. For example, it was mentioned in Champion [18] that, 'it took earlier generation cranes 6-8 seconds to accelerate to the top trolley traveling speed of 400 ft/min., whereas today's cranes accelerate/decelerate in 3-4 seconds with top speeds of up to 650 ft/min. This change has certainly magnified the horizontal sway inertia by three times.' In order to reduce turn-around time during the crane operation, two ways are generally suggested by the crane manufacturers: (1) to employ more cranes (2) to reduce the average cycle time (Bryfors [9]). In terms of financial as well as operational reasons, the second option is considered as a better strategy. There are many ways of reducing the time needed for one container operation, for example (as suggested by Bryfors [9]) :

- Division of the cycle into several independent parts
- Increased speeds and accelerations
- Automatic operation

Nevertheless, the industrial development has proved that the last suggestion is a more efficient and economical solution (Bryfors [9], Casper [17], Dodman [25], Hytönen [40], Macleod [48], Rudolf [65], Verschoof [89]). Therefore a semi- or fully-automatic crane control system which is capable of handling different situations is needed more than ever.

The fundamental function of the RTG crane is to direct the laden or unladen spreader onto the top of the target container. In order to achieve this, two procedures are

necessarily involved. Firstly, the spreader must be moved from some initial position to another position approximately above the desired container in as short a time as possible and, theoretically, also with minimised swing motion during the manoeuvre. Secondly, a subsequent operation is required to provide the final alignment in order to achieve the required relative positional accuracy between the spreader frame and the structure of the target container.

In the first procedure, two different types of anti-sway control systems, either *mechanical* or *electrical* (Hytonen [40]), can be implemented in cranes at this stage (Figure 2.1). Mechanical systems can also be categorized into two groups: rigid suspension and mechanically damped. The former prevents the load from swaying by using supplementary rigid mechanisms between the trolley and spreader. The latter employs hydraulic actuators and dampers mounted on the rope systems to dissipate the energy of the load sway. However, mechanical anti-sway systems still rely on human operators during their operation. Therefore they should be recognized as *passive* anti-sway solutions. Electrical anti-sway systems, which are either open loop or closed loop, use computers to calculate trolley speed and acceleration to avoid adverse load sway. Open loop systems incorporate the measurement of the load height and trolley position to form a pre-programmed trajectory and then move the load to a known end-position. Nevertheless, this solution cannot damp the initial sway, or the disturbances caused by external factors, like the wind. Closed loop systems, on the other hand, can compensate for those external factors during the manoeuvre and initial sway. Moreover, closed loop systems are also potentially capable of dealing with collisions and other unexpected load movements to give optimal accuracy. Therefore a closed loop automatic sensing system, namely a *global sensing system*, is proposed in section 2.2. This system is intended to guide the spreader to the location of the target container by using feedback signals from shaft encoders fitted to the drive motors on the actuated degrees of freedom, together with decoded information on the position as well as the orientation of the spreader, using digitized images from a video camera mounted on the trolley.

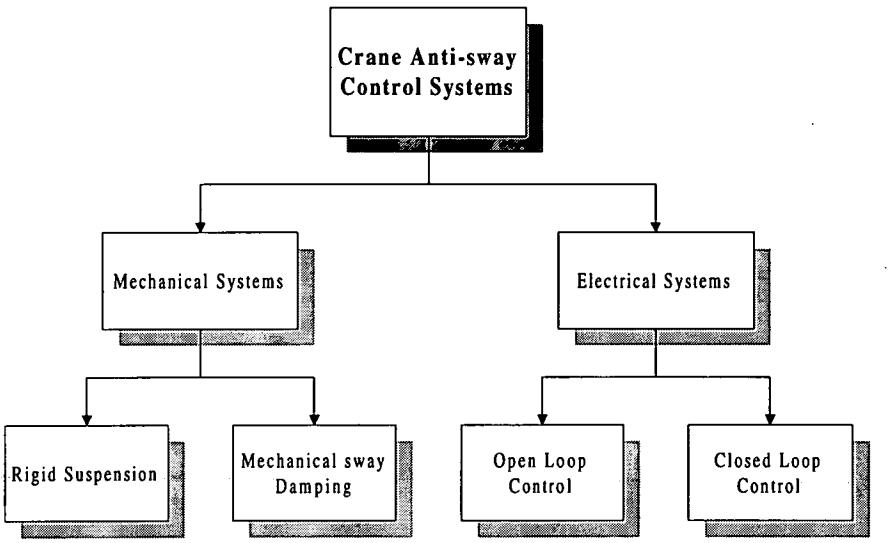


Figure 2.1 Crane anti-sway control systems

In the second procedure, a *human operator* needs to make the final decision as to whether to dock the laden, or unladen, spreader onto another container. This is the least researched problem in the automation of such crane systems. Therefore, in addition to the global sensing system, a *local sensing system* is proposed in section 2.3. This is intended to provide fine positional as well as orientational misalignment corrections after the global sensing system has been used to line up the respective centres of the topside of the container and the underside of the spreader. Laser technology is employed to provide feedback signals for the relative position between the spreader and the target container. The local sensing system can be considered as a supplementary system to the global sensing system to complete the fully automatic crane system, or possibly as an independent retro-fitted piece of equipment for existing cranes.

In this chapter, the necessary principles governing these two systems are investigated and described in full.

2.2 The Global Sensing System (GSS)

Although a considerable amount of research into crane control has been undertaken in recent years, most of it has concentrated on analyzing 2-D planar motion in which the suspended payload and the transporting trolley are together treated as a single pendulum system with a moving support. On this assumption, a mathematical model based on the governing equations of the mechanisms can be derived so that control strategies can then be implemented accordingly. However, one of the most difficult, yet crucial, aspects of any experimental test system is invariably the practical arrangement required for measuring the swing angle of the cable. This is required for the provision of a feedback signal for whatever system is implemented.

The first possible solution is to employ linear potentiometers as shown in Figure 2.2.

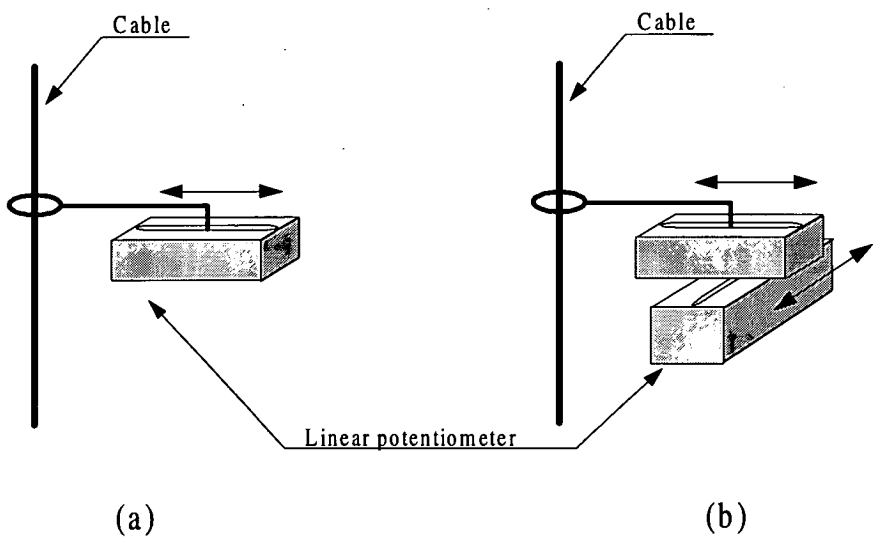


Figure 2.2 Employing linear potentiometers
for measuring the swing angle

There are however disadvantages with this arrangement. First, it is a contact measuring method. Thus the friction of the potentiometer will affect the free movement of the cable as shown in Figure 2.2(a). Secondly, it is not suitable for 2-D measurement. If the cable is no longer constrained to move in the motion of the plane, two co-ordinates are needed to represent these two degrees of freedom. Therefore, two linear potentiometers are required for serving this purpose. As shown in Figure 2.2(b), the

spatial arrangement is by no means easy, and the effect of the friction also becomes even more significant. In the paper presented by Yoon [96], a rotary potentiometer was put in at the pivot point of the cable. This arrangement is, however, only really feasible for a fixed length cable. Burg [11] also showed a similar idea to allocate a disc optical encoder to the pivot point of the cable. Thus the same disadvantage applies as for the rotary potentiometer.

For the non-contacting measuring method, inductive sensors can be configured to gauge the swing angle (ψ). The basis of the inductive sensor principle is an LC-oscillator, which can be influenced from outside by metal objects (Figure 2.3). Because the inductor in the LC-circuit incorporates a ferrite core, the inductance, L, will be altered when metal objects approach or move away from it. The resonant frequency, f, of the LC-circuit is known as $f = \frac{1}{2\pi\sqrt{LC}}$ where L is the inductance of the inductor and C is the capacitance of the capacitor. Therefore by plotting a graph of the output voltage versus the resonant frequency it is possible to determine the linear displacement between the core of the inductive sensor and the metal object. The disadvantages of this setup are: firstly, it is only an approximate measurement because the displacement of the top of the metal object or cable itself will be smaller than the displacement of the bottom and secondly, it presents certain difficulties in cases where it is developed into 2-D measurement because two of the inductive sensors are likely to influence each other. This method is used in the research presented by Yamada et.al [94], Kijima et.al [45], and Butler et.al [10], all of which study 1-D experiments where the payload undergoes planar motion.

An alternative to this is to use ultrasonic sensor devices, as discussed in Hamalainen et.al [35], and Marttinen [51][52][53]. An ultrasonic transmitter is installed on the payload which moves in the motion of the plane, and three receivers are placed on the trolley measuring the signals from the payload. The phase differences define the position changes of the transmitter. The difficulty in implementing this technology to 2-D measurement lies in the interference between more than one ultrasonic transmitter working at the same time. Also, it is difficult to isolate the echoes from the other structures in the experimental rig. A visual sensing system does not suffer from such

limitations. With the images taken from cameras and then digitised it is possible to obtain relevant information about the payload. This advantage is extended further when it is considered that the spreader in the experimental rig is designed for the multi-cable-model (Cartmell [14], Morrish [57]), where complex motion of the spreader is performed. It is thus considered reasonable not to allocate sensors on each cable for individual reading but to directly retrieve the swing angle (ψ) and the polar angle (α), (Figure 3.1, page 71) from the spreader itself. With all the above advantages, a *visual sensing system* (VSS) is chosen as the most appropriate for the global sensing system.

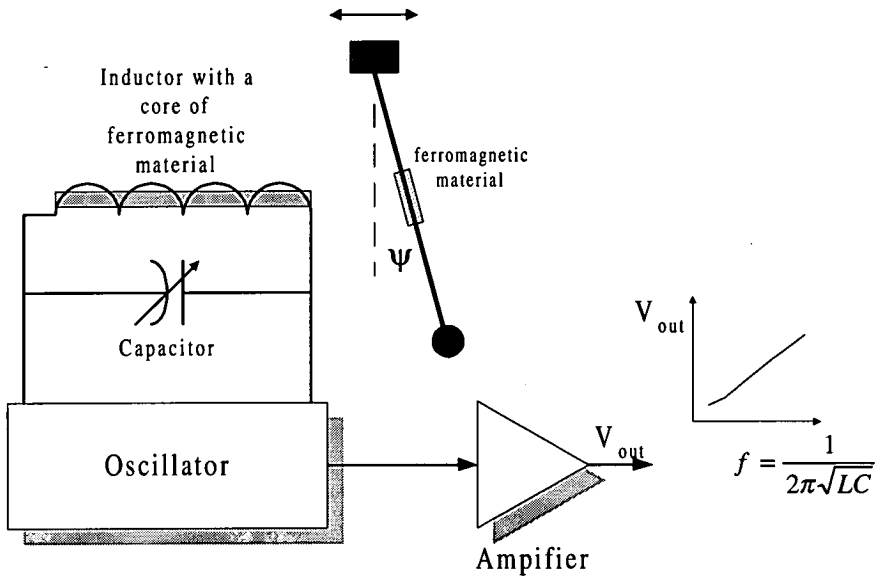


Figure 2.3 Employing inductive sensors to measure the swing angle

2.2.1 The basic concept of image formation (Monocular imaging projection)

A video camera can be considered as a monocular vision element in the same way as a single eye. For normal vision one uses two eyes to distinguish the distance between an object and the viewer by using the distance between the eyes to triangulate a sensation of perceived depth. Thus it is necessary to use two cameras to locate all the physical positions of the desired object under normal circumstances. However, using two cameras will significantly increase the cost and the overall complexity of the sensing system. Moreover, it becomes more impractical to use a two-camera system when operating in the real-time mode. Therefore, finding a feasible means of using a one-camera system is considered desirable.

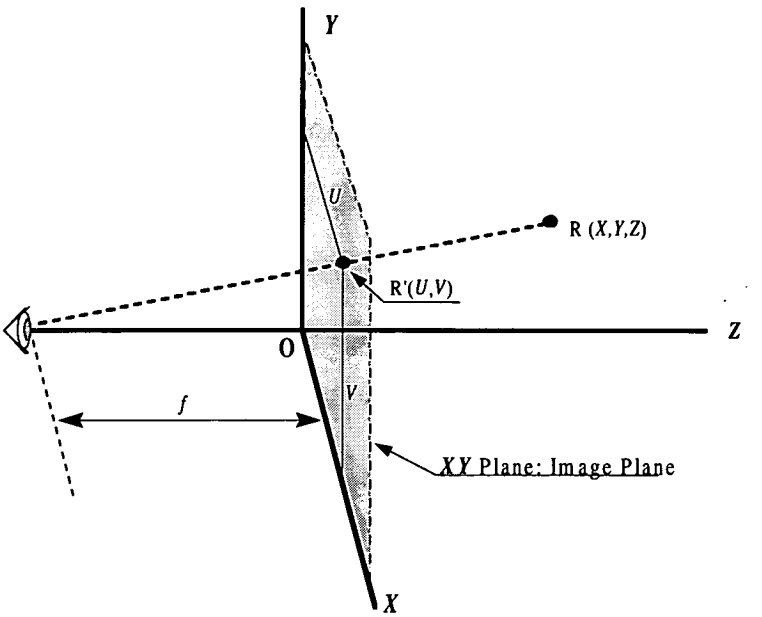


Figure 2.4 Geometry of Image Formation

Viewing a point through only one eye, as shown in Figure 2.4, loses necessary information required to describe the coordinate of the point R in *XYZ* by perspective transformation. This is effectively because a 2-D image is used to describe a 3D object; so all the points lying on the line RR′ possess the same coordinate R′(*U,V,0*) on the image plane (Z=0). This phenomenon is called *perspective distortion*. In Figure 2.4 , if the positive Z axis at (0,0,*f*) is looking toward the origin then *f* is the focal length, and the visible world is projected through the viewpoint onto the Z=0 image plane.

A similar triangle argument shows that the image plane point for any world point (*X,Y,Z*) is given by

$$(U,V) = \left[\frac{fX}{f+Z}, \frac{fY}{f+Z}\right] \qquad \text{------- (2.1)}$$

Hence it is necessary to find an alternative method to compensate for the effect of losing the depth information if only one camera can be involved in the application. One very important piece of data already available is the cable length, *l*, between the trolley and the cable attachment points of the spreader. The next section will show the mathematical models which relate this to other parameter, and which determine the necessary data for further applications to a single camera system.

2.2.2 A mathematical model for a single centrally located video camera mounted under the trolley

(i) A basic model

A trolley-spreader system is depicted as shown in Figure 2.5 with a video camera facing down to 'observe' the spreader (Figure 2.6), where the centre of the lens on the camera is made coincident with the central point of the trolley, P. If the so-called image plane (**I**-Plane) *parallel* to the trolley (i.e. plane **ABCD** // X_T-Y_T plane // ***I*** - plane) is inserted between the spreader and the trolley, as shown in Figure 2.7, *PE"* will intersect the image plane with a point $I_{E"}$. The mathematical model which follows is intended to obtain the coordinates of $E"(x_E , y_E , z_E)$ with respect to given information. This is the coordinates of A, $I_{E"}$ relative to the **T**-coordinate system, and also the length of the cable, *l*.

(Note: (1) *E"* denotes one of the attached points on the moving spreader. When the spreader is stationary, *E"* is coincident with *E* .

(2) the **T**-coordinate system is attached to the ground directly underneath the trolley)

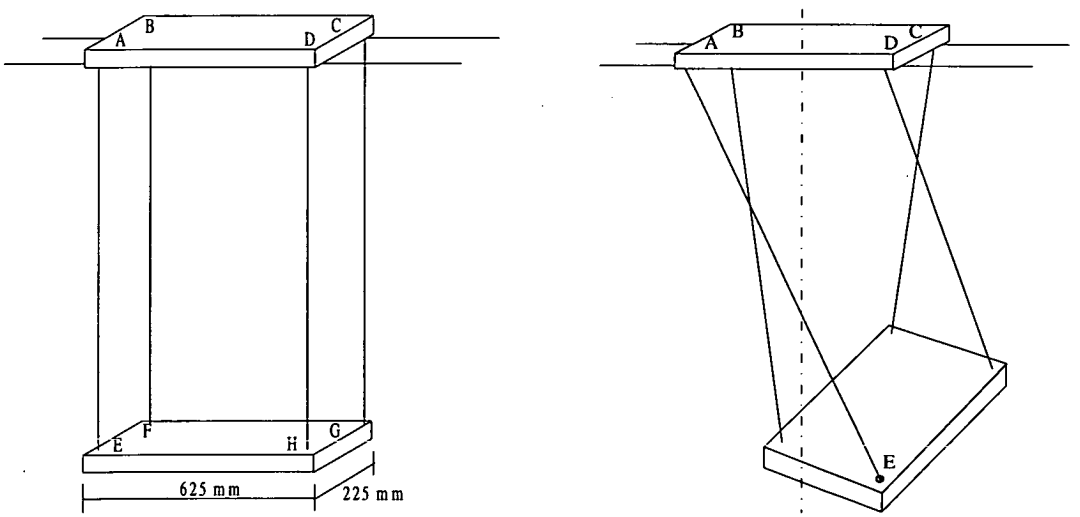


Figure 2.5 Trolley-spreader system

N.B. : The dimensions shown above are to 1/8 scale with respect to a typical RTG crane

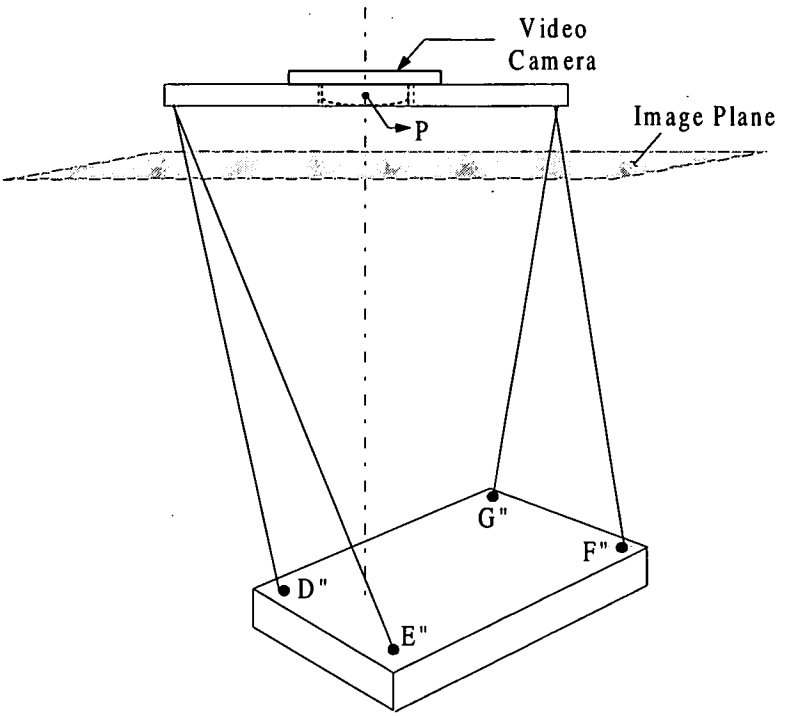


Figure 2.6 The video camera and four attached points

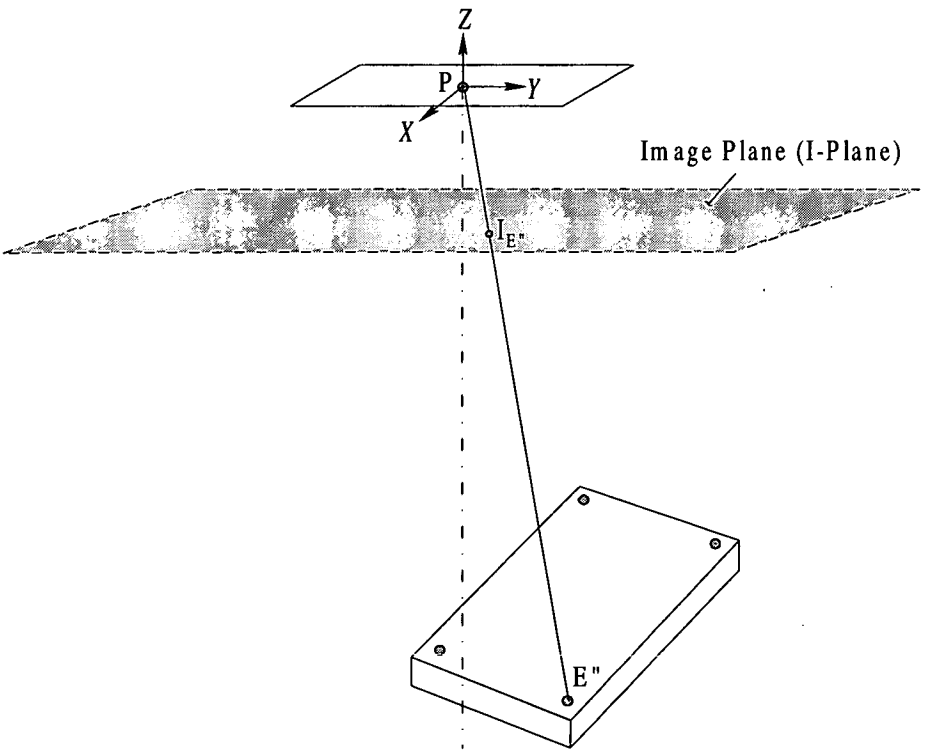


Figure 2.7 The image plane

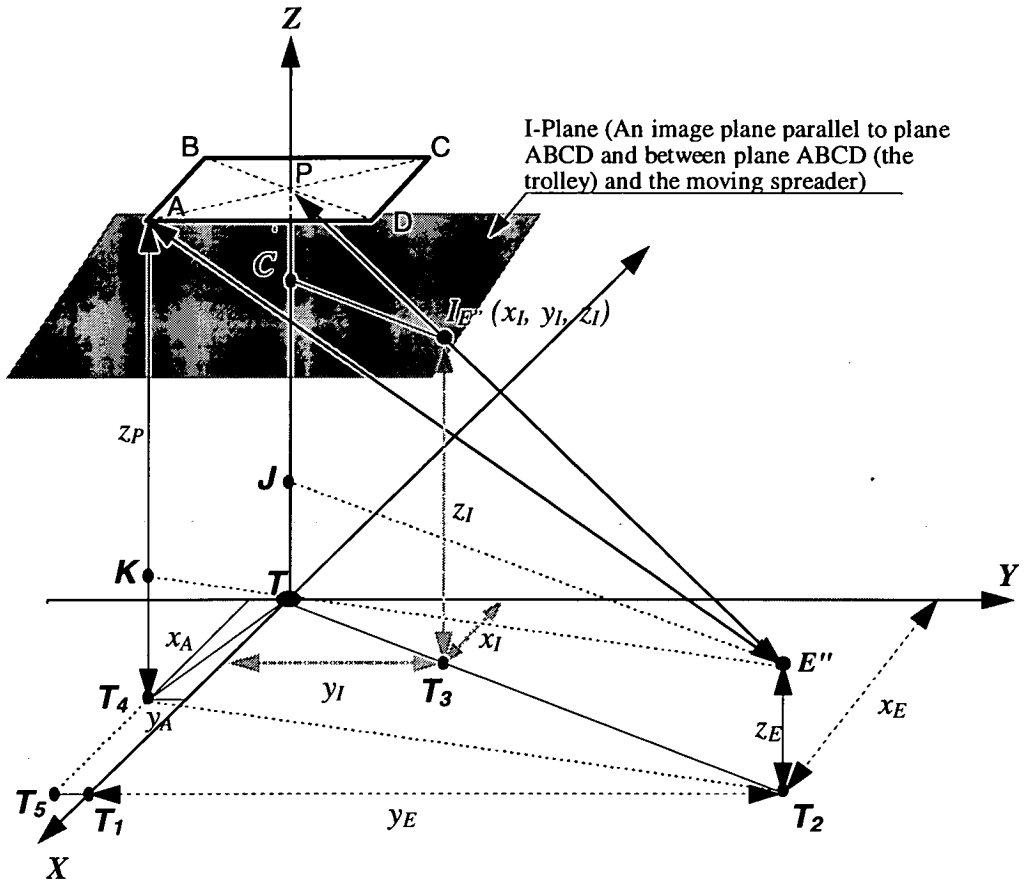


Figure 2.8 Nomination of the basic mathematical model

Given that

P(0 , 0 , z_P)	The central point of the trolley that is coincident with the centre of the lens on the video camera
A(x_A , y_A , z_P)	One of the four points attached the cable on the trolley. Its correspondent point on the spreader is ***E"***
$I_{E''}$ (x_I , y_I , z_I)	the intersecting point on the **I**-Plane along the line ***PE"***
$\|AE''\| = l$	The length of the cable

From ΔTT_1T_2 in Figure 2.8,

$$\frac{y_E}{y_I} = \frac{x_E}{x_I} \qquad \text{------- (2.2)}$$

From $\Delta PJE''$,

$$\frac{PC}{PJ} = \frac{CI_{E''}}{JE''} \qquad \text{------- (2.3)}$$

where $PC = z_P - z_I$ ------- (2.4)

$PJ = z_P - z_E$ ------- (2.5)

From ΔTT_1T_2 ,

$$\frac{y_E}{y_I} = \frac{TT_2}{TT_3} \qquad \text{------- (2.6)}$$

However, $\square JT_1T_2E''$ is a rectangle,

$\therefore$ $TT_2 = JE''$ ------- (2.7)

Also, $\square CTT_3I_{E''}$ is a rectangle. This leads to,

$TT_3 = P_1I_{E''}$ ------- (2.8)

with equations (2.7) and (2.8), equation (2.6) becomes,

$$\frac{y_E}{y_I} = \frac{TT_2}{TT_3} = \frac{JE''}{CI_{E''}} = \frac{PJ}{PC} \qquad \text{------- (2.9)}$$

On substituting equations (2.4) and (2.5) into equation (2.9),

$$\frac{y_E}{y_I} = \frac{z_P - z_E}{z_P - z_I} \qquad \text{------- (2.10)}$$

From $\Delta AKE''$,

$$(AE'')^2 = (z_P - z_E)^2 + (KE'')^2 \qquad \text{------- (2.11)}$$

$$\Rightarrow \quad (KE'')^2 = (AE'')^2 - (z_P - z_E)^2 \qquad \text{------- (2.12)}$$

Also from $\Delta T_4T_5T_2$,

$$(T_4T_2)^2 = (y_E - y_A)^2 + (x_E - x_A)^2 \qquad \text{------- (2.13)}$$

Because $\square KT_4T_2E''$ is a rectangle,

$$\therefore \quad KE'' = T_4T_2 \qquad \text{------- (2.14)}$$

According to the result of equation (2.14), the right hand sides of equations (2.12) and (2.13) are equal. Therefore,

$$(x_A - x_E)^2 + (y_A - y_E)^2 = (AE'')^2 - (z_P - z_E)^2 \qquad \text{------- (2.14)}$$

$\because$ $|AE''| = l$ (The length of the cable). This will lead to,

$$(x_A - x_E)^2 + (y_A - y_E)^2 = l^2 - (z_P - z_E)^2 \qquad \text{------- (2.15)}$$

From equation (2.2),

$$x_E = f(y_E) = y_E\left(\frac{x_I}{y_I}\right) \qquad \text{------- (2.16) --}\textit{linear}$$

Also from equation (2.10),

$$z_E = g(y_E) = z_P - y_E\left(\frac{z_P - z_I}{y_I}\right) \qquad \text{------- (2.17) --}\textit{linear}$$

On substituting equations (2.16) and (2.17) into equation (2.15), it leads to,

$$\boldsymbol{a}y_E^{\,2} + \boldsymbol{b}y_E + \boldsymbol{c} = 0 \qquad \text{------- (2.18)}$$

where,

$$a = 1 + \left(\frac{x_I}{y_I}\right)^2 + \left(\frac{z_P - z_I}{y_I}\right)^2 \qquad \text{------- (2.19)}$$

$$z_P - z_I = PC$$

$$b = -2\left(\frac{x_A x_I}{y_I} + y_A\right) \qquad \text{------- (2.20)}$$

$$c = x_A^{\,2} + y_A^{\,2} - l^2 \qquad \text{------- (2.21)}$$

thus from equations (2.16) ~ (2.21), it is possible to acquire y_E if PC is given. Then the x_E, z_E can be found from equations (2.2) and (2.10), respectively.

From the simple development shown above, it is possible to understand that as long as the **I**-Plane (image-plane) is parallel to the trolley (plane **ABCD**) ; that is, the plane **ABCD** // X_T-Y_T plane // $\boldsymbol{I}$ - plane, equations (2.7), (2.8), (2.9), and (2.10) are established and representative. *This also means that for any parallel plane anywhere between the trolley and the spreader the same development would apply.*

It is relevant to make an important assertion here:

> Assertion: *the length of all four cables must be the same; that is, the cables are equally extended.* From Cartmell [15] and Morrish [57], this

phenomenon only actually happens when the spreader is undergoing *pure translation.* In all other cases of spreader motion there is actually some asymmetry in the cable stretch. This asymmetry is neglected here as it is known to be relatively small for most motion cases.

Based on this, an assumption is made:

Assumption: *the small tilt angle ϕ of the spreader must be ignored in order to apply the foregoing theory* (Figure 2.10).

(ii) A modified model

By using two lighting sources, located close to the cable attachment points on the spreader, it is possible to obtain the co-ordinates of these two lighting sources as shown in the last section. The central point of the spreader and the rotation angle can then be calculated from these two co-ordinates. However, because of limitations in the viewing angle of the lens of a video camera, the potential problem of these lighting sources being outside the viewing range becomes significant for the short cable lengths frequently encountered at high hoist positions. Hence a modification is required to prevent this from happening.

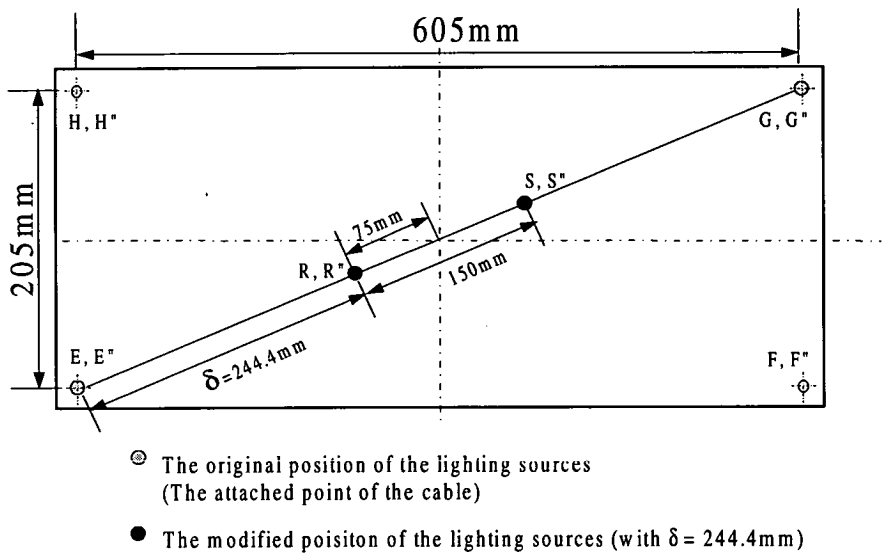


Figure 2.9 The 1/8 scale spreader with the modified location of two lighting sources (top view).

As shown in Figure 2.9, the distance between the two lighting sources on the spreader of the experimental crane has been reduced from the full diagonal length to a chosen length of 150 mm with an offset δ = 244.4(mm). This will reduce the chance of the lighting sources accidentally running out of the field of view of the video camera. However, by applying such an adjustment to the spreader, the mathematical model for obtaining the coordinates of the lighting sources also needs to be modified. Therefore, a new schematic presented in Figure 2.10 is derived from Figure 2.8.

The side view of the triangle *PG"E"* in Figure 2.10 is redrawn as in Figure 2.11(a), where *P* is the central point of the trolley and *G"E"* is the diagonal line of the spreader. *R"* and *S"* are the real positions of the lighting sources, located along the line between *G"* and *E"*. ϕ is the tilt angle between *E"R"* (or *E"G"*, *E"Q"*) and the X_TY_T -Plane (ground). From the assumption of section 2.2.2.i, G"E" and $I_{G"}I_{E"}$ are *parallel* to each other; that is, the triangles *PE"G"* and $PI_{E"}I_{G"}$ are analogous as shown in Figure 2.11(b). This also leads to the fact that the angle γ between the projected *E"R"* (or *E"G"*, *E"Q"*) on the X_TY_T -Plane(ground) and the X_T-axis will be the same as the angle between $I_{E"}I_{G"}$ and the X_T-axis, as shown in Figure 2.12(a), (b).

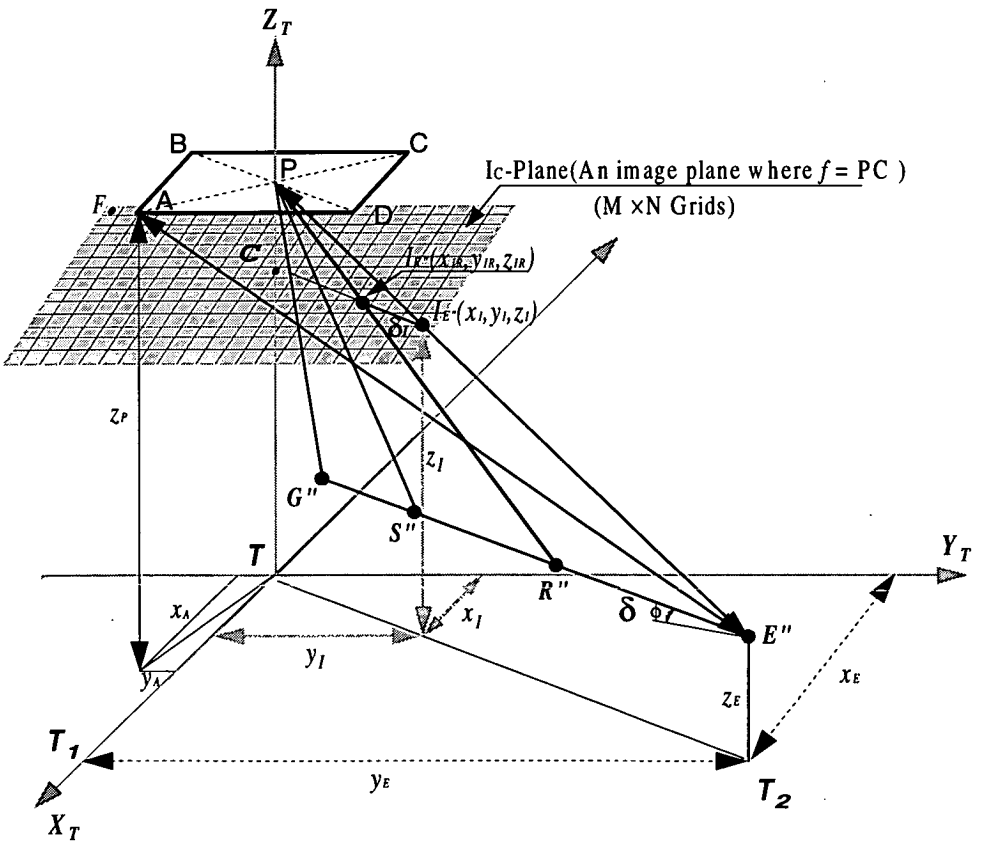


Figure 2.10 Nomination of the modified mathematical model

Amongst the infinite number of parallel image planes between the central point of the trolley **P** and the X_T- Y_T plane (the ground), the one positioned at the focal length f of the video camera lens is of specific interest. Therefore we label this specific plane as the $\mathbf{I_C}$ -Plane with the distance f = PC from the central point **P** of the trolley, when the central point of the lens of the video camera is coincident with it (Figure 2.10).

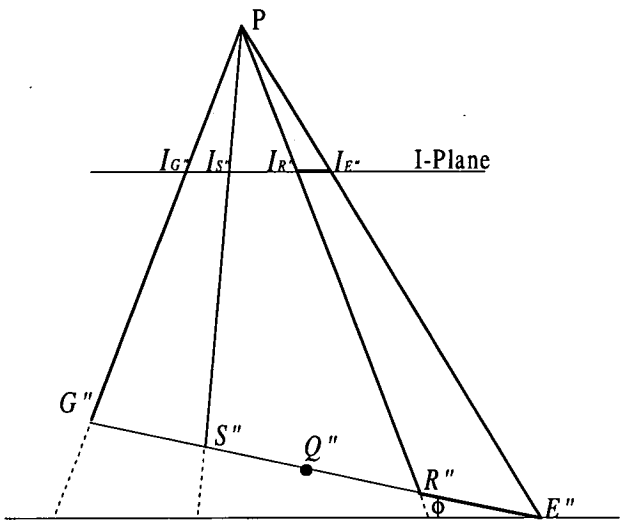


Figure 2.11(a) Tilt angle ϕ

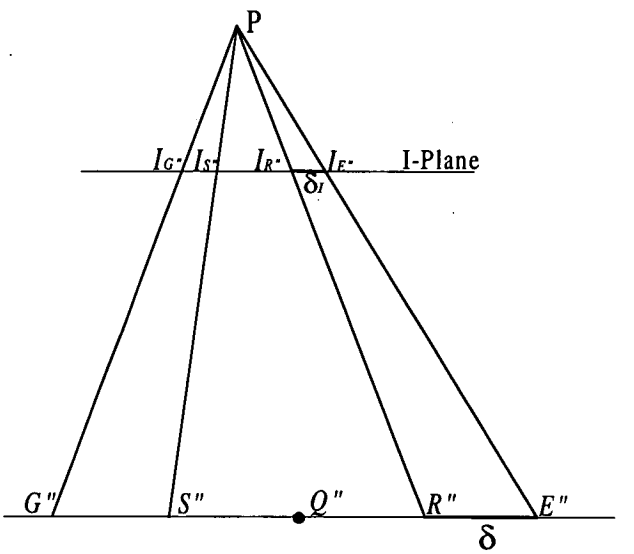
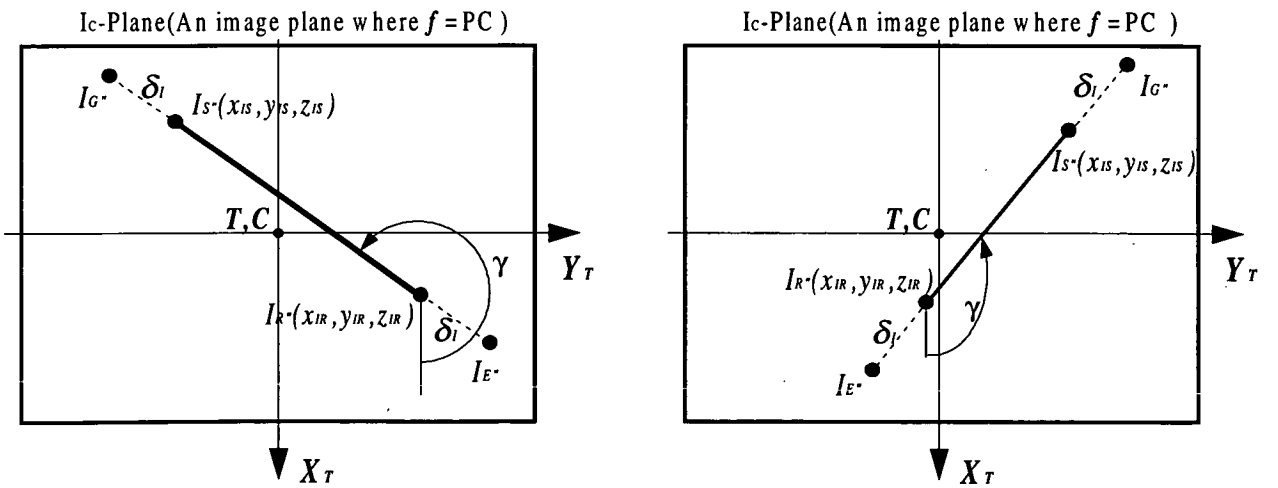


Figure 2.11(b) Tilt angle $\phi = 0$



(a) (b)

Figure 2.12(a)(b) The image plane where f = PC (top view)

From Figure 2.11(b), , it is true that a constant ratio λ exists because the I_C-Plane is parallel to the spreader.

$$\lambda = (\delta / R''S'') = \delta_I / I_{R''} I_{S''} \quad \text{------- (2.22)}$$

With the detected coordinates of the two lighting sources on the I_C-Plane thus,

$I_{R''}\ (x_{IR}, y_{IR}, z_I)$

$I_{S''}\ (x_{IS}, y_{IS}, z_I)$

the angle γ, in Figure 2.12(a)(b) can be calculated,

$$\gamma = \cos^{-1}\left(\frac{x_{IS} - x_{IR}}{I_{S''} I_{R''}}\right) = \cos^{-1}\left(\frac{x_{IS} - x_{IR}}{\sqrt{(x_{IS} - x_{IR})^2 + (y_{IS} - y_{IR})^2}}\right) \quad \text{------- (2.23)}$$

Therefore, the coordinates of $I_{G''}(x_{IG}, y_{IG}, z_{IG})$ (where $z_{IG} = z_I$) in Figure 2.12(a)(b), (the intersecting point of PG'' and the I_C-Plane,) can be obtained,

$$x_{IG} = x_{IS} + \delta_I \cos\gamma \quad \text{------- (2.24)}$$

$$y_{IG} = y_{IS} + \delta_I \sin\gamma \quad \text{------- (2.25)}$$

On substituting equation (2.23) into equations (2.24)(2.25),

$$x_{IG} = x_{IS} + \lambda(x_{IS} - x_{IR}) \quad \text{------- (2.26)}$$

$$y_{IG} = y_{IS} + \lambda(y_{IS} - y_{IR}) \quad \text{------- (2.27)}$$

Also

$PC = z_P - z_I = f$ (f: the focal length of the lens of the video camera)

------- (2.28)

Therefore, on substituting x_I (in equation (2.26)), y_I (in equation (2.27)) and $PC = z_P - z_I = f$ (in equation (2.28)) into equations (2.16) ~ (2.21) in section 2.2.2.i, the attachment point *E"* can be obtained.

In addition, the angle of rotation for the spreader θs can be calculated by,

$$\theta_s = \gamma - \gamma_s \qquad \text{------- (2.29)}$$

where γ_s is the angle between the line of the two lighting sources and the X-axis when the spreader is in the stationary position without swinging.

(iii) Inter-pixel spacing calculations for a specific Image Plane location

From the conclusion of the previous section a specific plane (the $\mathbf{I_C}$-Plane) can be defined at a distance *f* (the focal length of the lens on the video camera) from the central point **P** of the trolley coincident with the central point of the lens of the video camera. Assuming that (1) the depth of field is enough to contain the z-directional displacement of the spreader (in order to have a deeper depth of field, we could set the *f-stop* of the lens as high as possible) and (2) the movement of the lighting sources on the moving spreader will not exceed the field of view of the camera, the images of the focused I_C-Plane can then be taken by suitable image processing hardware, namely a *framegrabber,* capable of digitizing and retrieving information from the composite TV signals. The resolution of the digitized I_C-Plane can be set up by software downloaded to the framegrabber beforehand. This will be discussed further in Chapter 4 and Chapter 5. Now if the framegrabber is configured to an M (lines) × N (pixels per line) resolution then that the grabbed images will be digitized to M×N pixels. This can be considered as creating M×N grids on the $\mathbf{I_C}$ -Plane, as shown in Figure 2.10 and Figure 2.13. It should be noted that the pixels are always generated from top-left to bottom-right as the TV signals scan this way. A corresponding coordinate system **F,** related to the position of the M×N pixels, is employed, as shown in Figure 2.14.

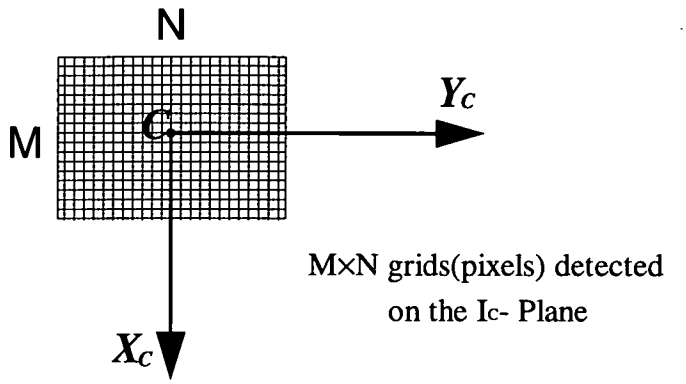


Figure 2.13 M×N grids (pixels) detected on the I_C - plane

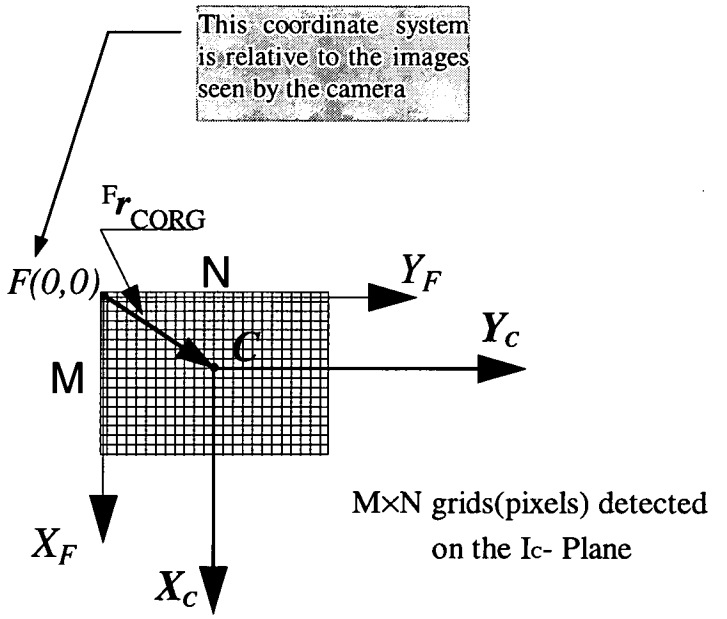


Figure 2.14 The coordinate system ***F***

It is important to note that the position (planar coordinates) of any pixel can only emerge in integer form (Figure 2.15). This implies that the **F**-coordinate system only provides a relative position of the pixels on the $\mathbf{I}_C$ -Plane rather than a physical dimension of the object represented by the pixels. Thus we need to find out the distance between the pixels by some convenient calibration procedure. This is outlined in the next section.

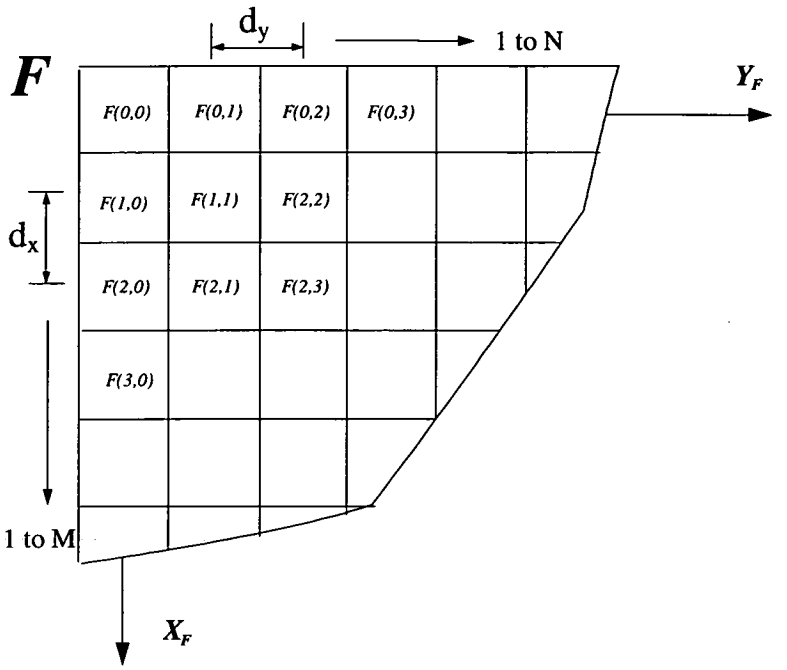


Figure 2.15 The scaling factors d_x and d_y

(iv) A procedure for calibration of the scaling factors (d_x & d_y) between the pixels on the I_C -Plane.

With an object of known length located under the video camera (M_1M_2 // X_TY_T plane // X_CY_C plane) as shown in Figure 2.16 it is necessary to develop a relevant image on the screen ($\mathbf{I_C}$ -Plane). The number of screen pixels , n , can be counted for this object. The other two parameters involved are the focal length f, which is fixed by the lens of the video camera, and the distance L_D from the central point of the lens **P** (i.e. the central point of the trolley) .

Therefore, by similar triangles shown in Figure 2.16 and 2.17 the following formulas are derived

$$\frac{(n-1)d}{D} = \frac{f}{L_D} \qquad \text{------- (2.30)}$$

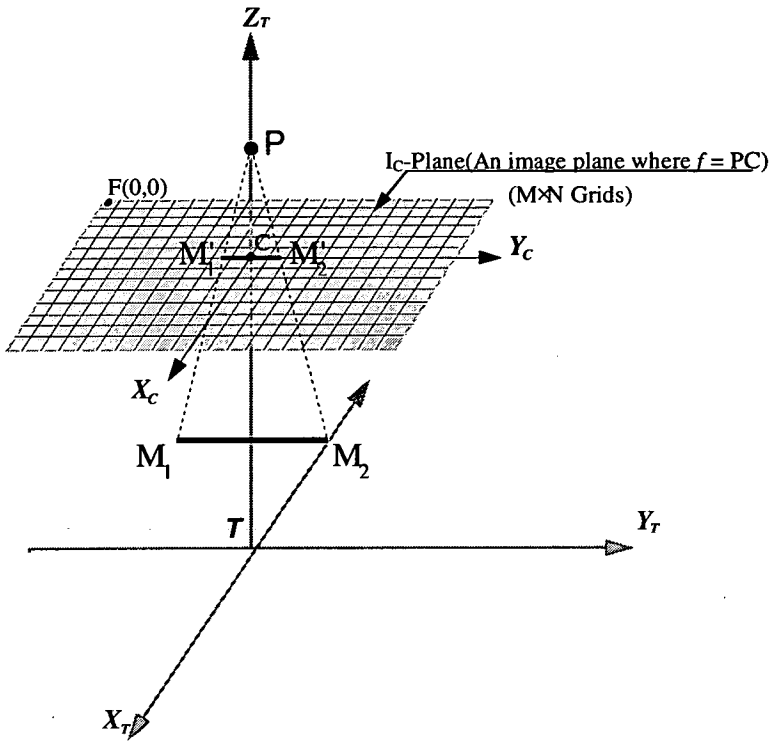


Figure. 2.16 The calibration procedure

$$d = \frac{fD}{L_D(n-1)} \qquad \text{-------} \quad (2.31)$$

However, the distance between the pixels along X_F may well be different from the ones along Y_F. Therefore it is safer to calibrate separately in these two directions. From this the desired *scaling factors* d_x & d_y are obtained for further calculation. This is discussed further in Chapter 7.

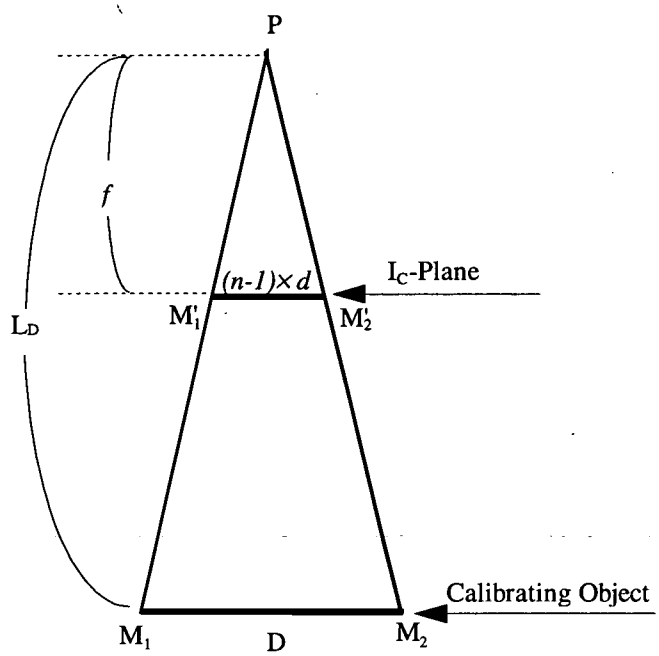


Figure. 2.17 The similar triangle from the calibration procedure

2.2.3 Coordinate Systems, Transformations, and Calculation of the Spreader Position and Orientation

According to the result from the last section it is possible to relate the pixel coordinate-system ***F*** to the physical dimensions of the object in the real world on the $\mathbf{I_C}$ -Plane (where f = PC) using the measured scaling factors, d_x & d_y. However, this information has to be interpreted further in terms of real world coordinates. To do this a world coordinate-system ***W*** is defined together with another moving coordinate system ***T*** fixed right under the trolley, as shown in Figure 2.18. The reason for defining the orientation of the world coordinate-system ***W*** this way is to coincide with the configuration of the control strategy which will be discussed in next chapter. By following the procedure of Figure 2.19, the coordinates of the central point (Q") of the spreader and the rotational angle (θs) of the spreader can then be calculated in real-world terms.

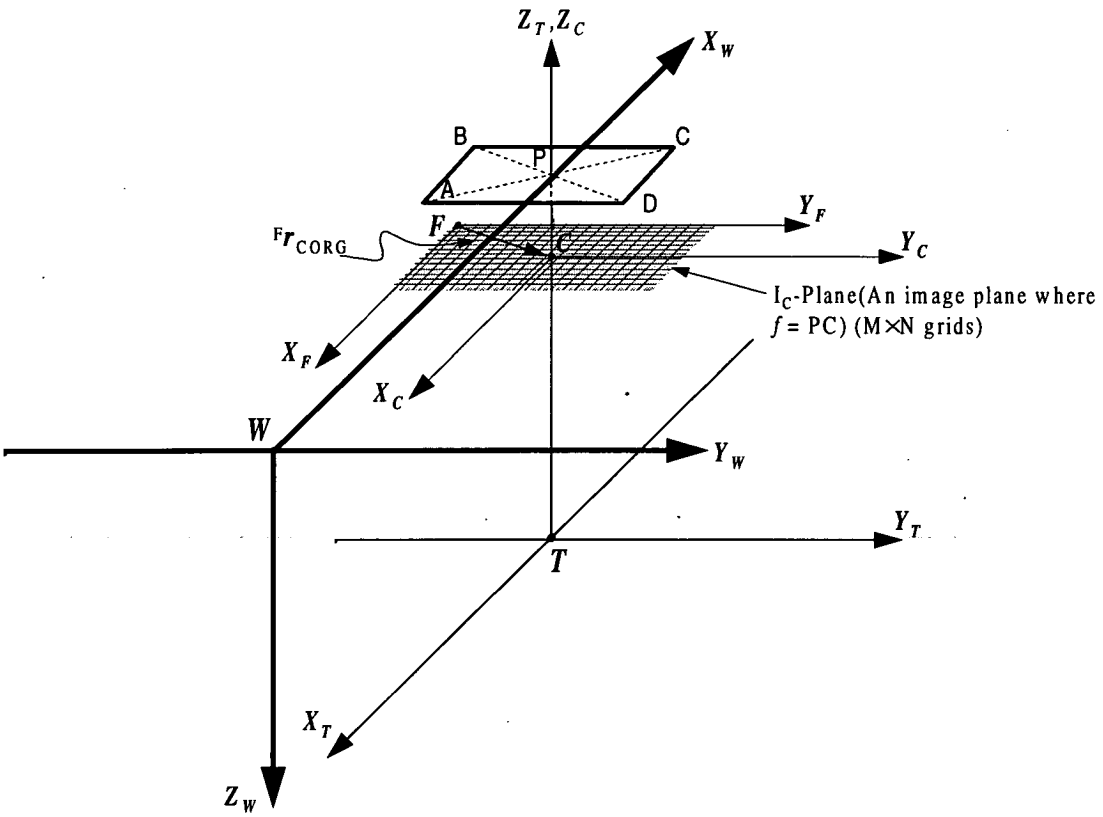


Figure. 2.18 The relationship between coordinate systems

(Transformation.1) $F \longrightarrow C$

The coordinate-system ***C***, as shown in Figure 2.14 and Figure 2.18, is sitting on the same $\mathbf{I}_C$ -Plane (where f = PC) with its origin C in the central point of the digitized images. Therefore, if the selectable resolution of the framegrabber is M×N, from Figure 2.14 and Figure 2.18, the offset vector ${}^F\boldsymbol{r}_{CORG}$, which locates the origin C of the coordinate-system ***C*** relative the coordinate-system ***F***, is,

$$ {}^F_i\boldsymbol{r}_{CORG} = ({}^F x_{CORG}, {}^F y_{CORG}) = ((M-1)/2,\ (N-1)/2) \qquad \text{-------} \quad (2.32) $$

where the pre-subscript, i, denotes that the contents, ${}^F x_{CORG}$ and ${}^F y_{CORG}$, of ${}^F_i\boldsymbol{r}_{CORG}$ are integers.

Also,

$$ {}^F_i P = {}^C_i P + {}^F_i\boldsymbol{r}_{CORG} \qquad \text{-------} \quad (2.33) $$

(Note: ${}^{F}_{i}P$ denotes a point P expressed in the coordinate system ***F*** and the pre-subscript, i, denotes that the contents of ${}^{F}_{i}P$ are integers)

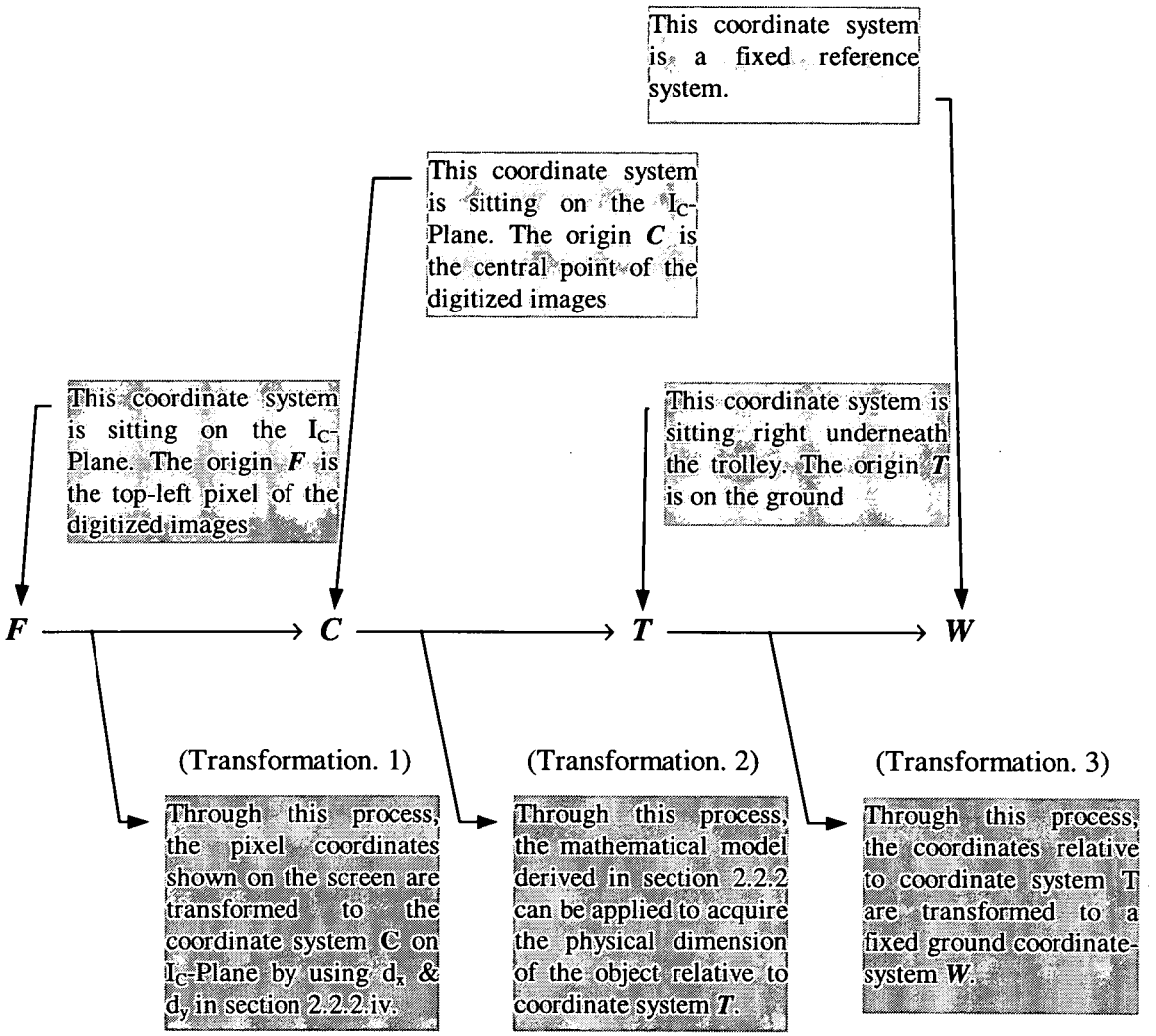


Figure. 2.19 The transformation of the coordinate systems

If the actual reading, $I_{E"}$ and $I_{G"}$ of the two lighting sources from the frame grabber are,

$${}^{F}_{i}I_{E"} = (x_{IE"F}, y_{IE"F}) \qquad \text{------- (2.34)}$$

$${}^{F}_{i}I_{G"} = (x_{IG"F}, y_{IG"F}) \qquad \text{------- (2.35)}$$

then it should be noted that these two coordinates are relative to the ***F*** coordinate-system, and $x_{IE"F}$, $y_{IE"F}$ $x_{IG"F}$, $y_{IG"F}$ are all *integers.*

In order to associate ${}^{F}I_{E''}$, ${}^{F}I_{G''}$ with the ***C*** coordinate-system, first it can be shown, from equation (2.33), that,

$$ {}^{C}_{i}P = {}^{F}_{i}P - {}^{F}_{i}r_{CORG} \qquad \text{------- (2.36)} $$

On substituting equation (2.32) into equation (2.36),

$$ {}^{C}_{i}I_{E''} = (x_{IE''C} , y_{IE''C}) = (x_{IE''F} - {}^{F}x_{CORG}, y_{IE''F} - {}^{F}y_{CORG}) \qquad \text{------- (2.37)} $$

$$ {}^{C}_{i}I_{G''} = (x_{IG''C} , y_{IG''C}) = (x_{IG''F} - {}^{F}x_{CORG}, y_{IG''F} - {}^{F}y_{CORG}) \qquad \text{------- (2.38)} $$

Multiplying equations (2.37) (2.38) by the scaling factors between pixels, d_x and d_y (in section 2.2.2.iv), the physical dimensions of ${}^{C}I_{E''}(x_{IE''} , y_{IE''})$ and ${}^{C}I_{G''}(x_{IG''} , y_{IG''})$ can be obtained as follows:

$$ {}^{C}I_{E''} = (x_{IE''} , y_{IE''}) = (d_x * x_{IE''C} , d_y * y_{IE''C}) \qquad \text{------- (2.39)} $$

$$ {}^{C}I_{G''} = (x_{IG''} , y_{IG''}) = (d_x * x_{IG''C} , d_y * y_{IG''C}) \qquad \text{------- (2.40)} $$

(Transformation.2) ***C*** $\longrightarrow$ ***T***

From equations (2.14) ~ (2.29) in the GSS mathematical model in section 2.2.2, it is possible to transform equations (2.39) and (2.40) to the coordinates relative to the ***T*** coordinate system (Figure 2.18); that is:

$$ {}^{C}I_{E''} = (x_{IE''} , y_{IE''}) \longrightarrow {}^{T}E'' = (x_{E''T} , y_{E''T} , z_{E''T}) \qquad \text{------- (2.41)} $$

$$ {}^{C}I_{G''} = (x_{IG''} , y_{IG''}) \longrightarrow {}^{T}G'' = (x_{G''T} , y_{G''T} , z_{G''T}) \qquad \text{------- (2.42)} $$

The central point of the spreader, Q'', in the coordinate-system ***T*** can then be obtained,

$$^{T}Q'' = (\,^{T}E'' + \,^{T}G'')/2 \qquad \text{------- (2.43)}$$

In addition, the rotation angle of the spreader, θ_S , will be given by applying equations (2.23) and (2.29) in section 2.2.2.

(Transformation.3) ***T*** $\longrightarrow$ ***W***

By taking a point, P, in space as an example the transformation between the coordinate-system ***T*** and the coordinate-system ***W*** can be represented as:

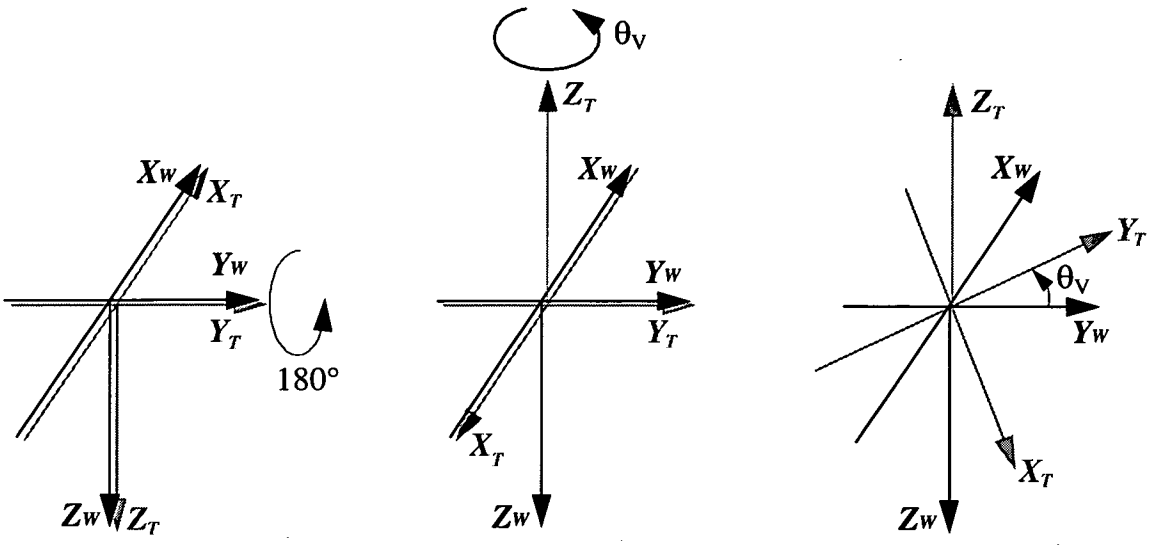
$$^{W}P = {}^{W}_{T}R\ {}^{T}P + {}^{W}\boldsymbol{r}_{TORG} \qquad \text{------- (2.44)}$$

where ${}^{W}_{T}R$ is a 3×3 rotation matrix, describing the orientation of the coordinate-system ***T*** with respect to the coordinate-system ***W***, and ^{W}P, ^{T}P, $^{W}\boldsymbol{r}_{TONG}$ (the offset vector describes the translation between the origins) are 3×1 column matrices.

From Craig [22] & Fu [32], it is understood that the rotation matrix, R , which rotates vectors (or axes in this case), is the same as the rotation matrix which describes a coordinate-system (***T***) rotated by R relative to the reference system (***W***). Based on this, the rotation matrix, ${}^{W}_{T}R$, can obtained by following the operation in Figure 2.20.

First, a symbolic coordinate-system ***T*** is rotated about the axis Y_W by 180°, as shown in Figure 2.20(a). This produces a rotation matrix, $R_{Y_W,180°}$,

$$R_{Y_W,180°} = \begin{bmatrix} \cos(180°) & 0 & -\sin(180°) \\ 0 & 1 & 0 \\ \sin(180°) & 0 & \cos(180°) \end{bmatrix} = \begin{bmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -1 \end{bmatrix} \qquad \text{------- (2.45)}$$



(a) (b) (c)

Figure 2.20 The rotation matrix, ${}^{W}_{T}R$

Secondly, in Figure 2.20(b), this symbolic coordinate-system ***T*** is rotated about the axis Z_W by $-\theta_V$. It should be noted that the negative sign of the θ_V is caused by the fact that it is initialized from the coordinate-system ***W*** rather than ***T***. Therefore,

$$R_{Z_W,-\theta_V} = \begin{bmatrix} \cos(-\theta_V) & -\sin(-\theta_V) & 0 \\ \sin(-\theta_V) & \cos(-\theta_V) & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad \text{------- (2.46)}$$

Thus the rotation matrix, ${}^{W}_{T}R$ is,

$${}^{W}_{T}R = R_{Z_W,-\theta_V} R_{Y_W,180^\circ} = \begin{bmatrix} -\cos(-\theta_V) & -\sin(-\theta_V) & 0 \\ -\sin(-\theta_V) & \cos(-\theta_V) & 0 \\ 0 & 0 & -1 \end{bmatrix} \qquad \text{------- (2.47)}$$

If the offset vector, ${}^{W}\boldsymbol{r}_{TORG}$, between the *origins* T and W is defined as

$${}^{W}\boldsymbol{r}_{TORG} = \begin{bmatrix} x_{TW} \\ y_{TW} \\ z_{TW} \end{bmatrix} \qquad \text{------- (2.48)}$$

In fact, x_{TW} and y_{TW} are the trolley and gantry travelling distances, respectively. z_{TW}, is the height between the ground and the trolley, which is a constant in this case. This relationship between the coordinate-system ***T*** and ***W*** is expressed in Figure 2.21.

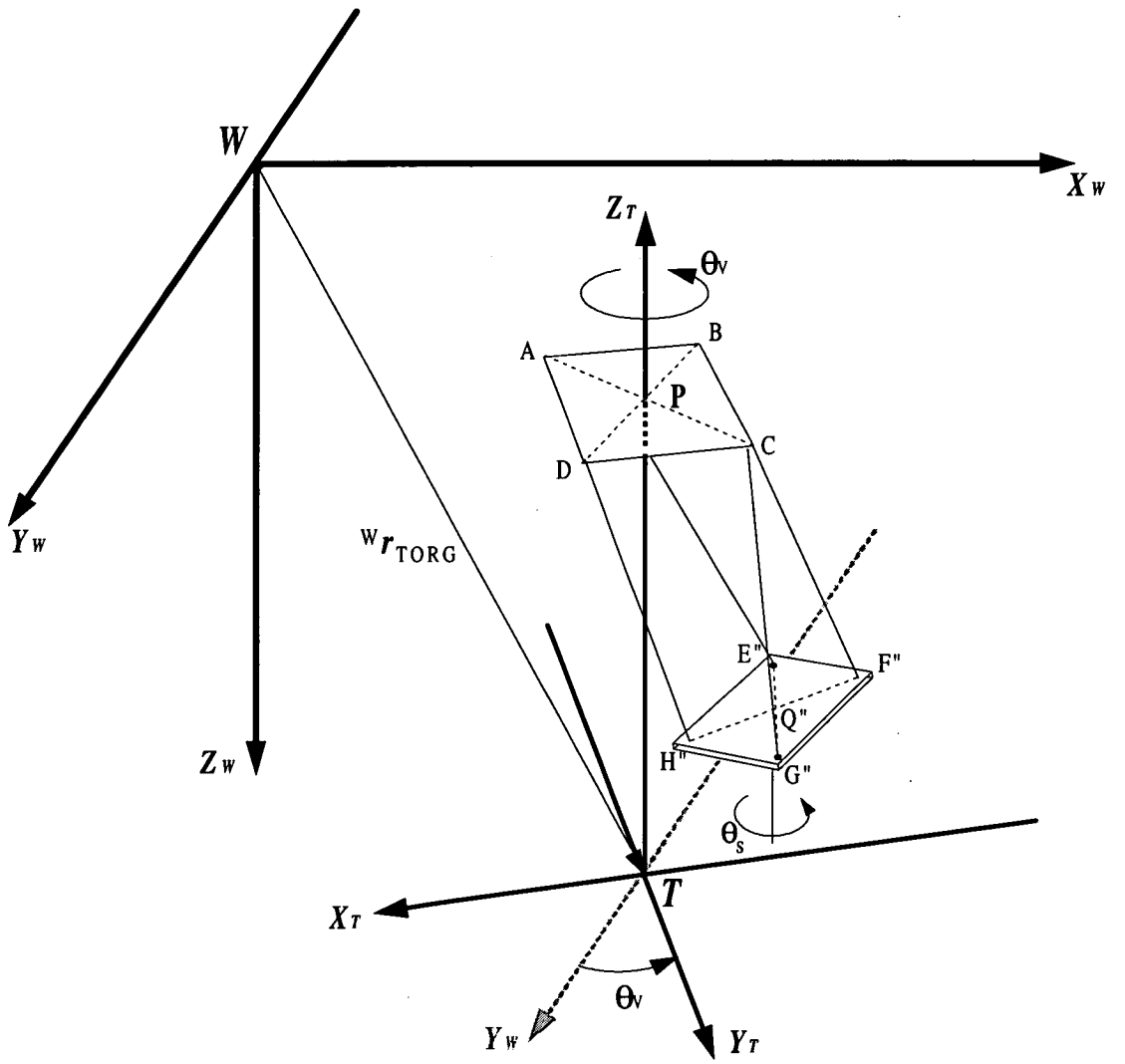


Figure 2.21 The relationship between coordinate system ***T*** and ***W***

2.3 The Local Sensing System (LSS)

Final alignment or 'docking' of the spreader from either the end position obtained from a global strategy (as described in section 2.2) or from a manually obtained rough alignment, is required prior to engagement of the spreader-to-container locking mechanism. This is achieved by implementation of a '*local sensing system*', which differs fundamentally from the 'global sensing system' in that the datum is the container itself, and so the spreader position is directly relative to this. The basic principle and design of a '*local sensing system*' is investigated and discussed in this section.

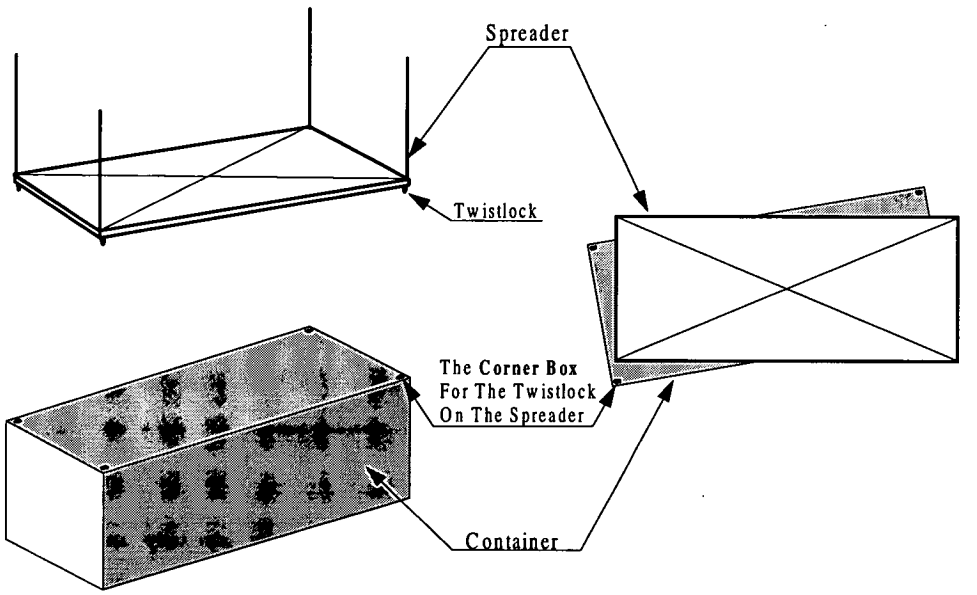


Figure 2.22 An unladen spreader above the target container

After positioning the spreader above the container by manual or globally sensed techniques (Figure 2.22), a *local sensing system* must then be employed to get the spreader perfectly lined up with the container underneath. However, there are two different cases to be dealt with here; first of all, where the spreader is *unladen*; that is to say, there is *no* container slung beneath the spreader (Figure 2.22) in which case the main purpose of the '*local sensing system*' system is to ensure that the twistlocks on the four corners of the spreader are within an acceptable distance from the container corner boxes before docking in. Secondly, where the spreader is *laden* with another container which must therefore be located on top of a stack of containers (up to four

or five high) as shown in Figure 2.23. Thus the sensing system that is proposed must have the ability to cater for these two situations.

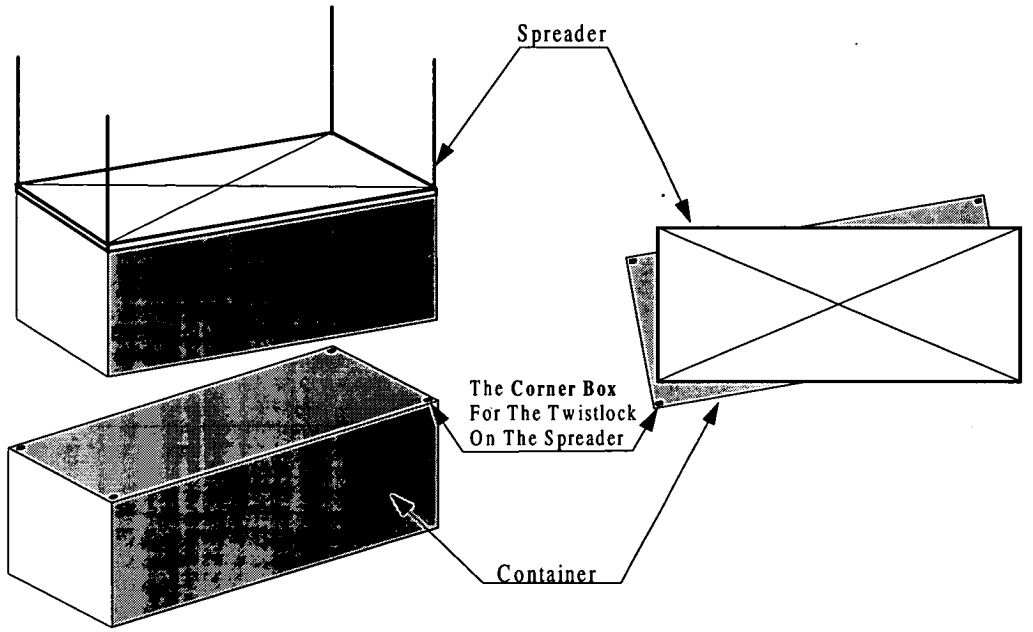


Figure 2.23 A laden spreader above the target container

A paper presented by Wen & Durrant-Whyte [92] has proposed a multiple sensor model to locate the hole of the corner box where the twistlock on the spreader will be accommodated (Figure 2.22, Figure 2.26). By precisely knowing the geometric features of the corner box in advance, the feedback signals from an ultrasonic/infrared transducer model are then compared to this localized information to provide the data for correction. However, there are disadvantages in this approach: (1) The geometric features of the corner boxes on the containers are not reliably consistent because repeated shock loading during the docking operations leads to the shape of the corner boxes changing with time due to wear. (2) This ultrasonic/infrared sensor model is also not feasible whilst the spreader is laden (Figure 2.23), because when the sensor model is directly positioned under the spreader the signals will be blocked by the laden container. It is therefore considered to be more sensible to use more reliable features as the datum for the proposed local sensing system, for instance, the *edges* of the containers.

In section 2.3.1, a variety of suitable sensory technologies are evaluated and discussed. In section 2.3.2, the detailed theory and design of appropriate sensory modules is

proposed. Finally, a misalignment correction algorithm for integration of the whole system is presented in section 2.3.3.

2.3.1 The sensor evaluation and selection

Because freight containers are standardized, it is not admissible to add extra parts to the containers themselves, therefore any contacting sensor technology must be based entirely on spreader mounted contacting sensors. However it is considered likely, indeed almost certain, that such systems would also be prone to eventual failure due to the many shock load conditions encountered during repeated final alignment operations. On this basis, only noncontacting sensing technologies are investigated and discussed for this application.

2.3.1.1 Vision systems using a database

In order to incorporate a vision system into the local sensing system, four steps are needed as follows:

(1) By using monochrome CCD (charged-coupled device) cameras, typical 64 (or 256) gray scale digitized images of the topside of the container could be obtained from commercialized image processing hardware; for example, like the *framegrabber* described in section 2.2.

(2) These digitized images could then be preprocessed using pixel processing algorithms (Data Translation [27]), like a Laplacian filter, for example, to enhance the characteristics of the images.

(3) For recognizing and interpreting the acquired image data, an *a priori* established *knowledge base* (Gonzalez et.al [34]) would have to be invoked to compare differences.

(4) According to the comparison, a decision could then be made to activate the driving mechanism of the spreader to reduce the detected differences.

However, because of the large variation of possible locations and orientations of the containers, a necessarily enormous and complex knowledge base would certainly cause considerable recognition problems and lengthy processing time, even though the

oscillation amplitude and frequency of the spreader is usually comparatively low (0.1~1 Hz) (Waddicor [91]). Also, the computing power required to process all the data in steps (2) ~ (4) would be prohibitive and therefore highly expensive because of the vast amount of image data needed to be dealt with on a real-time basis. Furthermore, any such equipment would be susceptible to damage from shock loading during the final docking operation. Based on the disadvantages described above, this approach is therefore considered to be too complicated and costly to implement.

2.3.1.2 Infrared thermal imaging

Infrared thermography (Merryman [55], Waddicor [91]) employs an array of sensors to detect the amount of infrared radiation emitted from an object's surface. This information can then be multiplexed and passed to an onboard computer. The output is in the form of a digital thermal map displaying temperature gradients across the area scanned by the sensors. This technology has been widely adapted for remote sensing in geographical and astronomical surveys (Key [43]).The basic procedure for processing the thermal maps can be adapted to handle the sort of images described in the last section. However, both technologies suffer from similar drawbacks such as complexity and high associated costs. Moreover, thermal images can be susceptible to interference from weather conditions. Based on these disadvantages this technology is also considered unsuitable for further investigation.

2.3.1.3 Acoustic (ultrasonic) sensors

Ultrasound uses mechanical radiation at a frequency higher than the audible range ($\geq$ 20 KHz) (Bury [12]). When a piezoelectric crystal (typically barium titanate) is subjected to an alternating voltage it expands and contracts in length and the mechanical vibration of the crystal causes adjacent molecules of the medium around it to vibrate, thus the acoustic energy is radiated outwards. If this radiation is directed towards an object then its reflection can be used for detecting dimensions and also proximity sensing for distances down to 1 m in air (Pallás-Areny et.al [60]). This is adequate for many range finding problems in manufacturing and robotic control. The quantity measured is the *time of flight* (TOF) of the sound from the object surface; and

by knowing the speed of sound it is possible to determine the distance of interest (Hickling [38]).

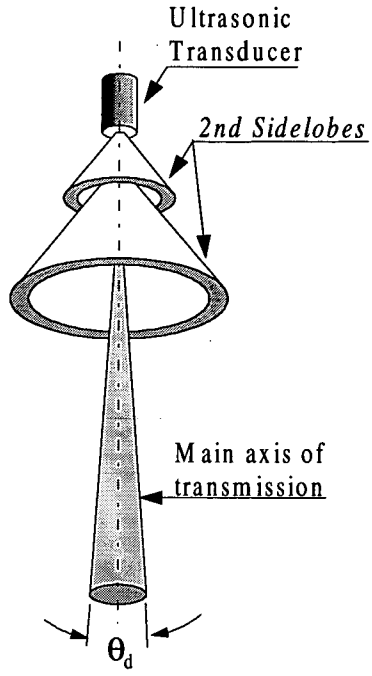


Figure 2.24 An ultrasonic transducer and its transmission signal

Although this type of sensor could potentially fulfill the basic requirements for the local sensing system to detect the edges of the container underneath, an initial concern is that the 2nd *sidelobes* of the transmission relate to a power reduction of one half at the centre of the beam (Wen [92]). These lower intensity signals, as shown in Figure 2.24 and which inevitably radiate out from the transducer, will inevitably cause unwanted echo signals (Waddicor [91]). Although the effect might be filtered out by adjustments of the recognition threshold level in the receiver, this will nevertheless reduce the available range of detection.

In addition, the divergence angle, θ_d ,of the main beam, as depicted in Figure 2.24, which produces a conical shape of the ultrasonic transmission signal, may cause unwanted reflection whilst the spreader is laden. This angle can range from 6° to 20° depending on what type of transducers are chosen. A configuration, as shown in Figure 2.25(a), where an ultrasonic sensor is located outside the spreader, could conceivably detect the edge of the container underneath. However, as shown in Figure 2.25(b), the signals may reflect from the side of the laden container underneath the spreader, and

therefore result in failure to detect the edge. As any local sensing system must accommodate stacking of containers, this limitation precludes the use of acoustic sensors.

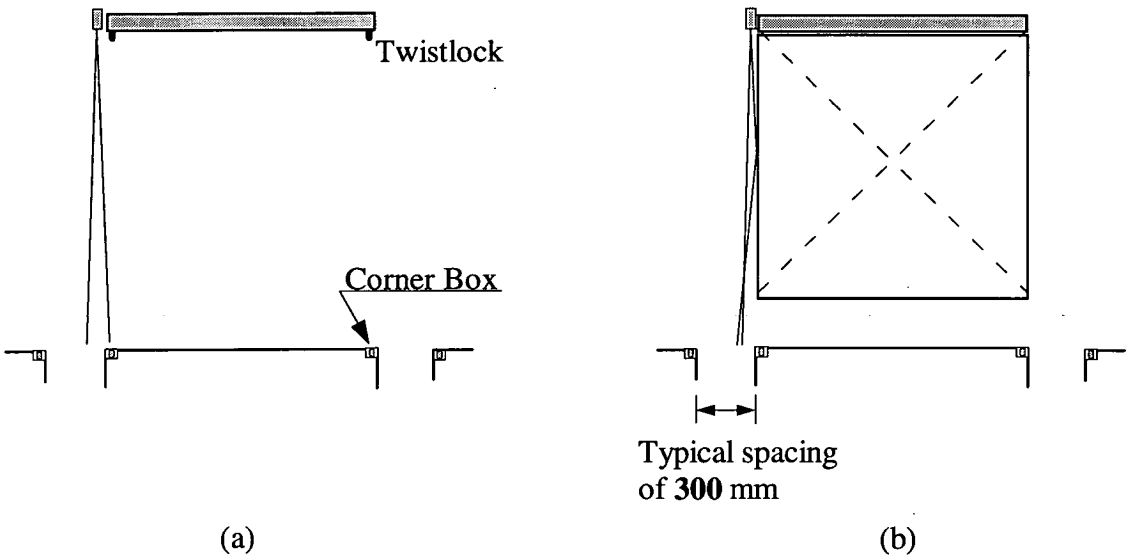


Figure 2.25 The disadvantages of using ultrasonic sensors

2.3.1.4 Laser sensors

Generation of a laser light beam commences with a collection of atoms or molecules in any form, solid liquid or gaseous. These atoms are then excited or 'pumped' to raise them from their lowest energy level into a higher energy level. This can be achieved in many ways but in the case of devices considered here it is achieved by direct electrical excitation using semi-conductors. If a condition called 'population inversion' occurs, whereby more atoms are excited into the higher level than remain in the lower level, so-called *laser* action can take place. If a beam of light tuned to the transition frequency between the two levels is then passed through the collection of atoms, the light will be amplified through the process of stimulated emission, which, conveniently, is *not* directly affected by most environmental conditions, providing that the lens for focusing the beam is kept clean and free from water droplets.

Typical applications for this technology can be found in range finding and navigation (Archibald [3], Yong [95]), where a laser generator produces a short burst of light and a receiver/processing unit measures the time-of-flight (TOF) of the light reflected off an object (Bury [12]). Laser systems are also employed for object feature recognition

(Okamoto [59]), because they are capable of producing very accurate object resolution, due to the narrow beam width of laser light.

For application to the local sensing system, pulses of light energy emitted from the laser generator could be directed at an object surface and the reflection collected using light sensitive devices. Unlike the ultrasonic devices described in section 2.3.1.3, modulated laser light beam can be highly collimated (directionally aligned) provided that the focusing lens is carefully adjusted to minimize beam divergence. Thus the phenomenon shown in Figure 2.25(b) can be avoided. Furthermore laser technology does not suffer from the specularity problem due to its narrow beam width and short wavelength. Despite the fact that lasers are considered hazardous (in cases where an operator is exposed directly to such light) it can be arranged such that the whole sensing system is activated only when the distance between the spreader and the container is within an expected detection range. This could be achieved by using spreader height positional feedback from a suitable encoder.

After taking account of all possible factors, such as the weather conditions, specularity and complexity for the different sensing methods mentioned, it is considered that a laser based system will be more feasible and less complicated to implement. Thus, laser technology is chosen for the proposed local sensing system. The setup and design of a local sensing system will henceforth be explored further in the next couple of sections.

2.3.2 The design of the local sensing system

There are two stages for development of the *local sensing system* in order to cover the two situations of an unladen or laden spreader (Figure 2.22 & 2.23). For the former, three *line laser modules* (SL1, SL2, SS) have been deployed, each of which includes a line laser generator, a lens to collect the reflected laser light and a line-scan optical diode to sense the focused beam. For the latter, four *spot laser modules* (A1, A2, A3, A4) have been arranged, each of which includes a spot laser and an 'area' optical receiver. The general layout of the design for the modules in the 1/8 spreader is shown in Figure 2.26. It should be noted that this experimental scale model spreader is specifically designed for accommodating all the modules around the edges of the

spreader owing to the limited space at the experimental scale chosen. There would not be these problems of space allocation in a full-sized RTG crane. In section 2.3.2.1 and 2.3.2.2, the design and the theory of the line laser module and spot laser module are specified, respectively.

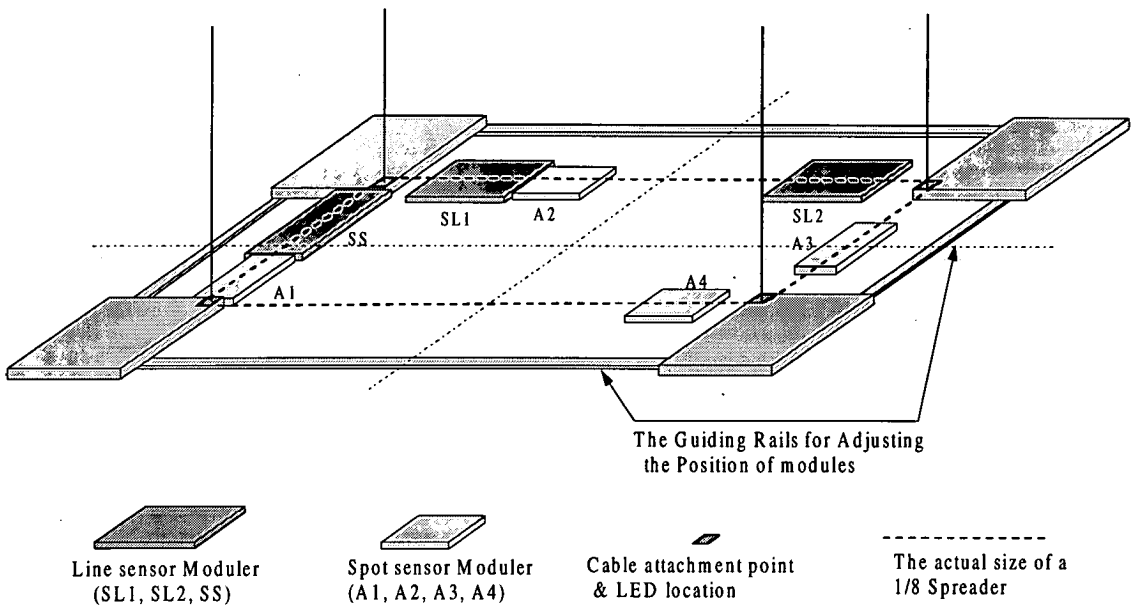


Figure 2.26 The *Experimental Scale Model Spreader* for the local sensing system

Before presenting the detailed design of the modules, the assumptions required for the local sensing system are stated as follows:

(1) The reflecting surface should be reasonably *smooth* and *flat* to ensure that the divergence of the reflected beam is minimized.

(2) The reflecting surface should be reasonably *level.*

(3) The projected line from the surface (for the line laser module in section 2.3.2.1) must be reasonably *parallel* to the pixel sensor line on the receiver device.

(1) and (2) are working environment factors which can generally be verified observation. (3) is controllable by precisely adjusting the modules.

2.3.2.1 The design and theory of the line laser module

The fundamental requirement of the line laser module optics is that the **line** projected by the fan-shaped laser beam is reflected by the edge of the target surface (the topside of the container). This reflection is then collected and focused onto a pixel line sensor (RS [62]) which outputs a voltage directly proportional to the amount of the incident light. Therefore, based on this output voltage, it is possible to detect the position of the spreader relative to the container underneath. For instance, if the maximum voltage from the pixel line sensor is V_{max}, this value indicates that the maximum amount of reflected laser can be detected from the topside of the container. By carefully locating the line laser generator and the receiving end (including the pixel line sensor and focusing lens), as shown in Figure 2.27, Figure 2.28, 1/2 V_{max} can be used to verify the detection of the edge of the container.

There were originally two types of component arrangement proposed for the line laser module, and these are discussed in details in Huang & Cartmell [39]. After careful consideration, it was decided to choose the second arrangement because the first arrangement (Figure 2.27) has to have either the laser generator or the line-scan diode (pixel line sensor) located outside the boundary of the spreader. This presents a potential risk of damage to the sensing system due to the narrow gaps (typically 300 mm as shown in Figure 2.28) between adjacent stacks of containers. Furthermore, this setup is unable to fully make use of the sensing range of the pixel line sensor because of the geometry of this configuration (Huang & Cartmell [39]). These disadvantages tend to preclude the feasibility of this approach.

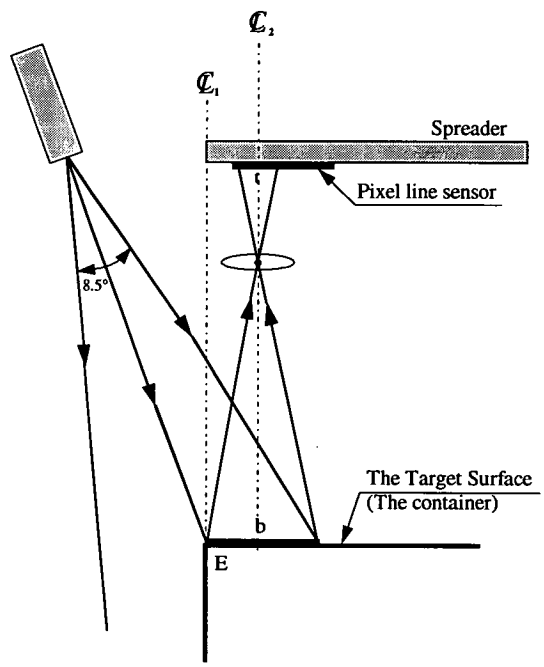


Figure 2.27 The first arrangement

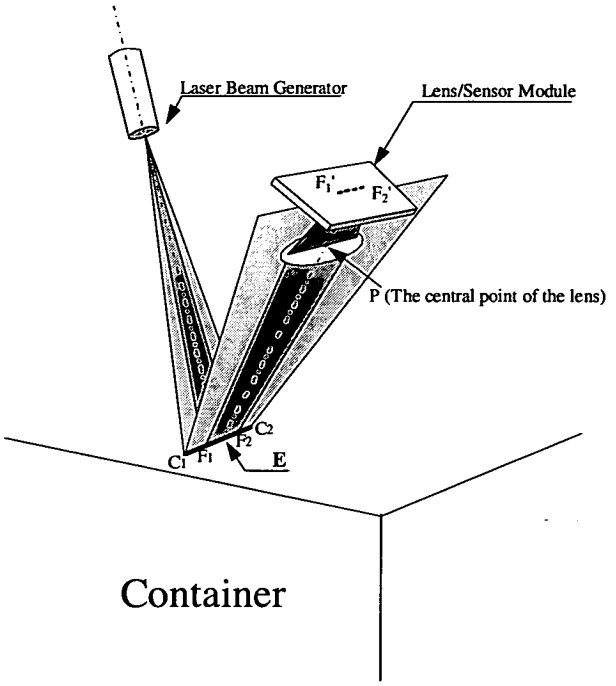


Figure 2.28 The second arrangement

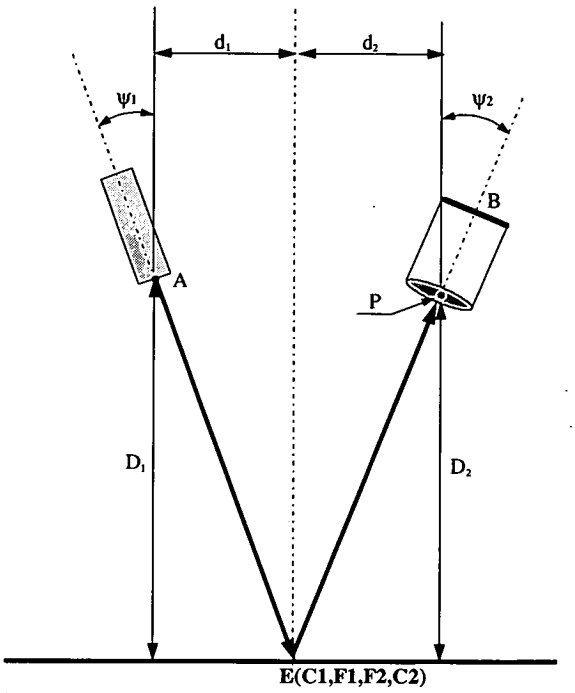


Figure 2.29 The side view of Figure 2.28

The second approach is to locate the laser beam and lens/sensor module right on the edge of the spreader as shown in Figure 2.28 and Figure 2.29. As a result, the weakness of the previous approach shown in Figure 2.27, of being unable to utilize the maximum capacity of the pixel line sensor fully and also possible damage to the modules, can be avoided. The basic idea behind this approach is to direct the central line AE (Figure 2.29) of the laser beam fan to project to the middle point of the pixel line sensor. However, as shown in Figure 2.28, the fan-shaped beam from the laser is divergent. If this V-shaped AEB is 'unfolded' into the triangular shape shown in Figure 2.30 it reveals that the sensor will not detect the full length of the incident line, C_1C_2, on the surface but a fraction of this line, i.e. F_1F_2, and this will depend on the diameter of the lens used.

From Figures 2.29 and 2.30 a mathematical model for acquiring all the important parameters for the line laser module is derived as follows:

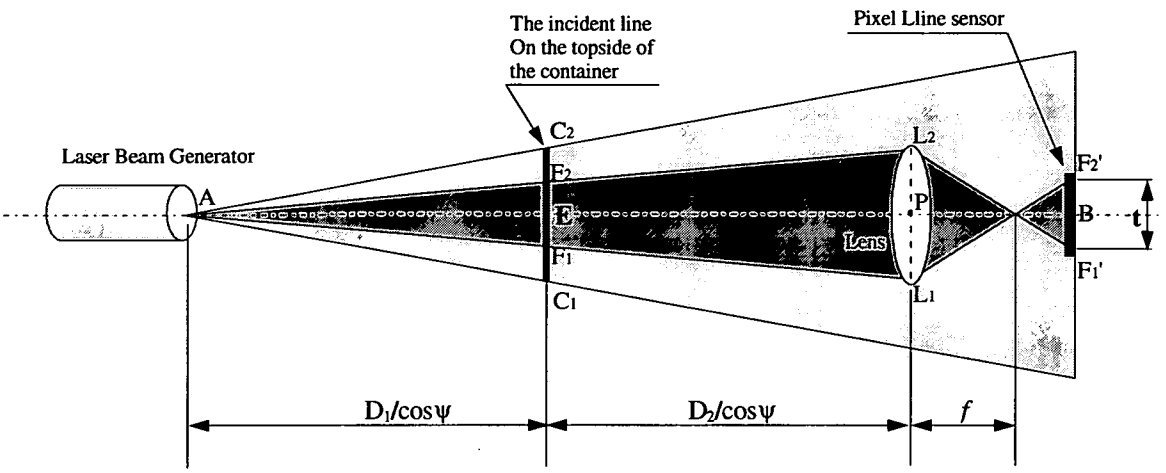


Figure 2.30 The relationship between the laser generator and the pixel line sensor

Nomenclature :

****

ψ_1 ,ψ_2 : The angles between the vertical and the centre-line of the laser beam fan. This is assumed to be the centre line of the cylindrical body of the laser generator.

b : The ***effective*** length of the incident line on the target surface. $b = F_1F_2$.

b_{max}: The maximum effective length of the incident line on the target surface.

D_1 : The vertical distance from the centre of the lens of the laser generator to the reflected surface.

D_2 : The vertical distance from the centre of the lens for the sensor to the reflected surface.

t : The length of the focused reflected line on the sensor. $t = F_1'F_2'$ (t_{max} is the maximum length of t, which can be found in the manufacturer's data sheet (RS [62])).

f: The focal length of the lens.

m: Magnification of the lens.

d_l The diameter of the lens ($d_l = L_1L_2$).

****

Because the incident angle and the reflected angle of the laser are equal, then from Figure 2.29 it follows that,

$$\psi_1 = \psi_2 = \psi \qquad \text{-------} \quad (2.49)$$

Also both the laser generator and the detector (the lens & the pixel line sensor) should be accommodated in the same module. Therefore it is reasonable to let

$$D_1 = D_2 = D \qquad \text{-------} \quad (2.50)$$

$$d_1 = d_2 = d \qquad \text{-------} \quad (2.51)$$

It follows that the angles, ψ_1 and ψ_2 , are,

$$\psi_1 = \psi_2 = \tan^{-1}(d/D) = \psi \qquad \text{-------} \quad (2.52)$$

The selection of the lens is associated with two important parameters: the focal length, f , and the diameter of the lens, d_l. Because of the spatial constraint of the lens holder on the line laser module, the focusing lens cannot be larger than this physical limit. Thus, according to this fact, the diameter of the lens can be fixed. From equation (2.50), it is understood that F_1F_2 is positioned halfway between vertex A and base L_1L_2 of ΔAL_1L_2, as shown in Figure 2.30. Also because the diameter of the lens, d_l (or L_1L_2), is known, the maximum *effective* length of the incident line on the target surface, b_{max} , is,

EX BIBL UNIV. EDI... JRGEN

$$b_{max} = F_1F_2 = d_l/2 \qquad \text{------- (2.53)}$$

Also the maximum sensing length of the pixel line sensor, t_{max}, can be found in the manufacturer's data sheet (RS [62]).

$$m \text{ (magnification)} = - t_{max}/b \qquad \text{------- (2.54)}$$

(N.B. : The formulae and the sign assignments are the convention of following paraxial optics. Please refer to Ealing [28], and Serway [69])

$$f = \frac{m(-EP)}{1-m} \qquad \text{------- (2.55)}$$

where

$$EP = D/\cos\psi \qquad \text{------- (2.56)}$$

From combining equations (2.54), (2.55) and the diameter d_l, a suitable lens can then be selected.

The theory is that when all three line sensors give the same output then the spreader is aligned directly above the target area. The ideal output should be *half* of the maximum possible voltage from the pixel line sensor for all three sensors. This would indicate that the spreader could be lowered down to pick up the underlying container. An algorithm described in section 2.3.3.1 is derived to correct the misalignment accordingly.

2.3.2.2 The design and theory of the spot laser module

When a container is already slung underneath the spreader, but needs to be deposited accurately on a stack (Figure 2.23), the application of the line laser modules described in the previous section is no longer feasible. Therefore, another design is necessary to deal with this situation. The basic arrangement for this, using '*spot laser modules*' is shown in Figure 2.31.

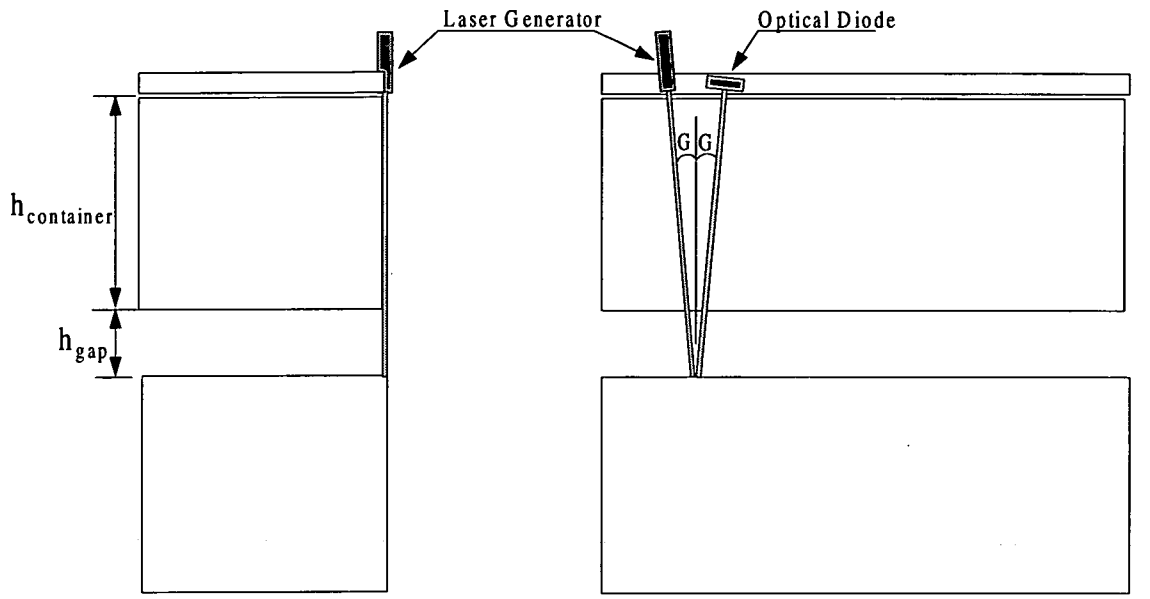


Figure 2.31 The basic arrangement of a spot laser module

Because of the highly concentrated power of the laser beam and the rather short distance between the laser generator and the target surface, it is reasonable to assume that most of the reflected rays mainly bounce back with the reflected angle equal to the incident angle. Consequently, by locating an optical diode on the path of the reflected rays as a receiver, it is possible to create the digital effect of ON('1') and OFF('0'). This output is fundamentally different to the 'analogue' signal obtained from the line laser module. The signal ON('1') is given whilst the laser beam reflects from the topside of the container as shown in Figure 2.31. On the contrary, the signal OFF('0') is set whilst the laser beam falls on the ground and does not reflect back with the correct geometry, as shown in Figure 2.32. As stated in section 2.3.1, there are potential problems for sensors in detecting correct signals in real crane operation during wet weather when the ground may be more reflective than the target surface, which could confuse the sensing system. However, with the design of the spot laser module, as shown in Figure 2.31 and Figure 2.32, it is clear that the primary beam of the reflected laser beam will NOT be 'seen' by the optical diode owing to the angle of the incidence and reflection defining a necessary reflection position below.

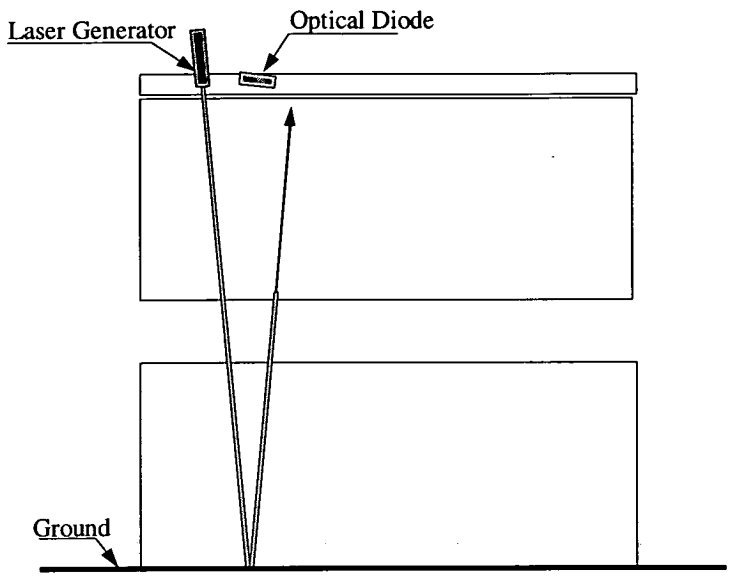


Figure 2.32 The condition where no signal received by the optical diode

The are a couple of parameters to be decided upon to complete this configuration:

(1) The angle G (Figure 2.31)

(2) The distance between the laser and the receiver device (optical diodes etc.)

(3) The height from the target surface of the container to the module ,($h_{container}$ + h_{gap}). It will be a function of the power of the laser and the sensitivity of the receiver device.

The theory is that when all four of these spot laser modules on each side of the container are all giving the same signal OFF('0'), then the container slung under the spreader must be aligned with the target stack of containers. This can be used to enable the control system to lower down the payload. The algorithm shown in section 2.3.3.2 has been derived to correct the misalignment using the 'four OFF case' as a target representing alignment.

2.3.3 The misalignment auto-correction algorithm of the local sensing system

The proposed misalignment auto-correction algorithm has two main sections which are related to the conceptual design for the overall local sensing system. In section 2.3.3.1 and section 2.3.3.2, algorithms associated with the line laser modules and spot laser modules are presented, respectively.

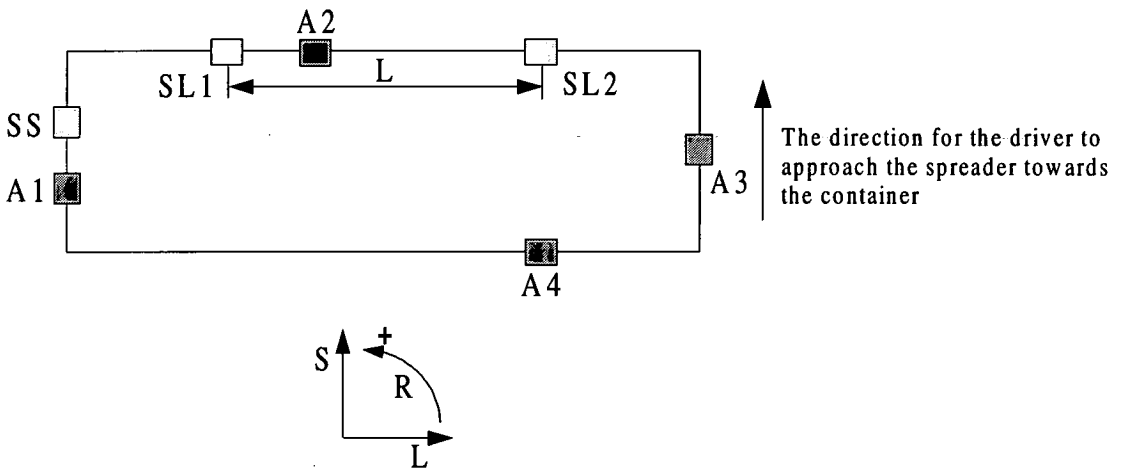


Figure 2.33 A conceptual layout for the location of the modules

Figure 2.31, as a supplement of Figure 2.26, shows an overhead view of a conceptual layout for the location of the modules around the edges of a spreader. A simple coordinate system LS is also defined to explain the operation during steps of the auto-correction. The direction of LS is coincident with the current axes of the spreader. For example, if the spreader is supposed to move in the 'L+' direction, this signals the driving actuators to advance along '→'. If the spreader is required to move in the 'S–' direction, this signals the driving actuators to advance along '↓'. The same criterion applies to 'R', where 'R+' represents the counter-clockwise direction, and 'R–' represents the clockwise direction.

2.3.3.1 The misalignment auto-correction algorithm for the line laser modules

This algorithm is invoked when there is no container slung beneath the spreader. The flowchart in Figure 2.40 at the end of the chapter defines the algorithm for the correction of misalignment using the three line laser modules, SL1, SL2, SS. Lengths b_{SL1} b_{SL2}, and b_{SS} are defined as the effective lengths of the incident light-lines on the topside of the target container at each sensor SL1, SL2, SS respectively. These are transformed to representative voltages by means of appropriate hardware. The value of b_{max}, as stated in section 2.3.2.1, represents the maximum possible effective length of the incident line detected by the pixel line sensor. According to Waddicor [91], two initial conditions must be satisfied prior to automatic final alignment:

❶ The ± 300 mm (as in a full-size RTG crane) clearance is reached by manual control or perhaps GPS (global positioning system).

❷ An angular accuracy of ± 5° between the long axes of the spreader and of the container is achieved.

In ❶, it is indicated that the maximum displacement between one of the corners of the spreader to the edge of the container will not exceed 300 mm , or 37.5 mm in the 1/8 experimental rig. These two requirements also assume that the spreader will not be too far away vertically from the target container underneath. At the end of the misalignment correction process the expected accuracy will be :

① ± 25 mm (as in a full-size RTG crane), or 3.125 mm (ε) on the 1/8 scale experimental rig.

② ± 2° between the long axes of the spreader and of the container.

The details of each step in Figure 2.40 are described as follows:

Step A: *To ensure that the spreader/container position is within the required range for subsequent steps by moving in the S direction*

The first stages of the algorithms are concerned with the detection of SL1 and SL2. If these two modules do not sense anything, then the spreader will be moved in the

direction of 'S–'. This motion will stop when values for b_{SL1} and b_{SL2} are detected. At this stage, the actual numerical values are not critically important; the necessity is to detect some reflection of the laser line, on either of the two sensors. The small dimensions involved in the optics of the sensing system dictate the importance of whether or not the readings scanned for b_{SL1} and b_{SL2} are equivalent to b_{max}. If this is the case the spreader has moved too far in the 'S–' direction. This can be rectified by moving the spreader in the opposite direction, 'S+'.

The spreader will continue to move in the 'S+' direction until b_{SL1} *or* b_{SL2} is less than b_{max}. At this point, an indicator, denoted 'I_B', is assigned a value of '1'. This indicator is part of a function employed to check that the necessary tasks have been performed. The reasons for including this indicator system, and also more detail of its operation, will become more apparent with further development of the algorithm.

This initial process is represented in graphical form in Figure 2.34.

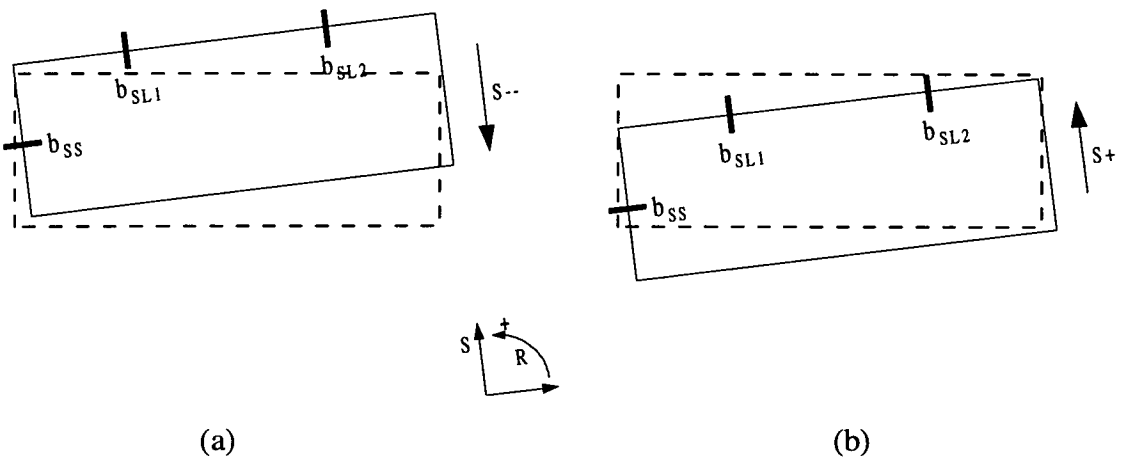


Figure 2.34 - (a) An initial movement to obtain some line reflection.
(b) When $b_{SL1} = b_{SL2} = b_{max}$.

Step B: *To Rectify the orientational misalignment by reducing the difference between b_{SL1} and b_{SL2} by rotation of the spreader.*

Firstly, a check is made to see if the difference between b_{SL1} and b_{SL2} is within the required accuracy. This is done by comparing the absolute value of the difference, $|b_{SL1}-b_{SL2}|$ to $(2°/180\pi)L$ which represents the maximum orientational tolerance, where L is the distance between SL1 and SL2 sensors. If this fails then the value of I_B is reset to '0' and a further check is carried out to find out the scale of the misalignment. To achieve this, the two sensor outputs are compared; if b_{SL1} is greater than b_{SL2}, then the spreader will be rotated in a counter-clockwise direction(R+). However, if the opposite is true, then the direction of rotation is clockwise(R–). This is shown in the diagram in Figure 2.35. This process is continued until $|b_{SL1}-b_{SL2}| <= (2°/180\pi)L$. Another indicator, '$I_C$', is assigned a value of '1'. It is important to note that indicator I_B is still set at '0' at the moment.

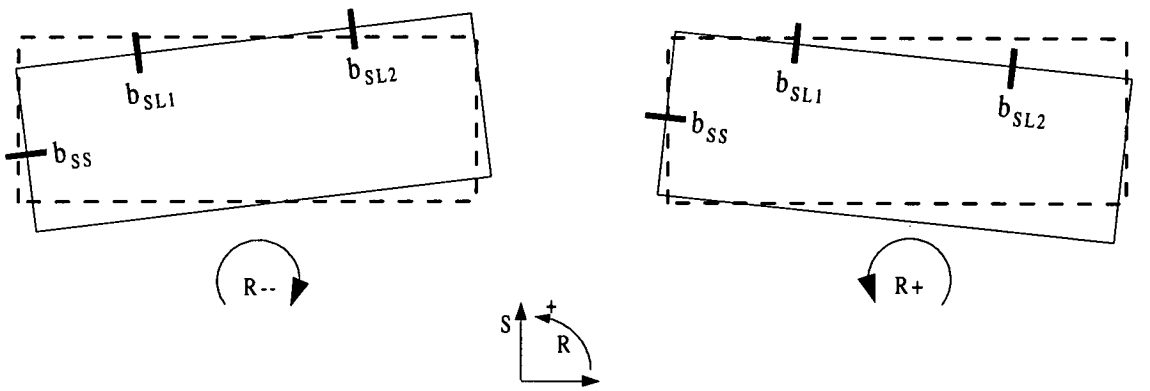


Figure 2.35 Step B

Step C: *To correct the displacement error measured by comparing b_{SS} to the required accuracy (ε or 3.125 mm in the 1/8 experimental rig) by means of translation in the 'L' direction.*

After Step B, the misalignment of the orientation between the spreader and the container has been decreased. However, this is not necessarily an indication that the spreader is in the ideal position above the target container. This is achieved if, and only if, the following conditions are satisfied:

$|b_{SL1}-b_2| < \varepsilon$, ------- (2.57)

$|b_{SL2}-b_2| < \varepsilon$, ------- (2.58)

$|b_{SS}-b_2| < \varepsilon$ ------- (2.59)

where $\mathbf{b_2}$ is equal to **0.5** b_{max}. Therefore, it is necessary for the output of the three line laser modules to satisfy these inequalities. In Step C, the output of SS is scanned and compared to b_2 to check whether it satisfies equation (2.59). If it does not, I_C will be reset to '0' and the spreader will be moved along 'L+' or 'L–', depending on whether b_{SS} is greater than b_2. This movement will continue until equation (2.59) is satisfied. A final indicator, I_D, is then set to a value of '1' and 'I_C remains at '0'. After the execution of this process, the spreader/container position could be the same as that shown in Figure 2.36.

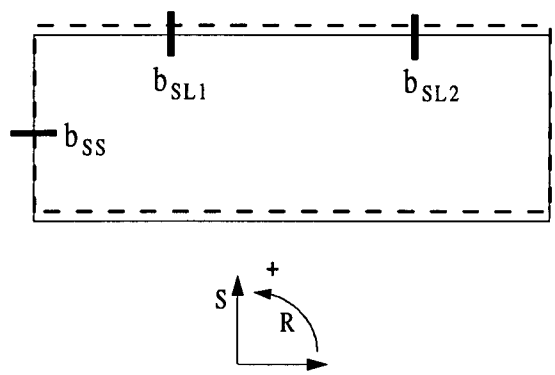


Figure 2.36 Step C

Step D: *To correct the displacement error measured by comparing b_{SLI} to the required accuracy (ε) by means of translation in the 'S' direction.*

The final step in this line-laser module algorithm is the alignment of the long side of the spreader with the corresponding side of the container underneath. This is done by making a comparison of b_{SL1} and b_2. However, if the equality is not satisfied, then I_D is reset to '0' and the alignment process is undertaken in much the same way as the alignment of the short side of the spreader described in Step C; the main difference being that, in this case, any adjusting movements are made in the 'S' direction.

The indicator system is, basically, a function to check that all the necessary conditions are satisfied before the spreader is lowered for picking up the target container. There are three indicators in the system, namely, I_B, I_C ,I_D which correspond to Steps B, C, D, respectively. Prior to entering the main operation of each step, they are set to '1'. If the main or first decision condition (inequalities (2.57), (2.58), (2.59)) in each step is not satisfied, then the indicator is reset to '0' and the system proceeds to a further correction operation. If, and only if, all these three main, or first decision, conditions (inequalities (2.57), (2.58), (2.59)) at each step are met then I_B, I_C, and I_D will all remain '1' and allow the whole process to stop. This will prevent the whole process from wrongly 'checking out' because of the operation of one individual step adversely affecting the achieved accuracy of other steps. The following diagrammatic sequence of Figure 2.37 concisely explains the process:

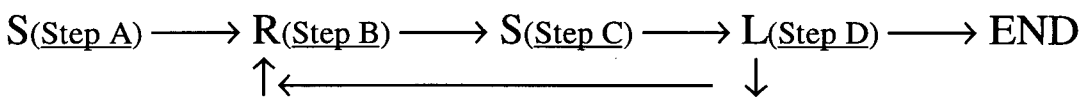


Figure 2.37 The auto-correction sequence of line laser modules

2.3.3.2 The misalignment auto-correction algorithm for the spot laser modules

It is shown in Figure 2.41 at the end of the chapter, that there is an alternative algorithm required for the case in which the spreader is already laden with another container, which is to be unloaded on top of a pre-existing stack of containers.

The output of each spot sensor module is digital; either '1' or '0'. From the setup described in section 2.3.2.2, when all four spot laser modules on each side of the container are signaling '0' then this indicates that the laden spreader is aligned with the target stack of containers. The details of the algorithm are as follow:

Step A1.1 and A4.1 : *To determine the direction D_L, D_S for Steps A1.2 and A4.2.*
As shown in Figure 2.26, A1 is located on the shorter side of the spreader. If '0' is output from the optical diode on this module it means that the side on which A1 is located is too far away from the corresponding side of the container on the stack underneath. As a result of this the spreader has to be moved in the 'L+' direction so that D_L becomes 'L+'. The same criterion applies to the A4 module to set up the direction D_S. It should be noticed that $A1(t_0)$ and $A4(t_0)$ define the detected signals at the moment when both Steps A1.1 and A4.1 are initialized.

Step A1.2 and Step A4.2 : *To Correct the offset in the 'L' and 'S' directions.*
According to the directions D_L, D_S set in Step A1.1 and A4.1, the appropriate actuators start to move the spreader until the moment that the current output is the *opposite* of the initial output, indicates that the laser beam has just passed the edge of the container. It should be noticed that A1(t) and A4(t) define the current detected signals and these are updated throughout the process.

Step A2.1 and A3.1 : *To decide the direction of rotation 'D_R' for the orientation misalignment correction.*
After the operation of Step A4, the output from the A2 module is scanned. If the container is being detected ('1'), the direction 'D_R' is set to anti-clockwise, 'R+'. On the other hand, if $A2(t_0)$ is sensed as '0', the final module A3 is checked to see whether its output is '0'. If it is, the whole process is satisfied and the operation can stop. If it is

not, the direction 'D_R' is set to the clockwise direction 'R+' and the system can move on to the final step.

Step A2.2 : *To correct the orientation misalignment.*

According to the direction 'D_R' acquired by Step A2.1 and A4.1 the spreader is rotated correspondingly until the moment that the *opposite* of the initial output is sensed.

Another indicator system similar to the one used in the algorithm for the line sensor modules is also applied to the above process to enhance its robustness. The flowchart is shown in Figure 2.41. The following figure explains the process:

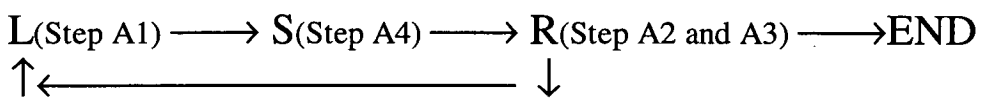
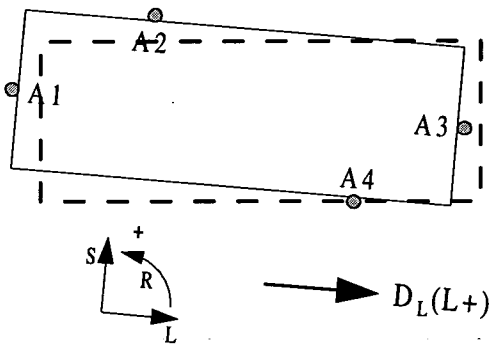


Figure 2.38 The auto-correction sequence of spot laser modules

An example of the execution of this algorithm is shown in Figure 2.39.



(a)

Step A.1.1: $A1(t_0) = '0' \Rightarrow D_L = L+$.

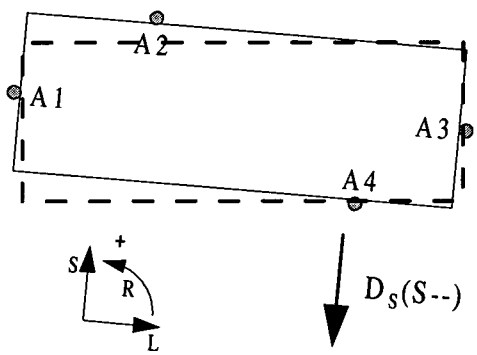
Step A.4.1: $A4(t_0) = '1' \Rightarrow D_S = S-$.

(b)

Step A.1.2:

Move toward D_L (L+).

Stop when $A1(t) = '1' = - A1(t_0)$.



(c)

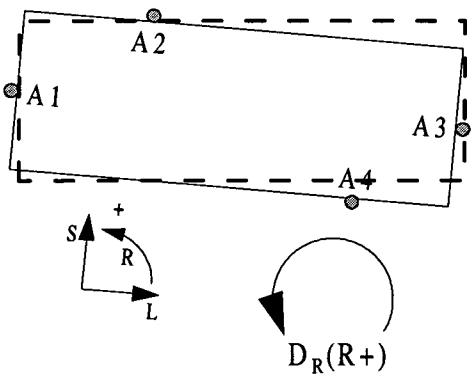
Step A.4.2:

Move toward D_S (S–).

Stop when $A4(t) = '0' = - A4(t_0)$.

Step A.2.1: $A2(t_0) = '0'$

Step A.3.1: $A4(t) = '1' \Rightarrow D_R = R+$.



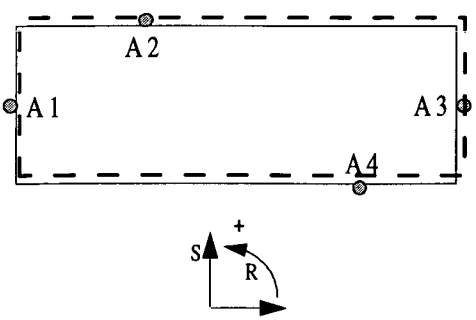
(d)

Step A.2.2:

Rotate about D_R (R+).

Stop when $A2(t) = '1' = - A2(t_0)$.

Figure 2.39 An example of auto-correction algorithm for spot laser modules



2.4 Conclusions

In this chapter, the sensing methodologies for the two proposed systems have been discussed.

Firstly, the *global sensing system* is intended for full automation of the RTG crane in situations where the target container is located at a known position and the RTG crane is located somewhere else (also known). By using the mathematical model in section 2.2.2 and the coordinate transformations defined in section 2.2.3 it is possible to correlate the information of the images 'seen' by the CCD camera on the trolley to the coordinates and orientation of the spreader relative to the world coordinate system. In the next chapter a control strategy for this global system is discussed and the results of simulation are presented. In Chapter 5, a search algorithm for pinpointing the coordinates of the two reference lighting sources on the spreader from the grabbed images of the CCD camera is proposed and implementation requirements are explored.

Secondly a *local sensing system* has been designed and it is seen that this can work as either a supplementary system for the final docking process (after the global sensing system has directed the spreader to a position approximately above the target container), or as an independent retro-fitted piece of equipment for existing RTG cranes. There are two types of sensor module which have been designed to adapt to two distinctly different situations occurring in real RTG crane operation. Three *line laser modules*, each of which contains a line-laser generator, a focusing lens and an optical pixel-line-sensor, are dedicated for resolving the misalignment situation when there is *no* container slung underneath the spreader. The other approach uses four spot laser modules, each containing a spot laser generator and an optical diode, and this is dedicated to the situation when there *is* a container slung underneath the spreader. Two auto-correction algorithms for these two cases are explained in section 2.3.3.2 for adjustment of the spreader using the information feedback from the modules.

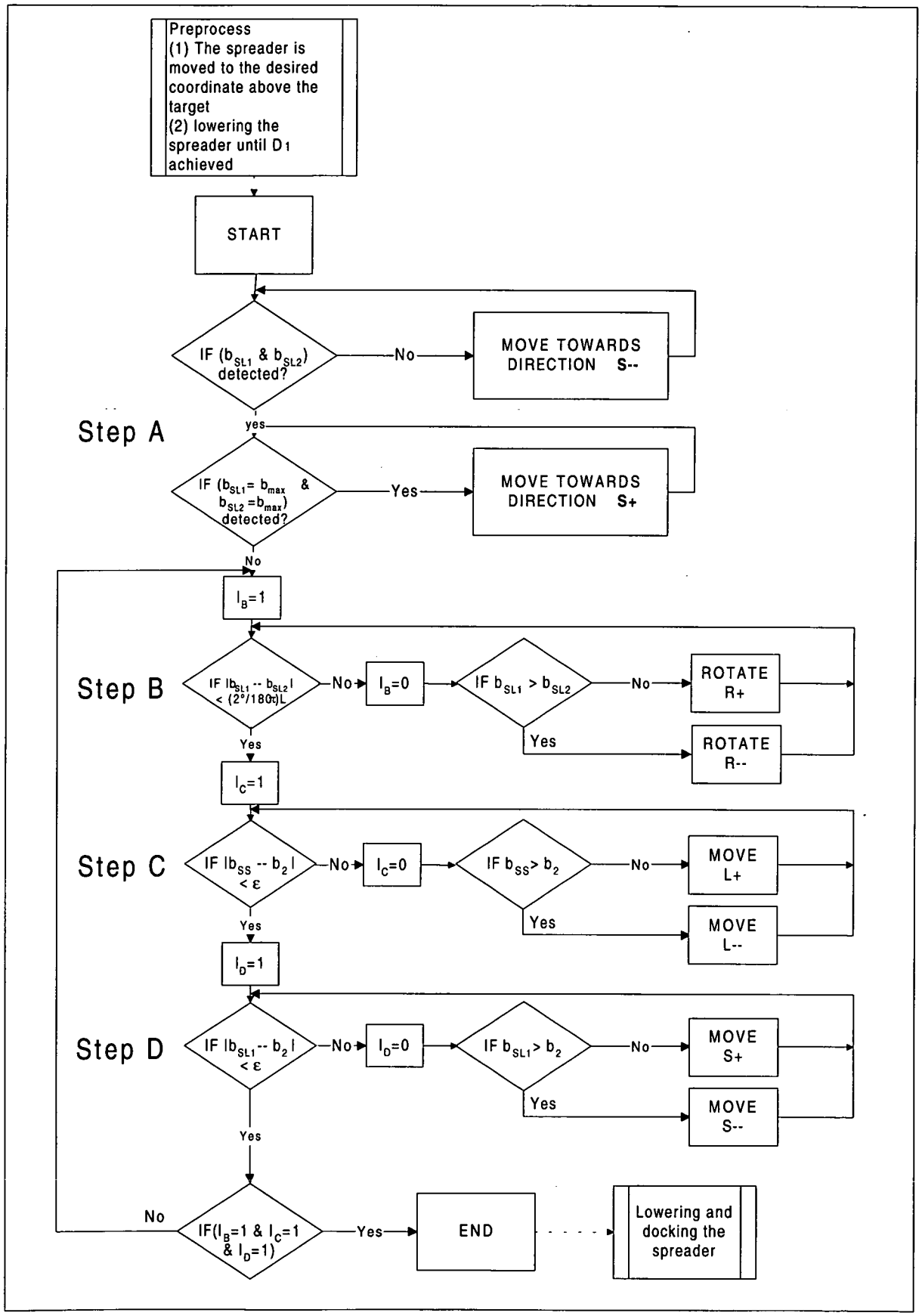


Figure 2.40 The auto-correction algorithm for line laser modules

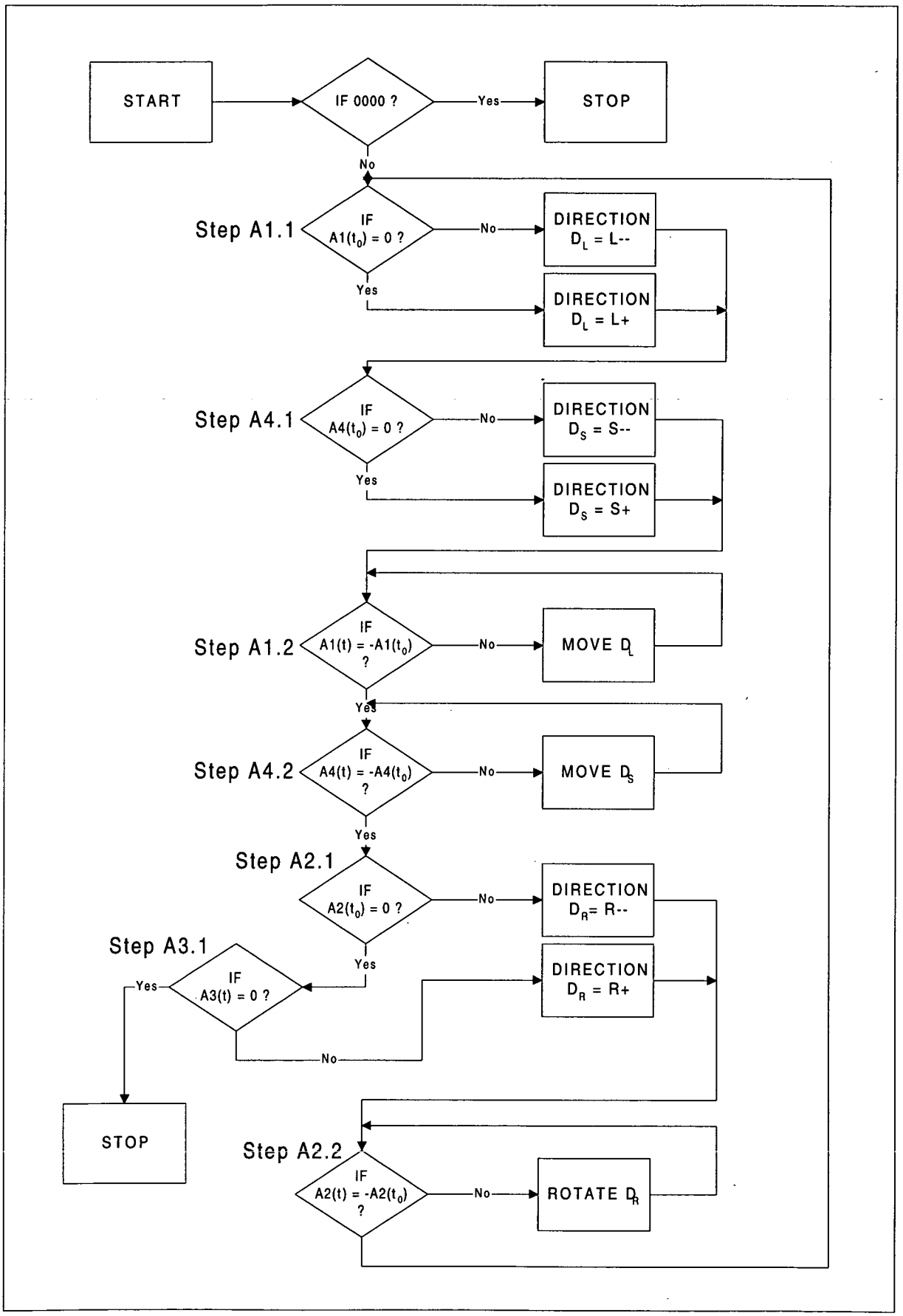


Figure 2.41 The auto-correction algorithm for spot laser modules

Chapter 3

A Dynamic Feedback Linearization Controller for Three Dimensional Control of the RTG Crane using the Single Cable Model

3.1 Introduction

An *automatic controller* is generally regarded as a mechanism that can correct its own dynamical performance to conform to some preset desired values by using information measured and fed back whilst the system operates. Since the first systematic study of feedback control stability appeared in the 19th century, many control methods have been proposed and applied in different areas. Generally speaking, control methods can be categorized into two fields : *classical control*, which uses the complex variable methods of root loci and Bode diagrams, and *modern control*, which works directly with the ordinary differential equations (ODEs) of the control system in "normal" or "state" form, and which typically calls for extensive use of computers. Due to the high inherent levels of non-linearity and the multi-input-multi-output (MIMO) conditions present in the crane system, it is considered to be more appropriate to choose *modern* procedures for automatic control of the system.

A number of previous works have addressed the problems of the dynamic modeling of cranes and different methods of control have also been provided to tackle those critical conditions in order to transport the payloads more rapidly, safely and without swing. For instances, *optimal control* of cranes has been examined by several researchers (Auernig [4], Hamalainen [35], Mason [54], Sakawa [67]). In those papers, the study of a minimum-time control strategy was the main subject, i.e. these models were based on time-optimal solutions. Simulated effects of open-loop control actions were shown. In principal, an ideal crane can start from the initial position and stop on the final position by an open-loop strategy, but the operation is sensitive to disturbances and,

moreover, the control actions are rough, giving rise to extra stresses and potential fatigue of materials. Starr (Starr [72]) proposed an open-loop control method to achieve swing-free motion of objects suspended from a path-control robot manipulator. However, the proposed model involves only swing in the plane of motion, and the controller is open-loop and depends critically on the "zero initial conditions" assumption. Moustafa et.al. (Moustafa [58]) developed a nonlinear model for a crane to allow for the study of an anti-sway control algorithm in which the *static linearized feedback control* scheme is implemented. Sakawa et.al. (Sakawa [67]) developed open-loop and closed-loop control strategies to minimize the payload swing for rotary crane systems. *Robust control*, on the other hand, was also discussed for a certain class of non-linear mechanical and distributed parameter systems whose desired state may correspond to a mechanical energy level of zero (Chung [20], Hashimoto [37]). In such systems, the control objectives are achieved only by dissipating their physical energy. The idea of energy dissipation control is formulated and asymptotically stable controls are demonstrated for a crane system and for a flexible robotic arm. The control objective of the system is to control the trolley position suppressing the load swing only by the trolley driving force. These papers proposed an energy dissipation control which is available for a class of non-linear mechanical systems and for a class of distributed parameter systems. The control law derived from the energy dissipation analysis of the system has advantages of easiness to implement and high robustness against parametric errors. However, the swing angle of the load has not been utilized for control because of physical difficulty in measuring it. In other crane control work, Butler [10] used modal decomposition to develop an adaptive controller. Martindale [50] designed an integrator backstepping controller based on an approximate crane model. In the paper presented by Burg et.al (Burg [11]), a saturation control scheme is implemented after the underactuated crane dynamics have been transformed to a form similar to the ball-and-beam dynamics. Joshi [42] considered the control of crane systems where the payload cable is flexible.

Fuzzy control is also introduced to this area (Benhidjeb [5], Cartmell [16], Kijima [45], Suzuki [74],Yamada [94]). It is a technique which realizes the skill of human operators, i.e. a nonmathematical pattern of human linguistic behaviour (Tong [78], Zadeh [97]). The control rules are obtained from skilled operators' experience and

then travelling acceleration time is calculated by considering the position of obstructions and the rope length at the maximum travelling speed. Finally, when the trolley decelerates, the travelling time of the constant speed section is adjusted by referring to the swing angle and the rate of change of swing angle at the beginning of deceleration. The other modern control method uses artificial neural networks (Javed [41]). The goal of the system proposed by this technique is to learn the operational routines of a gantry crane. Neural networks (Fukuda [33],Wang [90]), such as the multi-layered perceptron, have shown learning capability by experiencing examples of the task and then utilizing their acquired knowledge. These networks can be trained reasonably well on a simulated system environment using the intended operational routines. It is, however, perceived that some degree of on-line training and readapting facility is essential prior to the autonomous application of a neural network based controlling system. In the real world, a human operator, after extensive training and practical experience, can acquire appropriate expertise to accomplish a required task. The neural network-based controller is required to learn the whole process by observing the system parameters, e.g. load position and swing angle, in a manner that exhibits human-like expertise. The neural network-based controller can be trained initially in a simulated environment and then further refinements can be accomplished by on-line re-adaptation. The system can then improve its controlling capabilities by generalizing the acquired knowledge and producing an appropriate sequence of control signals, depending on the position error and the angle of swing.

However, a potential shortcoming of many the above approaches is that the controllers are based on *linearized* crane models, leaving unanswered questions of the effect of the resulting controller on the nonlinear crane dynamics. Moreover, the models which these authors presented were mostly based on two degrees of freedom (the actuating degree, x ,and the swing angle, θ). In a practical working environment the other two actuating degrees of freedom (transverse movement, y, and the length of the cable, l), and the polar angle projected onto the ground, α, have to be taken into account to get a 3 dimensional representation of the problem. Therefore, a more comprehensive, and generally encompassing, approach would be to directly consider all the five degrees of freedom and the system nonlinearities during the control design.

In general, the motion control problem consists of (1) obtaining dynamic models of the system, and (2) using these models to determine control laws or strategies to achieve the desired system response and performance. This chapter is principally concerned with investigating how to use modern analytical control strategies to optimize the trajectory of a moving spreader system towards the target container. In section 3.2, a 3-D single-cable-model (SCM) (Cartmell [15], Morrish [57]) of the RTG crane based on the 1/8 experimental crane rig illustrated in Chapter 2 is established, from which the five governing equations of motion are derived. Due to the highly cross-coupled nature of these equations, Feedback Linearization Control is suggested, by which the effect of the system nonlinearities can be conveniently incorporated in a reasonably computationally efficient manner (section 3.3). In section 3.4 the optimal values of the position and velocity gains in the FLC model are examined. In section 3.5, a simulation model using *SIMULINK* and *MATLAB* has been set up to evaluate the results of the chosen strategy.

3.2 System dynamics and equations of motion

In Figure. 3.1 a schematic of the simplified 3-D RTG crane model is given. The convention of the coordinate system and the notation of the variables follows preliminary work carried out by Cartmell et al [14], [15]. The multi-cable spreader, in this case, is treated as a simple-cable lumped mass, m_s . The gantry , m_c , is restricted to move only in the x - direction and the trolley, m_t , is constrained to move in the y - direction. In the other words, axis X' is parallel to axis X and axis Y' is parallel to axis Y. It should be noted that the movement of the experimental crane is slightly different from the real-size RTG crane because of the construction and setup of the experimental rig which is discussed in Chapter 4. For simulating travelling motion of the trolley, m_c and m_t move together along the x-direction. For simulating transversing motion of the gantry, only m_t moves along the y-direction. Clearly the system has five degrees of freedom, namely: traveling motion of the trolley (x), transverse motion of

the gantry (y), the length of the cable (l), sway angle of the lumped mass m_s (ψ), and the polar angle projected onto the XY-plane (α).

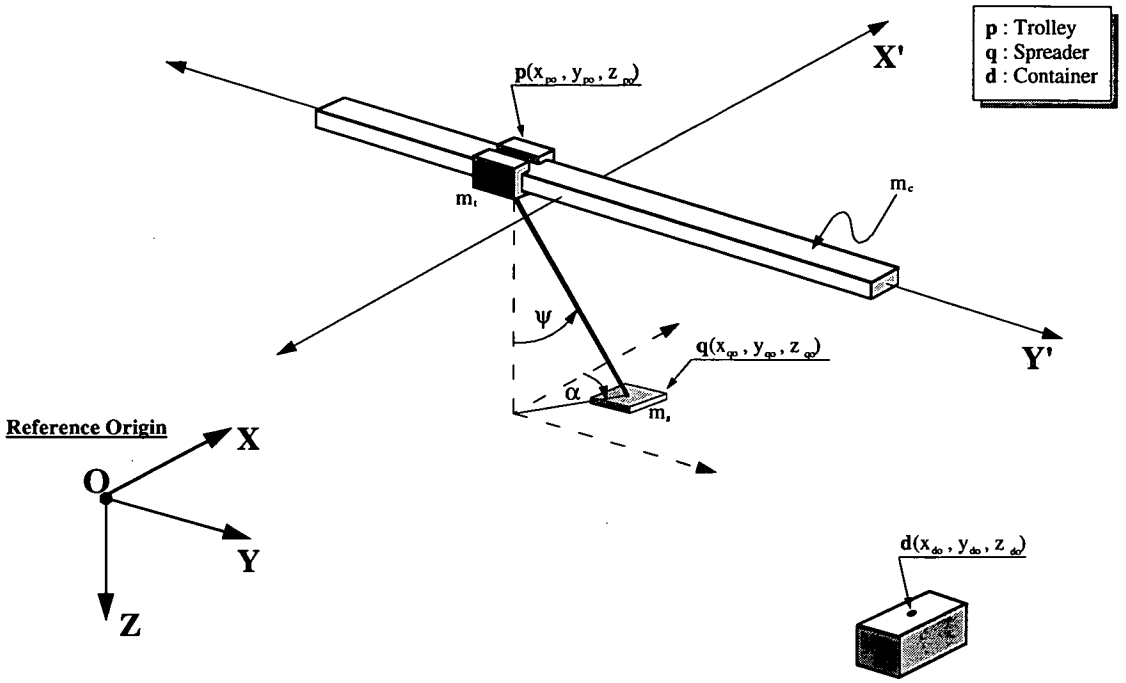


Figure 3.1 The 3-D RTG crane model

For simplicity, the following assumptions are made:

- The elasticity of the crane structure elements, dissipation effects like rolling resistance and losses in the drive mechanism, and effects such as wind forces, are neglected.
- The load is assumed to be concentrated at a point and hanging at the end of a massless cable with negligible length changes due to load swing.
- The rotation of the lumped mass m_s about its own mass centre is ignored

The Cartesians x_{po} , y_{po} , z_{po} (noting the suffix 'po' denotes p relative to O, the reference origin) are given by :

$$x_{po} = x_{pd} + x_{do} \qquad \text{-------} \quad (3.1)$$

$$y_{po} = y_{pd} + y_{do} \qquad \text{-------} \quad (3.2)$$

$$z_{po} = z_{pd} + z_{do} \qquad \text{-------} \quad (3.3)$$

leading to :

$$x_{qo} = x_{pd} + x_{do} + l \sin\psi \cos\alpha \qquad \text{-------} \quad (3.4)$$

$$y_{qo} = y_{pd} + y_{do} + l \sin\psi \sin\alpha \qquad \text{-------} \quad (3.5)$$

$$z_{qo} = z_{pd} + z_{do} + l \cos\psi \qquad \text{-------} \quad (3.6)$$

By letting $p(x_{po}, y_{po}, z_{po}) = p(x, y, z)$,

$q(x_{qo}, y_{qo}, z_{qo}) = q(x_m, y_m, z_m)$,

then

$$x_m = l \sin\psi \cos\alpha + x \qquad \text{-------} \quad (3.7)$$

$$y_m = l \sin\psi \sin\alpha + y \qquad \text{-------} \quad (3.8)$$

$$z_m = l \cos\psi \qquad \text{-------} \quad (3.9)$$

$$\dot{x}_m = \dot{l} \sin\psi \cos\alpha + l\dot{\psi} \cos\psi \cos\alpha - l\dot{\alpha} \sin\psi \sin\alpha + \dot{x} \qquad \text{-------} \quad (3.10)$$

$$\dot{y}_m = \dot{l} \sin\psi \sin\alpha + l\dot{\psi} \cos\psi \sin\alpha + l\dot{\alpha} \sin\psi \cos\alpha + \dot{y} \qquad \text{-------} \quad (3.11)$$

$$\dot{z}_m = \dot{l} \cos\psi - l\dot{\psi} \sin\psi \qquad \text{-------} \quad (3.12)$$

To obtain the kinetic energy of the spreader, the simple formula below is used,

$$(a+b+c+d)^2 = a^2 + b^2 + c^2 + d^2 + 2(ab + ac + ad + bc + bd + cd)$$

and squaring the velocities :

$$\begin{aligned} \dot{x}_m^2 = {} & \dot{l}^2 \sin^2\psi \cos^2\alpha + l^2\dot{\psi}^2 \cos^2\psi \cos^2\alpha + l^2\dot{\alpha}^2 \sin^2\psi \sin^2\alpha + \dot{x}^2 + \\ & 2[l\dot{l}\dot{\psi} \sin\psi \cos\psi \cos^2\alpha + (-l\dot{l}\dot{\alpha} \sin^2\psi \sin^2\alpha \cos\alpha) + \\ & \dot{x}\dot{l} \sin\psi \cos\alpha + (-l^2\dot{\psi}\dot{\alpha} \sin\psi \cos\psi \sin\alpha \cos\alpha) + \\ & \dot{x}l\dot{\psi} \cos\psi \cos\alpha + (-\dot{x}l\dot{\alpha} \sin\psi \sin\alpha)] \end{aligned} \qquad \text{-------} \quad (3.13)$$

$$\dot{y}_m^2 = \dot{l}^2 \sin^2\psi \sin^2\alpha + l^2\dot{\psi}^2 \cos^2\psi \sin^2\alpha + l^2\dot{\alpha}^2 \sin^2\psi \cos^2\alpha + \dot{y}^2 +$$
$$2[l\dot{l}\dot{\psi} \sin\psi \cos\psi \sin^2\alpha + l\dot{l}\dot{\alpha} \sin^2\psi \sin^2\alpha \cos\alpha +$$
$$\dot{y}\dot{l} \sin\psi \sin\alpha + l^2\dot{\psi}\dot{\alpha} \sin\psi \cos\psi \sin\alpha \cos\alpha +$$
$$\dot{y}l\dot{\psi} \cos\psi \sin\alpha + \dot{y}l\dot{\alpha} \sin\psi \cos\alpha] \quad \text{------- (3.14)}$$

$$\dot{z}_m^2 = \dot{l}^2 \cos^2\psi + l^2\dot{\psi}^2 \sin^2\psi + l^2\dot{\alpha}^2 \sin^2\psi \cos^2\alpha + (-2l\dot{l}\dot{\psi} \sin\psi \cos\psi) \text{------- (3.15)}$$

Then it follows that,

$$\dot{x}_m^2 + \dot{y}_m^2 + \dot{z}_m^2 = \dot{x}^2 + \dot{y}^2 + \dot{l}^2 +$$
$$2[\dot{x}\dot{l} \sin\psi \cos\alpha + \dot{y}\dot{l} \sin\psi \sin\alpha + \dot{x}l\dot{\psi} \cos\psi \cos\alpha +$$
$$\dot{y}l\dot{\psi} \cos\psi \sin\alpha + (-\dot{x}l\dot{\alpha} \sin\psi \sin\alpha) + \dot{y}l\dot{\alpha} \sin\psi \cos\alpha] \quad \text{-------(3.16)}$$

So the kinetic energy of the spreader will be

$$T_m = \frac{1}{2} m_s (\dot{x}_m{}^2 + \dot{y}_m{}^2 + \dot{z}_m{}^2) \quad \text{------- (3.17)}$$

Equations (3.16) and (3.17) can thus be used to formulate the system kinetic energy (see equation (3.20) below).

The equations of motion of a dynamic system can often be derived in terms of generalized coordinates by use of **Lagrange's equations**. For a n-degree of freedom dynamic system, Lagrange's equations can be stated as

$L = T - V$ L : Lagrangian,
T : Kinetic Energy,
V : Potential Energy

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}_j}\right) - \left(\frac{\partial L}{\partial q_j}\right) = Q_j^{(n)} \quad , \quad j = 1, 2, ..., n \quad \text{------- (3.18)}$$

where $\dot{q}_j = \partial q_j / \partial t$ is the generalized velocity and $Q_j^{(n)}$ is the nonconservative generalized force corresponding to the generalized coordinate q_j. For example, if F_{xk} F_{yk} , and F_{zk} represent the external forces acting on the k^{th} mass of the system in the x, y, and z directions, respectively, then the generalized force $Q_j^{(n)}$ can be computed as follows:

$$Q_j^{(n)} = \sum_k \left(F_{xk}(\frac{\partial x_k}{\partial q_j}) + F_{yk}(\frac{\partial y_k}{\partial q_j}) + F_{zk}(\frac{\partial z_k}{\partial q_j}) \right) \qquad \text{------- (3.19)}$$

where x_k , y_k , and z_k are the displacements of the k^{th} mass in the x, y, and z directions, respectively.

For the scaled-down experimental rig the whole dynamics of the system are considered as a simple pendulum swinging under a moving trolley, as shown in Figure 3.1. Therefore the kinetic and potential energies of the complete system are :

$$T = \frac{1}{2}(m_t + m_c)\dot{x}^2 + \frac{1}{2}m_t\dot{y}^2 + \frac{1}{2}m_s(\dot{x}_m{}^2 + \dot{y}_m{}^2 + \dot{z}_m{}^2) \qquad \text{------- (3.20)}$$

$$V = m_s l(1-\cos\psi)g \qquad \text{------- (3.21)}$$

With recourse to the Lagrangian $L = T - V$ and Lagrange's equation (3.18), the five governing equation are derived as follows:

1) For the x - coordinate (traveling motion of the trolley, m_t moving together with m_c)

$$\left(\frac{\partial L}{\partial \dot{x}}\right) = (m_t + m_c)\dot{x} + \frac{1}{2} m_s (2\dot{x} + 2\dot{l}\sin\psi\cos\alpha + 2l\dot{\psi}\cos\psi\cos\alpha - 2l\dot{\alpha}\sin\psi\sin\alpha)$$

------- (3.22)

$\Rightarrow$

$$\begin{aligned}\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{x}}\right) = {} & (m_t + m_c)\ddot{x} + \\ & m_s[\ddot{x} + (\ddot{l}\sin\psi\cos\alpha + \dot{l}\dot{\psi}\cos\psi\cos\alpha - \dot{l}\dot{\alpha}\sin\psi\sin\alpha) + \\ & (\dot{l}\dot{\psi}\cos\psi\cos\alpha + l\ddot{\psi}\cos\psi\cos\alpha - l\dot{\psi}^2\sin\psi\cos\alpha - l\dot{\psi}\dot{\alpha}\cos\psi\sin\alpha) + \\ & (-\dot{l}\dot{\alpha}\sin\psi\sin\alpha - l\ddot{\alpha}\sin\psi\sin\alpha - l\dot{\psi}\dot{\alpha}\cos\psi\sin\alpha - l\dot{\alpha}^2\sin\psi\cos\alpha)] \\ = {} & (m_t + m_c + m_s)\ddot{x} + \\ & m_s[\ddot{l}\sin\psi\cos\alpha + l\ddot{\psi}\cos\psi\cos\alpha\dot{l} - l\ddot{\alpha}\sin\psi\sin\alpha - l\dot{\psi}^2\sin\psi\cos\alpha \\ & -l\dot{\alpha}^2\sin\psi\cos\alpha + 2\dot{l}\dot{\psi}\cos\psi\cos\alpha - 2\dot{l}\dot{\alpha}\sin\psi\sin\alpha - 2l\dot{\alpha}\dot{\psi}\cos\psi\sin\alpha]\end{aligned} \quad (3.23)$$

$$\left(\frac{\partial L}{\partial x}\right) = 0 \qquad \text{------- (3.24)}$$

If the nonconservative external force acting on the x - coordinate is F_x , the equation of motion is,

$$\left(1 + \frac{m_t + m_c}{m_s}\right)\ddot{x} + (\ddot{l}\sin\psi\cos\alpha + l\ddot{\psi}\cos\psi\cos\alpha - l\ddot{\alpha}\sin\psi\sin\alpha - l\dot{\psi}^2\sin\psi\cos\alpha$$

$$-l\dot{\alpha}^2\sin\psi\cos\alpha + 2\dot{l}\dot{\psi}\cos\psi\cos\alpha - 2\dot{l}\dot{\alpha}\sin\psi\sin\alpha - 2l\dot{\alpha}\dot{\psi}\cos\psi\sin\alpha) = \frac{F_x}{m_s}$$

------- (3.25)

2) For the y - coordinate (transversing motion of the gantry, only m_t moving along in the y-direction),

$$\left(\frac{\partial L}{\partial \dot{y}}\right) = m_t \dot{y} + \frac{1}{2} m_s (2\dot{y} + 2\dot{l}\sin\psi\sin\alpha + 2l\dot{\psi}\cos\psi\sin\alpha + 2l\dot{\alpha}\sin\psi\cos\alpha) \quad (3.26)$$

$$\begin{aligned}\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{y}}\right) = {} & m_t \ddot{y} + \\ & m_s[\ddot{y} + (\ddot{l}\sin\psi\sin\alpha + \dot{l}\dot{\psi}\cos\psi\sin\alpha + \dot{l}\dot{\alpha}\sin\psi\cos\alpha) + \\ & (\dot{l}\dot{\psi}\cos\psi\sin\alpha + l\ddot{\psi}\cos\psi\sin\alpha - l\dot{\psi}^2\sin\psi\sin\alpha + l\dot{\psi}\dot{\alpha}\cos\psi\cos\alpha) + \\ & (\dot{l}\dot{\alpha}\sin\psi\cos\alpha + l\ddot{\alpha}\sin\psi\cos\alpha + l\dot{\psi}\dot{\alpha}\cos\psi\cos\alpha - l\dot{\alpha}^2\sin\psi\sin\alpha)] \quad (3.27) \\ = {} & (m_t + m_s)\ddot{y} + \\ & m_s(\ddot{l}\sin\psi\sin\alpha + l\ddot{\psi}\cos\psi\sin\alpha + l\ddot{\alpha}\sin\psi\cos\alpha - l\dot{\psi}^2\sin\psi\sin\alpha \\ & -l\dot{\alpha}^2\sin\psi\sin\alpha + 2\dot{l}\dot{\psi}\cos\psi\sin\alpha + 2\dot{l}\dot{\alpha}\sin\psi\cos\alpha + 2l\dot{\psi}\dot{\alpha}\cos\psi\cos\alpha)\end{aligned}$$

$$\left(\frac{\partial L}{\partial y}\right) = 0 \quad \text{-------} \quad (3.28)$$

If the nonconservative external force acting on the y - coordinate is F_y, the equation of motion is,

$$\begin{aligned}&\left(1 + \frac{m_t}{m_s}\right)\ddot{y} + (\ddot{l}\sin\psi\sin\alpha + l\ddot{\psi}\cos\psi\sin\alpha + l\ddot{\alpha}\sin\psi\cos\alpha - l\dot{\psi}^2\sin\psi\sin\alpha \\ &-l\dot{\alpha}^2\sin\psi\sin\alpha + 2\dot{l}\dot{\psi}\cos\psi\sin\alpha + 2\dot{l}\dot{\alpha}\sin\psi\cos\alpha + 2l\dot{\psi}\dot{\alpha}\cos\psi\cos\alpha) = \frac{F_y}{m_s}\end{aligned} \quad (3.29)$$

3) For the l - coordinate (the length of the cable)

$$\left(\frac{\partial L}{\partial \dot{l}}\right) = \frac{1}{2} m_s [2\dot{l}\sin^2\psi\cos^2\alpha +$$
$$2(l\dot{\psi}\sin\psi\cos\psi\cos^2\alpha - l\dot{\alpha}\sin^2\psi\sin\alpha\cos\alpha + \dot{x}\sin\psi\cos\alpha) + 2\dot{l}\sin^2\psi\sin^2\alpha +$$
$$2(l\dot{\psi}\sin\psi\cos\psi\sin^2\alpha + l\dot{\alpha}\sin^2\psi\sin\alpha\cos\alpha + \dot{y}\sin\psi\sin\alpha) + 2\dot{l}\cos^2\psi$$
$$-2l\dot{\psi}\sin\psi\cos\psi]$$
$$= m_s(\dot{l} + \dot{x}\sin\psi\cos\alpha + \dot{y}\sin\psi\sin\alpha)$$

------- (3.30)

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{l}}\right) = m_s[\ddot{l} + (\ddot{x}\sin\psi\cos\alpha + \dot{x}\dot{\psi}\cos\psi\cos\alpha - \dot{x}\dot{\alpha}\sin\psi\sin\alpha) +$$
$$(\ddot{y}\sin\psi\sin\alpha + \dot{y}\dot{\psi}\cos\psi\sin\alpha + \dot{y}\dot{\alpha}\sin\psi\cos\alpha)] \quad \text{------- (3.31)}$$

$$\left(\frac{\partial L}{\partial l}\right) = \frac{1}{2} m_s [2l\dot{\psi}^2\cos^2\psi\cos^2\alpha + 2l\dot{\alpha}^2\sin^2\psi\cos^2\alpha +$$
$$2(\dot{l}\dot{\psi}\sin\psi\cos\psi\cos^2\alpha - \dot{l}\dot{\alpha}\sin^2\psi\sin\alpha\cos\alpha - 2l\dot{\psi}\dot{\alpha}\sin\psi\cos\psi\sin\alpha\cos\alpha +$$
$$\dot{x}\dot{\psi}\cos\psi\cos\alpha - \dot{x}\dot{\alpha}\sin\psi\sin\alpha) + 2l\dot{\psi}^2\cos^2\psi\sin^2\alpha + 2l\dot{\alpha}^2\sin^2\psi\cos^2\alpha +$$
$$2(\dot{l}\dot{\psi}\sin\psi\cos\psi\sin^2\alpha + \dot{l}\dot{\alpha}\sin^2\psi\sin\alpha\cos\alpha - 2l\dot{\psi}\dot{\alpha}\sin\psi\cos\psi\sin\alpha\cos\alpha +$$
$$\dot{y}\dot{\psi}\cos\psi\sin\alpha + \dot{y}\dot{\alpha}\sin\psi\cos\alpha) + 2l\dot{\psi}^2\sin^2\psi\sin^2\alpha - 2l\dot{\psi}\sin\psi\cos\alpha] +$$
$$[-m_s(1-\cos\psi)]g$$
$$= m_s[l\dot{\psi}^2 + l\dot{\alpha}^2\sin^2\psi + \dot{x}\dot{\psi}\cos\psi\cos\alpha + \dot{y}\dot{\psi}\cos\psi\sin\alpha - \dot{x}\dot{\alpha}\sin\psi\sin\alpha +$$
$$\dot{y}\dot{\alpha}\sin\psi\cos\alpha] + [-m_s(1-\cos\psi)]g$$

------- (3.32)

If the nonconservative external force acting on the l - coordinate is F_l, the equation of motion is,

$$\ddot{l} + \ddot{x}\sin\psi\cos\alpha + \ddot{y}\sin\psi\sin\alpha - l\dot{\psi}^2 - l\dot{\alpha}^2\sin^2\psi + g(1-\cos\psi) = \frac{F_l}{m_s} \qquad (3.33)$$

4) For the ψ - coordinate (the swing angle of the lumped mass m_s)

$$\left(\frac{\partial L}{\partial \dot{\psi}}\right) = \frac{1}{2} m_s [2l^2\dot{\psi} + 2(\dot{x}l\cos\psi\cos\alpha + \dot{y}l\cos\psi\sin\alpha)] \qquad \text{-------} \quad (3.34)$$

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\psi}}\right) = \frac{1}{2} m_s [4\dot{l}l\dot{\psi} + 2l^2\ddot{\psi} + 2(\ddot{x}l\cos\psi\cos\alpha + \dot{x}\dot{l}\cos\psi\cos\alpha - \dot{x}l\dot{\psi}\sin\psi\cos\alpha - \dot{x}l\dot{\alpha}\cos\psi\sin\alpha) + 2(\ddot{y}l\cos\psi\sin\alpha + \dot{y}\dot{l}\cos\psi\sin\alpha - \dot{y}l\dot{\psi}\sin\psi\sin\alpha + \dot{y}l\dot{\alpha}\cos\psi\cos\alpha)] \quad (3.35)$$

$$\left(\frac{\partial L}{\partial \psi}\right) = \frac{1}{2} m_s [2l^2\dot{\alpha}^2\sin\psi\cos\alpha + 2(\dot{x}\dot{l}\cos\psi\cos\alpha + \dot{y}\dot{l}\cos\psi\sin\alpha - \dot{x}l\dot{\psi}\sin\psi\cos\alpha - \dot{y}l\dot{\psi}\sin\psi\sin\alpha - \dot{x}l\dot{\alpha}\cos\psi\sin\alpha + \dot{y}l\dot{\alpha}\cos\psi\cos\alpha)] - m_s l\, g\sin\psi$$

------- (3.36)

Because there is no actuating force on the ψ - coordinate , $F_\psi = 0$, leading to the following equation of motion,

$$l^2\ddot{\psi} + l\ddot{x}\cos\psi\cos\alpha + l\ddot{y}\cos\psi\sin\alpha + 2\dot{l}l\dot{\psi} - l^2\dot{\alpha}^2\sin\psi\cos\alpha + l\, g\sin\psi = 0 \qquad (3.37)$$

or,

$$l\ddot{\psi} + \ddot{x}\cos\psi\cos\alpha + \ddot{y}\cos\psi\sin\alpha + 2\dot{l}\dot{\psi} - l\dot{\alpha}^2\sin\psi\cos\alpha + g\sin\psi = 0 \qquad (3.38)$$

5) For the α - coordinate (the polar angle projected onto the XY-plane)

$$\left(\frac{\partial L}{\partial \dot{\alpha}}\right) = \frac{1}{2} m_s [2l^2\dot{\alpha}\sin^2\psi - 2(\dot{x}l\sin\psi\sin\alpha + \dot{y}l\sin\psi\cos\alpha)] \qquad \text{-------} \quad (3.39)$$

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\alpha}}\right) = \frac{1}{2}m_s[(2l^2\ddot{\alpha}\sin^2\psi + 4\dot{l}l\dot{\alpha}\sin^2\psi + 4l^2\psi\dot{\alpha}\sin\psi\cos\psi) +$$
$$2(-\ddot{x}l\sin\psi\sin\alpha - \dot{x}\dot{l}\sin\psi\sin\alpha - \dot{x}l\dot{\psi}\cos\psi\sin\alpha - \dot{x}l\dot{\alpha}\sin\psi\cos\alpha) + \quad (3.40)$$
$$2(\ddot{y}l\sin\psi\cos\alpha + \dot{y}\dot{l}\sin\psi\cos\alpha + \dot{y}l\dot{\psi}\cos\psi\cos\alpha - \dot{y}l\dot{\alpha}\sin\psi\sin\alpha)]$$

$$\left(\frac{\partial L}{\partial \alpha}\right) = \frac{1}{2}m_s[2(-\dot{x}\dot{l}\cos\psi\sin\alpha + \dot{y}\dot{l}\sin\psi\cos\alpha - \dot{x}l\dot{\psi}\cos\psi\sin\alpha \quad \text{-------} \quad (3.41)$$
$$+\dot{y}l\dot{\psi}\cos\psi\cos\alpha - \dot{x}l\dot{\alpha}\sin\psi\cos\alpha - \dot{y}l\dot{\alpha}\sin\psi\sin\alpha)]$$

Because there is no actuating force on the α- coordinate , $F_\alpha = 0$, and the equation of motion is,

$$(2l^2\sin^2\psi)\ddot{\alpha} + (-2l\sin\psi\sin\alpha)\ddot{x} + (2l\sin\psi\cos\alpha)\ddot{y} + 4\dot{\alpha}\dot{l}\sin^2\psi + 4\dot{\alpha}l^2\dot{\psi}\sin\psi\cos\psi = 0$$

------- (3.42)

hence,

$$(l\sin\psi)^2\ddot{\alpha} + (-l\sin\psi\sin\alpha)\ddot{x} + (l\sin\psi\cos\alpha)\ddot{y} + 2\dot{\alpha}\dot{l}\sin^2\psi + 2\dot{\alpha}l^2\dot{\psi}\sin\psi\cos\psi = 0$$

------- (3.43)

Equations (3.25), (3.29), (3.33), (3.38), (3.43) are the governing equations of the simplified crane system, to which the chosen control strategy can be applied.

3.3 Feedback Linearization Control (FLC)

As shown in the previous section, the governing equations (3.25), (3.29), (3.33), (3.38), (3.43), are highly nonlinear and cross-coupled and so the *feedback linearization* or *computed torque* (Paul [61]) control strategy has been chosen for its ability to handle such nonlinearities. Generally speaking, this approach involves generating a torque control vector which is a function of desired and actual states

(Charlet [19], Craig [21][22], Spong [71]). The governing equations of system dynamics are written in matrix form as below (Cartmell [16]) :

$$[D]\underset{\sim}{\ddot{q}} + C(\underset{\sim}{q},\underset{\sim}{\dot{q}})\underset{\sim}{\dot{q}} + f(\underset{\sim}{q}) = \underset{\sim}{\tau} \qquad \text{------- (3.44)}$$

where $\underset{\sim}{q}$ is the column vector of the generalized coordinates, $[D]$ is a configuration dependent inertia matrix, $[C(\underset{\sim}{q},\underset{\sim}{\dot{q}})]$ contains centripetal Coriolis accelerations and $f(\underset{\sim}{q})$ is a potential energy term, in this case due to gravity. $[D]$ is positive definite or invertible and symmetric. $C(\underset{\sim}{q},\underset{\sim}{\dot{q}})$, on the other hand, is not unique and is only decomposed from $\underset{\sim}{\dot{q}}$ for computational purposes at a later stage.

The governing equations (3.25), (3.29), (3.33), (3.38), (3.43), derived earlier in section 3.2 can be rearranged in the matrix form of (3.44), such that :

$$\underset{\sim}{q} = \begin{bmatrix} x \\ y \\ l \\ \psi \\ \alpha \end{bmatrix}, \quad \underset{\sim}{\dot{q}} = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{l} \\ \dot{\psi} \\ \dot{\alpha} \end{bmatrix}, \quad \underset{\sim}{\ddot{q}} = \begin{bmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{l} \\ \ddot{\psi} \\ \ddot{\alpha} \end{bmatrix},$$

$$[D] = \begin{bmatrix} 1+\dfrac{m_t+m_c}{m_s} & 0 & \sin\psi\cos\alpha & l\cos\psi\cos\alpha & -l\sin\psi\sin\alpha \\ 0 & 1+\dfrac{m_t}{m_s} & \sin\psi\sin\alpha & l\cos\psi\sin\alpha & l\sin\psi\cos\alpha \\ \sin\psi\cos\alpha & \sin\psi\sin\alpha & 1 & 0 & 0 \\ l\cos\psi\cos\alpha & l\cos\psi\sin\alpha & 0 & l^2 & 0 \\ -l\sin\psi\sin\alpha & l\sin\psi\cos\alpha & 0 & 0 & (l\sin\psi)^2 \end{bmatrix}$$

------- (3.45)

$$C(\underset{\sim}{q},\dot{\underset{\sim}{q}}) =$$

$$\begin{bmatrix} 0 & 0 & 2\dot{\psi}\cos\psi\cos\alpha - 2\dot{\alpha}\sin\psi\sin\alpha & -l\dot{\psi}\sin\psi\cos\alpha & -l\dot{\alpha}\sin\psi\cos\alpha - 2l\dot{\psi}\cos\psi\sin\alpha \\ 0 & 0 & 2\dot{\psi}\cos\psi\sin\alpha + 2\dot{\alpha}\sin\psi\cos\alpha & -l\dot{\psi}\sin\psi\sin\alpha & -l\dot{\alpha}\sin\psi\sin\alpha + 2l\dot{\psi}\cos\psi\cos\alpha \\ 0 & 0 & 0 & -l\dot{\psi} & -l\dot{\alpha}\sin^2\psi \\ 0 & 0 & 2l\dot{\psi} & 0 & -l^2\dot{\alpha}\sin\psi\cos\psi \\ 0 & 0 & 0 & 0 & l\sin\psi(2\dot{l}\sin\psi + 2l\dot{\psi}\cos\psi) \end{bmatrix}$$

------- (3.46)

$$f(\underset{\sim}{q}) = \begin{bmatrix} 0 \\ 0 \\ g(1-\cos\psi) \\ gl\sin\psi \\ 0 \end{bmatrix}$$

------- (3.47)

$$\underset{\sim}{\tau} = \begin{bmatrix} \frac{F_x}{m_s} \\ \frac{F_y}{m_s} \\ \frac{F_l}{m_s} \\ 0 \\ 0 \end{bmatrix}$$

------- (3.48)

The idea of feedback linearization is to seek out a nonlinear feedback control law. In general,

$$\underset{\sim}{\tau} = g(\underset{\sim}{q},\dot{\underset{\sim}{q}})$$

------- (3.49)

which, as stated in Craig [21][22], when substituted into equation (3.44) will result in a linear closed loop system. At first glance, this might seem to be quite difficult. However, the control torque τ can be chosen according to the equation

$$\underset{\sim}{\tau} = [D]\underset{\sim}{V} + C(\underset{\sim}{q}, \underset{\sim}{\dot{q}})\underset{\sim}{\dot{q}} + f(\underset{\sim}{q}) \qquad \text{-------} \quad (3.50)$$

Substitution of equation (3.50) into equation (3.44), leads to,

$$[D]\underset{\sim}{\ddot{q}} = [D]\underset{\sim}{V} \qquad \text{-------} \quad (3.51)$$

and since the inertia matrix $[D]$ is reasonably assumed to be invertible, the previous equation can be further reduced to,

$$\underset{\sim}{\ddot{q}} = \underset{\sim}{V} \qquad \text{-------} \quad (3.52)$$

Equation (3.52) is known as a *double integrator system* (Spong [71]) as it represents n *uncoupled* integrators, which will be equal in number to the number of generalized coordinates. The nonlinear control law in equation (3.50) is called a *feedback linearization control* (FLC). In other words, this 'new' system in equation (3.52) can be considered as a linear system with a controller $\underset{\sim}{V}$. $\underset{\sim}{V}$ is assumed to be of the following form,

$$\underset{\sim}{V} = -\{h\}\underset{\sim}{\dot{q}} - \{g\}\underset{\sim}{q} + r \qquad \text{-------} \quad (3.53)$$

where the $\{h\}$ and $\{g\}$ are diagonal matrices for position and velocity gains, which can be stated as

$$\{h\} = \begin{bmatrix} h_x & 0 & 0 & 0 & 0 \\ 0 & h_y & 0 & 0 & 0 \\ 0 & 0 & h_l & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \qquad \text{------- (3.54)}$$

$$\{g\} = \begin{bmatrix} g_x & 0 & 0 & 0 & 0 \\ 0 & g_y & 0 & 0 & 0 \\ 0 & 0 & g_l & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \qquad \text{------- (3.55)}$$

On substituting (3.52) into (3.51), the following results,

$$\ddot{\underset{\sim}{q}} + \{h\}\dot{\underset{\sim}{q}} + \{g\}\underset{\sim}{q} = r \qquad \text{------- (3.56)}$$

Now, given a *desired trajectory* for the target container

$$\underset{\sim}{q_d} = \begin{bmatrix} x_d \\ y_d \\ l_d \\ \psi_d \\ \alpha_d \end{bmatrix}, \quad \dot{\underset{\sim}{q}}_d = \begin{bmatrix} \dot{x}_d \\ \dot{y}_d \\ \dot{l}_d \\ \dot{\psi}_d \\ \dot{\alpha}_d \end{bmatrix}, \quad \ddot{\underset{\sim}{q}}_d = \begin{bmatrix} \ddot{x}_d \\ \ddot{y}_d \\ \ddot{l}_d \\ \ddot{\psi}_d \\ \ddot{\alpha}_d \end{bmatrix},$$

and setting

$$r = \ddot{\underset{\sim}{q}}_d + \{h\}\dot{\underset{\sim}{q}}_d + \{g\}\underset{\sim}{q_d} \qquad \text{------- (3. 57)}$$

and then letting the tracking error be,

$$\underset{\sim}{\varepsilon} = \underset{\sim}{q}_d - \underset{\sim}{q} \qquad \text{-------} \quad (3.58)$$

It can then be easily shown that

$$(\ddot{\underset{\sim}{q}}_d - \ddot{\underset{\sim}{q}}) + \{h\}(\dot{\underset{\sim}{q}}_d - \dot{\underset{\sim}{q}}) + \{g\}(\underset{\sim}{q}_d - \underset{\sim}{q}) = \ddot{\underset{\sim}{\varepsilon}} + \{h\}\,\dot{\underset{\sim}{\varepsilon}} + \{g\}\,\underset{\sim}{\varepsilon} = 0 \qquad \text{-------} \quad (3.59)$$

where equation (3.59) denotes a set of *globally decoupled*, linear, second order, homogeneous, ordinary differential equations in the error ε, in which PD (Proportional & Derivative) control strategies can be applied.

One thing to be aware of is that the system is only partially actuated; that is equation (3.38) in coordinate ψ and equation (3.43) in coordinate α are both not directly excited. Therefore, all the behaviour of the spreader is dependent on choosing the appropriate gains $\{h_x , h_y , h_l\}$ and $\{g_x , g_y , g_l\}$ because in the physical context there is no direct actuation capability for coordinates ψ and α.

Figure 3.2 illustrates the notion of *inner-operation/outer-operation control* for feedback linearization. By this the computation of the nonlinear control in equation (3.50) can be performed as an inner operation, perhaps with a dedicated hardware interface (Khatib [44]), with the vectors $\underset{\sim}{q}, \dot{\underset{\sim}{q}}, \underset{\sim}{V}$ as its inputs and $\underset{\sim}{\tau}$ as output, equation (3.50). The outer operation in the system is then the computation of the additional input term $\underset{\sim}{V}$. Note that $\underset{\sim}{V}$ in the outer operation is more in line with the notion of a feedback control in the usual sense of being error driven as shown in equations (3.53)~(3.59). The design of feedback control in the outer operation is (in theory) greatly simplified since it is designed for the plant represented by the dotted lines in Figure 3.2, which now appears to be a linear system.

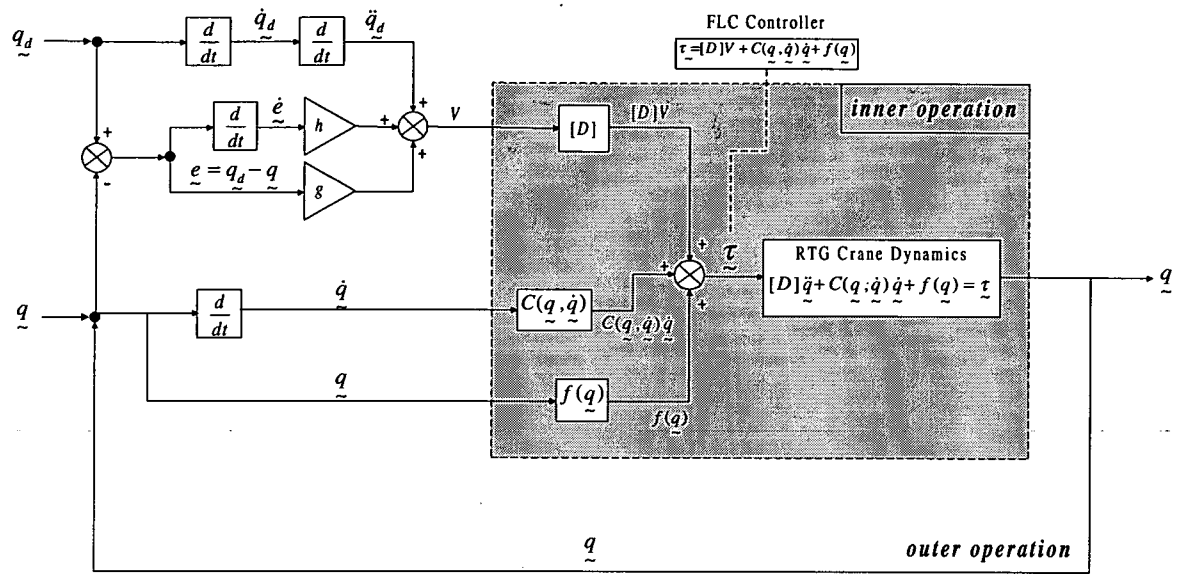


Figure 3.2 Feedback linearization control

3.4 The determination of optimal values for the gains {h} and {g}

From the previous section it was established that

$$\underset{\sim}{\varepsilon} = \underset{\sim}{q}_d - \underset{\sim}{q} \qquad \text{-------} \quad (3.58)$$

$$(\ddot{\underset{\sim}{q}}_d - \ddot{\underset{\sim}{q}}) + \{h\}(\dot{\underset{\sim}{q}}_d - \dot{\underset{\sim}{q}}) + \{g\}(\underset{\sim}{q}_d - \underset{\sim}{q}) = \ddot{\underset{\sim}{\varepsilon}} + \{h\}\,\dot{\underset{\sim}{\varepsilon}} + \{g\}\,\underset{\sim}{\varepsilon} = 0 \qquad \text{-------} \quad (3.59)$$

Equation (3.59) is a second order homogeneous differential equation, so by letting the initial condition

$$\underset{\sim}{\varepsilon}(0) = \begin{bmatrix} \varepsilon_{x0} \\ \varepsilon_{y0} \\ \varepsilon_{l0} \\ \varepsilon_{\psi 0} \\ \varepsilon_{\alpha 0} \end{bmatrix} \qquad \text{-------} \quad (3.60)$$

where, under normal circumstances, $\varepsilon_{\psi 0} = \varepsilon_{\alpha 0} = 0$, as the crane starts off from and finishes in a stationary position.

By choosing the error in the x- coordinate as an example, then from equations (3.54), (3.55) and (3.59), the specific error equation is,

$$\ddot{\varepsilon}_x + h_x \dot{\varepsilon}_x + g_x \varepsilon_x = 0 \qquad \text{-------} \quad (3.61)$$

where

$$\varepsilon_x = x_d - x \qquad \text{-------} \quad (3.62)$$

If equation (3.61) deals with data representative of the *critical damping* condition, then the analytic solution would be,

$$\varepsilon_x = (c_1 + c_2 t)\, e^{-at} \qquad \text{-------} \quad (3.63)$$

where c_1 , c_2 are constants, and $h_x^{\,2} - 4g_x = 0$, for critical damping, and,

$$a = \frac{h_x}{2} \qquad \text{-------} \quad (3.64)$$

$$g_x = \frac{h_x^2}{4} \qquad \text{-------} \quad (3.65)$$

From equation (3.60), $\varepsilon_x(0) = \varepsilon_{x0}$ and the difference (or error) in the initial velocities is set to $\dot{e}_x(0) = 0$

From equation (3.63) it is clear that,

$$\dot{\varepsilon}_x = c_2 e^{-at} - a(c_1 + c_2 t) e^{-at} \qquad \text{------- (3.66)}$$

with the specified initial condition,

$$\Rightarrow \qquad c_1 = \varepsilon_{x0}\ , \qquad c_2 = a\varepsilon_{x0}$$

$$\Rightarrow \qquad \varepsilon_x = \varepsilon_{x0}(1 + at)e^{-at} \qquad \text{------- (3.67)}$$

$$\Rightarrow \qquad \dot{\varepsilon}_x = -a^2 \varepsilon_{x0} t e^{-at} \qquad \text{------- (3.68)}$$

$$\Rightarrow \qquad \ddot{\varepsilon}_x = -a^2 \varepsilon_{x0}(1 - at)e^{-at} \qquad \text{------- (3.69)}$$

The physical limits of the velocities of the actuators are introduced as,

$$\underset{\sim}{\dot{q}}_{\max} = \begin{bmatrix} \dot{x}_{\max} \\ \dot{y}_{\max} \\ \dot{l}_{\max} \\ \dot{\psi}_{\max} \\ \dot{\alpha}_{\max} \end{bmatrix} \qquad \text{------- (3.70)}$$

Here, $\dot{\psi}_{\max}$, $\dot{\alpha}_{\max}$ will be ignored because physical limits for indirectly actuated coordinates are clearly not meaningful.

If the error in x is taken as an example again, then from equation (3.69), when $\ddot{\varepsilon}_x = 0$, there is a maximum velocity occurring, thus,

$$0 = \ddot{\varepsilon}_x = -a^2 \varepsilon_{x0}(1-at)e^{-at} \quad \Rightarrow \quad 0 = (1-at)e^{-at}$$

and because $e^{-at} \neq 0$ this leads to

$$t = \frac{1}{a} \qquad \text{------- (3.71)}$$

On substituting equation (3.69) into equation (3.68), it is shown that,

$$\dot{\varepsilon}_x(t = \frac{1}{a}) = -a\varepsilon_{x0}e^{-1} \qquad \text{------- (3.72)}$$

For the RTG crane it can be assumed that $\dot{\underset{\sim}{q}}_d = \ddot{\underset{\sim}{q}}_d = 0$, implying that the target container is located at a stationary position, which is generally true. From this it is ascertained that the maximum velocity of the controller must NOT exceed the physical limit of the actuators, thus,

$$\left|\dot{\varepsilon}_x(t = \frac{1}{a})\right| \leq \dot{x}_{\max} \qquad \text{------- (3.73)}$$

$$a_x|\varepsilon_{x0}|e^{-1} \leq \dot{x}_{\max} \qquad \text{------- (3.74)}$$

And so by applying equations (3.64) and (3.65), the gains can be stated in terms of the maximum velocity in x.

$$h_x \leq 2\frac{e \cdot \dot{x}_{\max}}{|\varepsilon_{x0}|} \qquad \text{------- (3.75)}$$

$$g_x \leq \left(\frac{e \cdot \dot{x}_{\max}}{|\varepsilon_{x0}|}\right)^2 \qquad \text{------- (3.76)}$$

The same procedure can also be applied to the other directly actuated coordinates, y and l, therefore,

$$h_y \le 2\frac{e \cdot \dot{y}_{\max}}{|\varepsilon_{y0}|} \qquad \text{-------} \quad (3.77)$$

$$g_y \le \left(\frac{e \cdot \dot{y}_{\max}}{|\varepsilon_{y0}|}\right)^2 \qquad \text{-------} \quad (3.78)$$

$$h_l \le 2\frac{e \cdot \dot{l}_{\max}}{|\varepsilon_{l0}|} \qquad \text{-------} \quad (3.79)$$

$$g_l \le \left(\frac{e \cdot \dot{l}_{\max}}{|\varepsilon_{l0}|}\right)^2 \qquad \text{-------} \quad (3.80)$$

From equations (3.75)~(3.80), the optimal values of $\{h_x , h_y , h_l\}$ and $\{g_x , g_y , g_l\}$, should be as close to the values commensurate with the desirable condition of critical damping (in this case, *active* critical damping) in the right-hand side of these equations as possible to achieve the shortest settling time. However, in order to determine the values of $\{h_x , h_y , h_l\}$ and $\{g_x , g_y , g_l\}$, the denominators $\{\varepsilon_{x0} , \varepsilon_{y0} , \varepsilon_{l0}\}$ in the right-hand side of the equations (3.75)~(3.80) have to be chosen beforehand. It is thought to be reasonable to choose the maximum physical spans $\{\varepsilon_{x0,\max} , \varepsilon_{y0,\max} , \varepsilon_{l0,\max}\}$ in the x-, y-, l- coordinates, respectively. If this is not the case, (that is, the gains are determined by values other than the physical maximum spans,) then the corresponding maximum velocities will exceed the physical maximum velocities of the actuators. The proof for this is shown in Appendix B.

3.5 Simulation of the Feedback Linearization Controller using *SIMULINK*

A *block diagram* is used to identify the major computational components in any intended control system. *SIMULINK* (Leonard [47], Mathworks [83]), which is a toolbox of the mathematically-oriented package *MATLAB* (Biran [6], Mathworks [82]), is based on this convention and uses the metaphor of a block diagram to reduce the complexity of the human-computer interface. The response of the system can then be simulated and observed using different numerical methods and initial conditions.

From the derivation of the FLC model in the previous section, a block diagram using *SIMULINK*, as shown in Figure 3.3 is proposed. The vectorization facility is exploited to reduce the complexity of the entire system. In Figure 3.3 , the thick lines denote vectors, representing several multiplexed scalar signal simultaneously, and the thin lines denote one scalar element. This proves to be rather important for further development with the *Real-Time Workshop* [84] toolbox of *MATLAB* , in which the entire *SIMULINK* model can be transferred to control directly the actuators of the system (in this case the drive motors,) and also to receive the signals from the measuring sensors. Therefore, the simulation model itself becomes the actual controller of the system, provided that the hardware is compatible with the *Real-time workshop* toolbox. The details of this are discussed in Chapter 6.

The flexibility of the FLC model in *SIMULINK* is another feature and, for example, setting the target coordinates $\underset{\sim}{q_d}$ as a variable means that a trajectory planner could be added on to it later. This would then cater for the moving target problem. The other important thing that needs to be mentioned is that all the physical parameters, like the masses of the gantry, trolley and spreader are based on the 1/8 scaled-down RTG crane experimental rig which has been constructed and described in Chapter 2. The general layout and setup of the structure has been shown in Figure 3.1. Unlike the research in Cartmell [16], no significant passive damping factors are involved in this simulation; in other words, only active damping is present during the simulation process. However, passive effects, which occur in practice, will necessarily aid the

operation of the controller by further reducing the settling time for most control scenarios (Cartmell [16]).

The common physical parameters for the experimental rig are as follows:

1) The mass of the gantry : 47.5 Kg
2) The mass of the trolley (including the hoisting devices such as motor, drives, and reeling drum etc.) : 54 Kg.
3) The mass of the spreader : 4.15 Kg.

The simulation was undertaken by calculating the relevant gains using the physical limit of the actuators (Table 3.1) and the maximum spans (Table 3.2) of the experimental rig first. The result was then compared to the physical limit of the actuators to justify whether the gains had been set up correctly (so as not to exceed the capacity of the actuators). In the beginning, only one direction (x or y) is simulated. The results are shown in sections [a] and [b]. In section [c], the simulation commences in the x- and y-directions simultaneously. Finally, the simulation invokes the actuation in all three actuated coordinates, x, y, l and the result is shown in section [d]. All the figures (Figure 3.4 ~ Figure 3.34) in this section are lumped together at the end of this chapter.

It should be noted that the inertia matrix, [D], is singular when the swing angle, ψ, is equal to zero. Because the inversion of [D] is required for the simulation, a small artificial offset, which will 'push' the payload away, is employed to get around this problem.

Actuator	Max. rpm with 180 Vdc input voltage setup (rpm)	Current input voltage setup (volts)	Current max. rpm (rpm)	Drum diameter (metre)	Max. linear velocity (m/sec)
X-axis (#1)	300	60	100	0.05	0.5236
Y-axis (#2)	120	90	60	0.05	0.3142
Rotation(#3)	120	30	20	*	120 (deg./sec)
Hoist (#4)	120	90	60	0.0375	0.2356

Table 3.1 : The physical limit of actuator velocities:

	X-axis (#1)	Y-axis (#2)	Rotation(#3)	Hoist (#4)
Span	0 ~ 1,500 mm	-500 ~ +500 mm	±90°	700 ~ 1,200 mm

Table 3.2 The maximum spans of the experimental rig:

[a]

From Table 3.1, we set $\dot{x}_{max} = 0.5$ metres/sec. The x- maximum span is taken as 1.5 metres. From equations (3.75) and (3.76),

$h_x \leq 1.8122$ ------- (3.81)

and

$g_x \leq 0.8210$ ------- (3.82)

The gains are set and the system is actuated with the following initial conditions :

$\{h\}=[1.8122\ 0.0\ 0.0\ 0\ 0]^T$, $\{g\}=[0.821\ 0.0\ 0.0\ 0\ 0]^T$

$\mathbf{q_i} = \mathbf{[0.0\ 0.0\ 1.0\ 0.0\ 0.0]^T}$, $\mathbf{q_d} = \mathbf{[1.5\ 0.0\ 1.0\ 0.0\ 0.0]^T}$

($[...]^T$ means the transpose of the relevant matrix, and $\mathbf{q_i}$ is the initial position of the spreader.)

The physically realistic maximum velocity, $\dot{x}_{max}$,of the x actuator is 0.5 m/sec

The detailed simulation result is shown in Figures 3.4 ~ 3.6, in which we acquire the test results .

$\dot{x}$ (max.) = 0.4795 m/sec ψ(max.) $\approx 7°$

t_s (settling time, within 1% of error) $\approx$ 8.5 sec

It is interesting to see the ψ diagram showing an approximate swing frequency of 0.5 Hz. This is correct owing to the fixed cable length of 1 metre, which gives a fundamental natural swing frequency of 0.4985 Hz based on a $\frac{1}{2\pi}\sqrt{\frac{g}{l}}$ approximation. This partially implies that the simulation is correctly set up.

[b]

Next, actuation in the y direction is simulated. From equations (3.77) and (3.78), with the maximum velocity of the y actuator set to 0.3 metres/sec and the y- maximum span 1 metre, the gains in the y direction will be

$h_y = 1.6310$ ------- (3.83)

and

$g_y = 0.6650$ ------- (3. 84)

The initial conditions are as follows:

$\{h\}=[0.0\ 1.631\ 0.0\ 0\ 0]^T$, $\{g\}=[0.0\ 0.665\ 0.0\ 0\ 0]^T$

$\mathbf{q_i=[0.0\ -0.5\ 1.0\ 0.006\ 0.0]^T}$, $\mathbf{q_d=[0.0\ 0.5\ 1.0\ 0.0\ 0.0]^T}$

The physically realistic maximum velocity, $\dot{y}_{max}$, of the y actuator is 0.3 m/sec

The simulation result is

$\dot{y}$ (max.)= 0.2950 m/sec $\psi(\infty)=2.7°$

t_s (settling time, within 1% of error) ≈ 14 sec

It is interesting to see that ψ no longer converges to zero, instead it approaches a dc offset of 2.7 degrees. The frequency of ψ also increases to 1 Hz. This is because a small initial offset, (0.006 radian in ψ,) has been introduced to avoid the inversion of [D] in equation (3.45) becoming singular. Also because this initial offset was not in the y-direction, which is the actuating direction in this simulation, the phenomenon of conical displacements in the α-direction was clearly inevitable. The details are shown in Figure 3.7 ~ Figure 3.10.

[c]

At this stage, actuations in the x- and y-direction are both invoked simultaneously. The initial conditions are as follows:

$\{h\}=[1.8122\ 1.631\ 0.0\ 0\ 0]^T$, $\{g\}=[0.821\ 0.665\ 0.0\ 0\ 0]^T$

$\mathbf{q_i}=\mathbf{[0.0\ -0.5.0\ 1.0\ 0.0\ 0.0]^T}$, $\mathbf{q_d}=\mathbf{[1.5\ 0.5\ 1.0\ 0.0\ 0.0]^T}$

The physically realistic maximum velocity, $\dot{x}_{max}$,of the x actuator is 0.5 m/sec

The physically realistic maximum velocity, $\dot{y}_{max}$,of the y actuator is 0.3 m/sec

This simulation result is

$\dot{x}$ (max.) = 0.4784 m/sec

$\dot{y}$ (max.)= 0.3414 m/sec $\psi(\infty)=3.06°$

t_{xs} (settling time in x -direction, within 1% of error) ≈ 9.5 sec

t_{ys} (settling time in y -direction, within 1% of error) ≈ 9.5 sec

The details are shown in Figure 3.11 ~ Figure 3.16.

Although the settling time is acceptable in both directions, the maximum y-velocity did, however, exceed the physical velocity limit of the y-actuator. Therefore, a slight reduction in the physical velocity limit of the y-actuator must be applied. The new setup and initial conditions are,

$\{h\}=[1.8122\ 1.3591\ 0.0\ 0\ 0]^T$, $\{g\}=[0.821\ 0.4617\ 0.0\ 0\ 0]^T$

$\mathbf{q_i}=\mathbf{[0.0\ -0.5.0\ 1.0\ 0.0\ 0.0]^T}$, $\mathbf{q_d}=\mathbf{[1.5\ 0.5\ 1.0\ 0.0\ 0.0]^T}$

The physically realistic maximum velocity, $\dot{x}_{max}$,of the x actuator is 0.5 m/sec

The physically realistic maximum velocity, $\dot{y}_{max}$, of the y actuator is 0.25 m/sec

This results in

$\dot{x}$ (max.) = 0.4783 m/sec

$\dot{y}$ (max.) = 0.2927 m/sec $\psi(\infty)=3.6°$

t_{xs} (settling time in x -direction, within 1% of error) $\approx$ 9.5 sec

t_{ys} (settling time in y -direction, within 1% of error) $\approx$ 12 sec

Despite the increase in the settling time in the y-direction, $\dot{y}$ did fall within the imposed limit. The details are shown in Figure 3.17~ Figure 3.22. In Figure 3.23 & 3.24, and Figure 3.25 & 3.26, the fixed cable length is set to be 1.2 metres and 0.8 metre, respectively. It is interesting to see that the different fixed cable length does not affect the outcome of the result.

[d]

Finally all three actuators are operated simultaneously. The physical velocity limit in the l-direction will be 0.22 metres/sec. From equations (3.79), (3.80), the gains are,

$h_l = 2.3918$ ------- (3.85)

and

$g_l = 1.4302$ ------- (3.86)

The initial conditions for the setup are,

$\{h\}=[1.8122\ 1.359\ 2.3918\ 0\ 0]^T$, $\{g\}=[0.821\ 0.4617\ 1.4302\ 0\ 0]^T$

$\mathbf{q_i}=\mathbf{[0.0\ -0.5.0\ 0.7\ 0.0\ 0.0]^T}$, $\mathbf{q_d}=\mathbf{[1.5\ 0.5\ 1.2\ 0.0\ 0.0]}$

The physically realistic maximum velocity, $\dot{x}_{max}$, of the x actuator is 0.5 m/sec

The physically realistic maximum velocity, $\dot{y}_{max}$,of the y actuator is 0.25 m/sec

The physically realistic maximum velocity, $\dot{l}_{max}$,of the l actuator is 0.22 m/sec

This results in

$\dot{x}$ (max.) = 0.4783 m/sec $\dot{y}$ (max.) = 0.2927 m/sec

$\dot{l}$ (max.) = 0.2182 m/sec $\psi(\infty)=\mathbf{2.5°}$

t_{xs} (settling time in x -direction, within 1% of error) $\approx$ 9.5 sec

t_{ys} (settling time in y -direction, within 1% of error) $\approx$ 12 sec

t_{ls} (settling time in l -direction, within 1% of error) $\approx$ 6.5 sec

The swinging angle, ψ, has a dc offset of 2.5°. The details are shown Figure 3.27 ~ Figure 3.34. Another test was undertaken by changing the cable length from 1.2 metres initially to the desired 0.7 metre. The test result is almost identical to the previous one except that the dc offset of ψ increases to 5.2 °.

3.6 Conclusion

Owing to the governing equations being globally decoupled by using the feedback linearization control strategy, the motions in the x-, y-, *l*-, directions can be found from the solution of a set of independent second order homogeneous error equations $\ddot{\underset{\sim}{\varepsilon}} + \{h\}\dot{\underset{\sim}{\varepsilon}} + \{g\}\underset{\sim}{\varepsilon} = 0$. From the critical damping configuration of these error equations a set of gains {h} and {g} has been determined based on the maximum velocity which the specific actuators can provide in practice and the maximum spans of the system in the x-, y-, *l*-, directions. The simulation results are reasonably good despite the fact that the coordinates ψ and α are not directly actuated.

If the trolley starts moving in the x-, y-, *l*-, directions simultaneously, a dc offset in ψ is inevitable; that is, the spreader moves as a conical pendulum over the target container. This is, of course, dissipated naturally due to the passive damping found in practice.

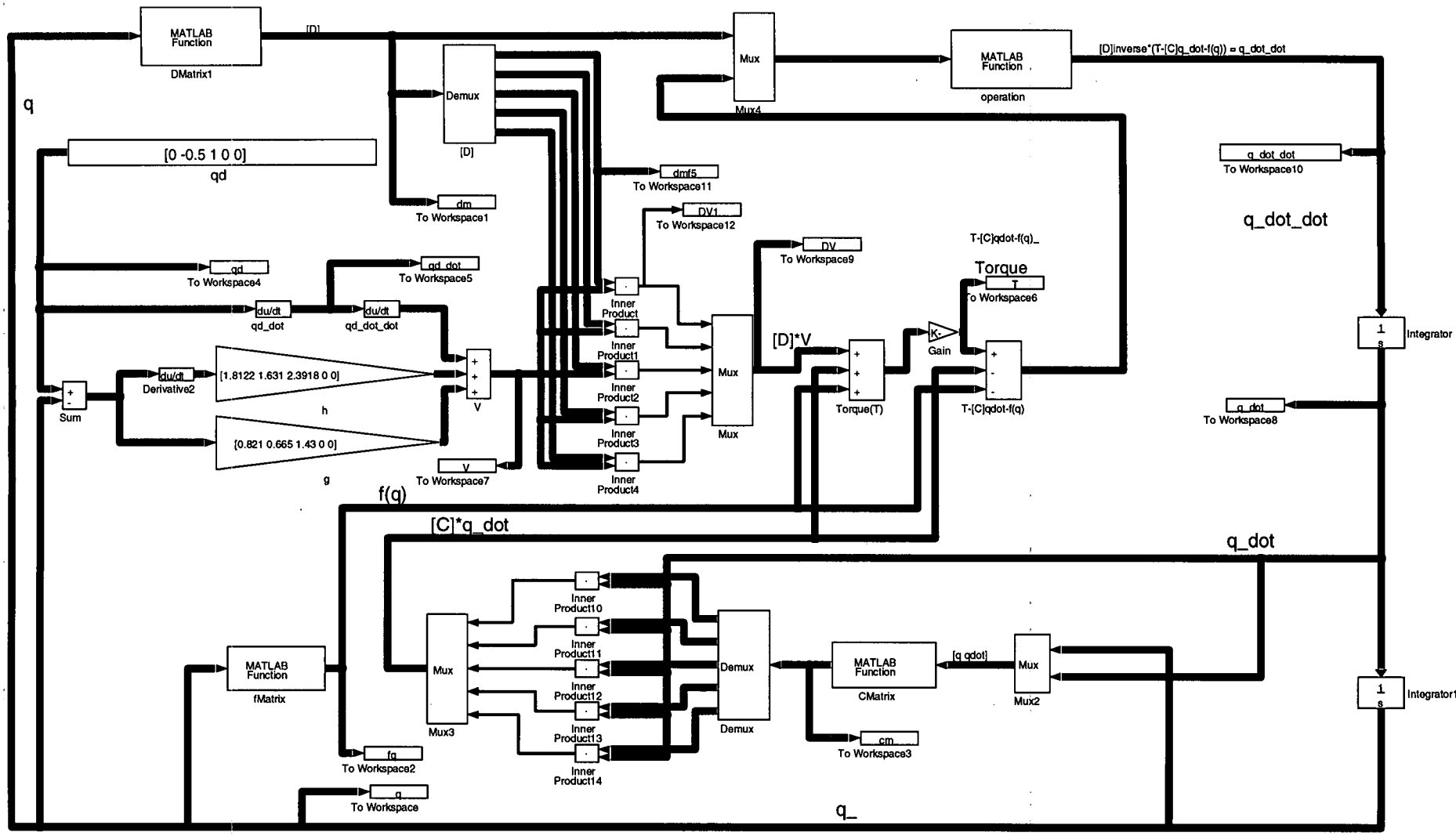


Figure 3.3 FLC model in *SIMULINK*

Simulation results for section [a] in section 3.5

Setup:

$\{h\}=[1.8122\ 0.0\ 0.0\ 0\ 0]^T$, $\{g\}=[0.821\ 0.0\ 0.0\ 0\ 0]^T$

$\mathbf{q_i=[0.0\ 0.0\ 1.0\ 0.0\ 0.0]^T}$, $\mathbf{q_d=[1.5\ 0.0\ 1.0\ 0.0\ 0.0]^T}$

The assumed physically realistic maximum velocity, $\dot{x}_{max}$, of the x actuator= 0.5 m/sec

Test results:

$\dot{x}$ (max.)= 0.4795 m/sec ψ**(max.)** $\approx$ **7°**

t_s (settling time, within 1% of error) $\approx$ 8.5 sec

x:(metre)

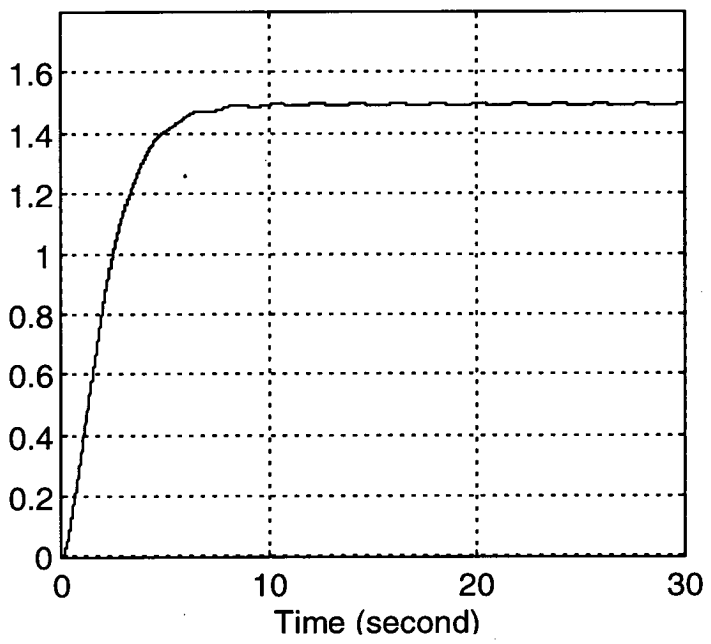


Figure 3.4 Position of the trolley in the x- direction

x-(velocity)(m/sec)

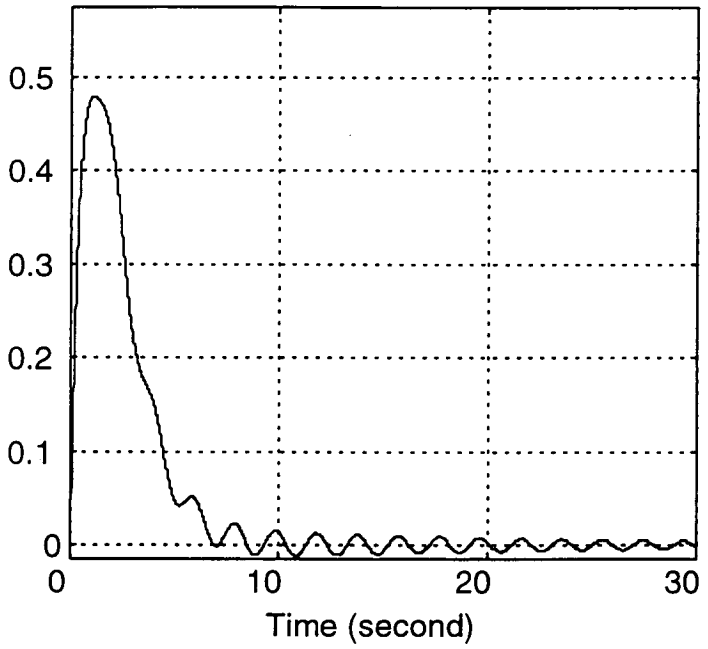


Figure 3.5

ψ:(degrees)

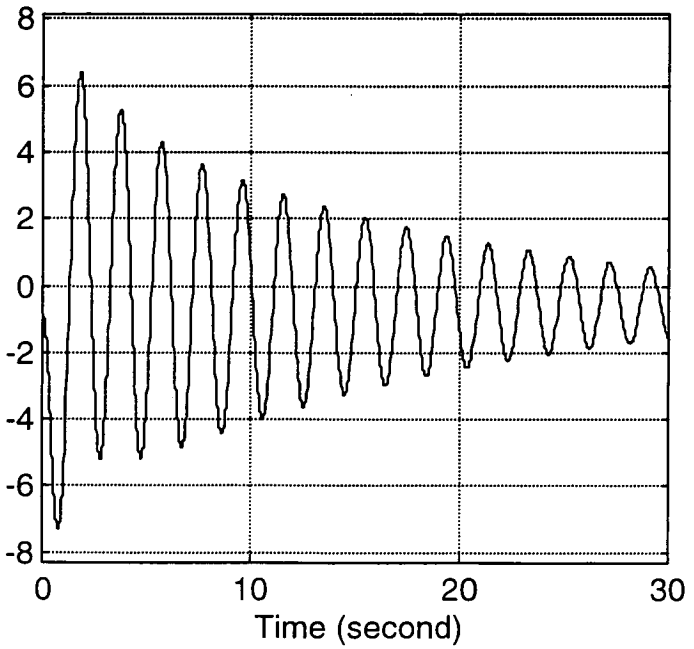


Figure 3.6

Simulation results for section [b] in section 3.5

Setup:

$\{h\}=[0.0\ 1.631\ 0.0\ 0\ 0]^T$, $\{g\}=[0.0\ 0.665\ 0.0\ 0\ 0]^T$

$\mathbf{q_i=[0.0\ -0.5.0\ 1.0\ 0.006\ 0.0]^T}$, $\mathbf{q_d=[0.0\ 0.5\ 1.0\ 0.0\ 0.0]^T}$

The assumed physically realistic maximum velocity, $\dot{y}_{max}$, of the y actuator= 0.3 m/sec

Test results:

$\dot{y}$ (max.)= 0.2950 m/sec $\psi(\infty)=2.7°$

t_s (settling time, within 1% of error) ≈ 14 sec

y:(metre)

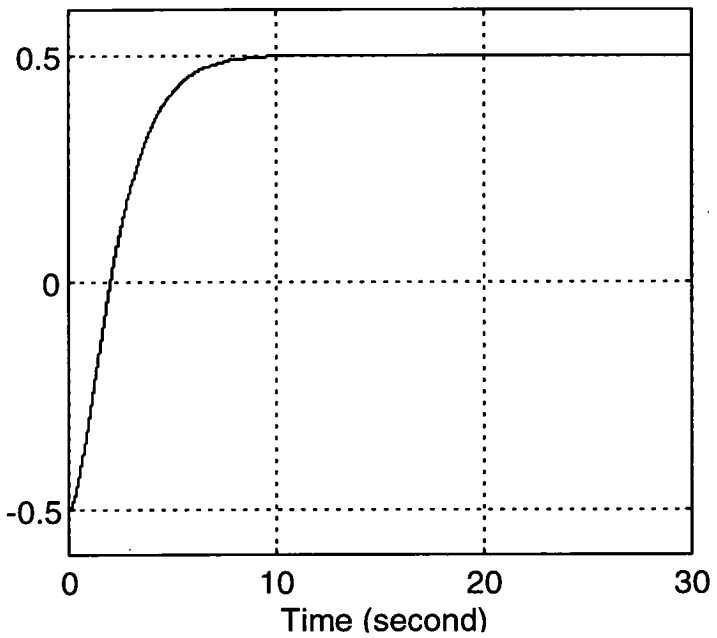


Figure 3.7 Position of the trolley in the y- direction

y-(velocity)(m/sec)

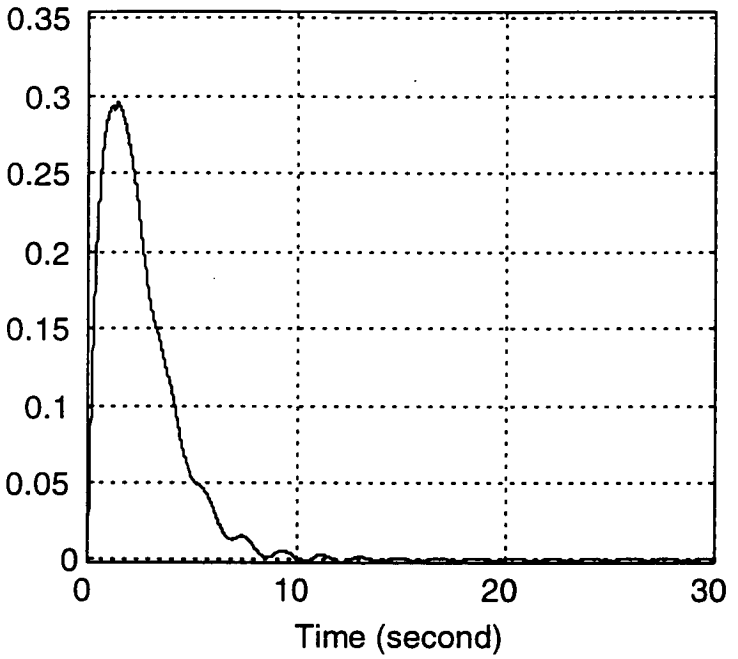


Figure 3.8

ψ:(degrees)

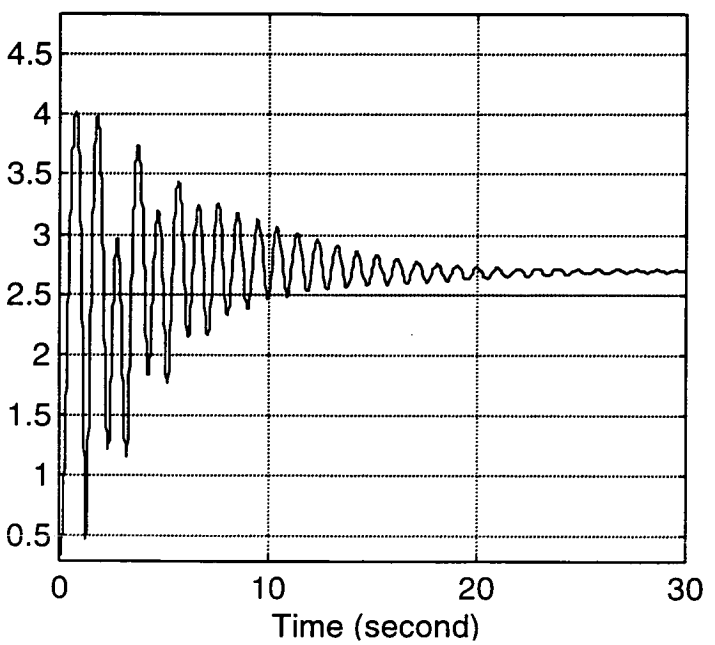


Figure 3.9

α : (degrees)

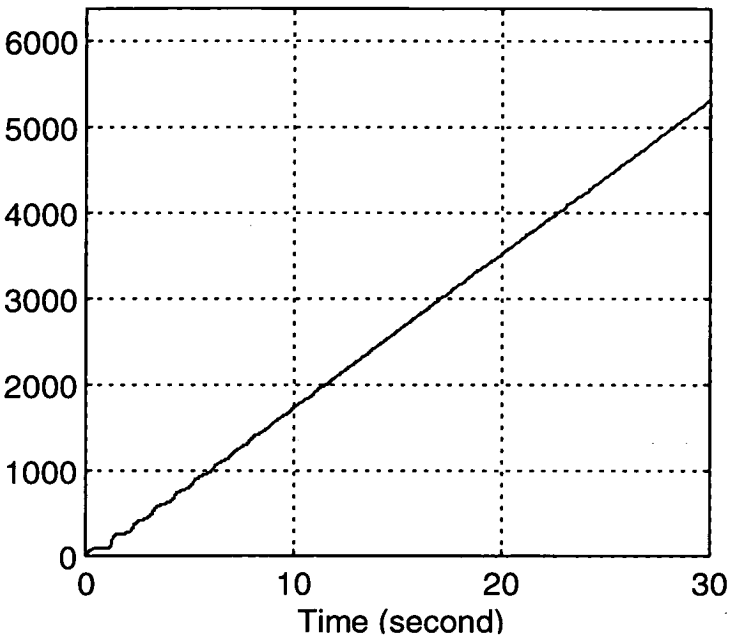


Figure 3.10 The polar angle

Simulation results for section [c] in section 3.5

$\{h\}=[1.8122\ 1.631\ 0.0\ 0\ 0]^T$, $\{g\}=[0.821\ 0.665\ 0.0\ 0\ 0]^T$

$\mathbf{q_i=[0.0\ -0.5.0\ 1.0\ 0.0\ 0.0]^T}$, $\mathbf{q_d=[1.5\ 0.5\ 1.0\ 0.0\ 0.0]^T}$

The assumed physically realistic maximum velocity, $\dot{x}_{max}$, of the x actuator= 0.5 m/sec

The assumed physically realistic maximum velocity, $\dot{y}_{max}$, of the y actuator= 0.3 m/sec

$\dot{x}$ (max.) = 0.4784 m/sec

$\dot{y}$ (max.)= 0.3414 m/sec $\psi(\infty)=\mathbf{4.06}°$

t_{xs} (settling time in x -direction, within 1% of error) ≈ 9.5 sec

t_{ys} (settling time in y -direction, within 1% of error) ≈ 9.5 sec

x:(metre)

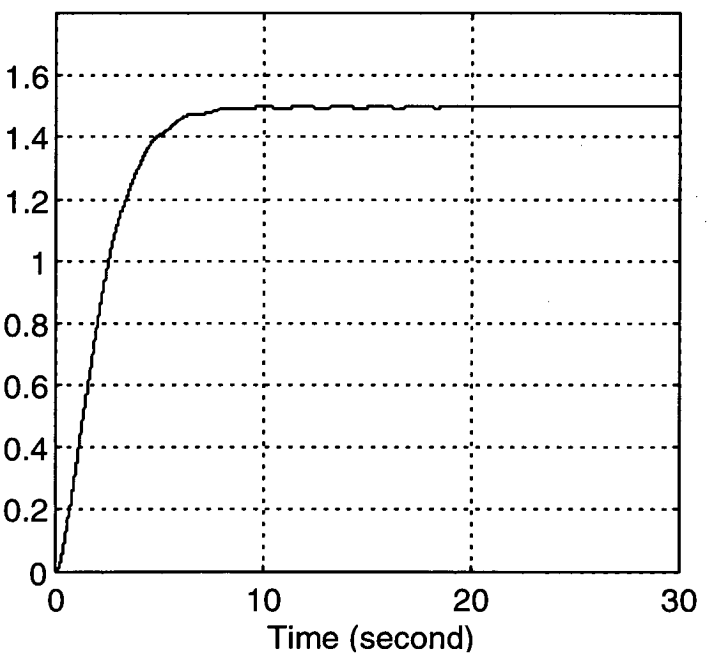


Figure 3.11 Position of the trolley in the x- direction

x-(velocity)(m/sec)

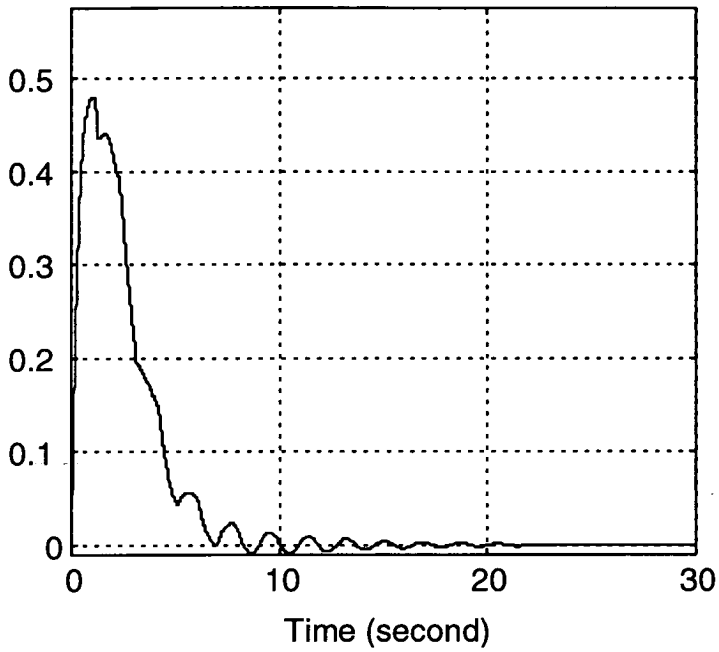


Figure 3.12 Velocity of the trolley in the x- direction

y:(metre)

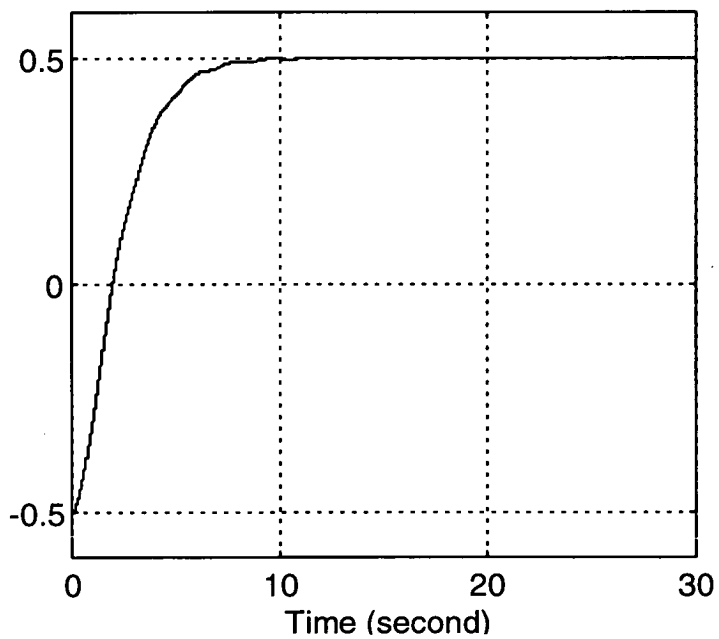


Figure 3.13 Position of the trolley in the y- direction

y-(velocity)(m/sec)

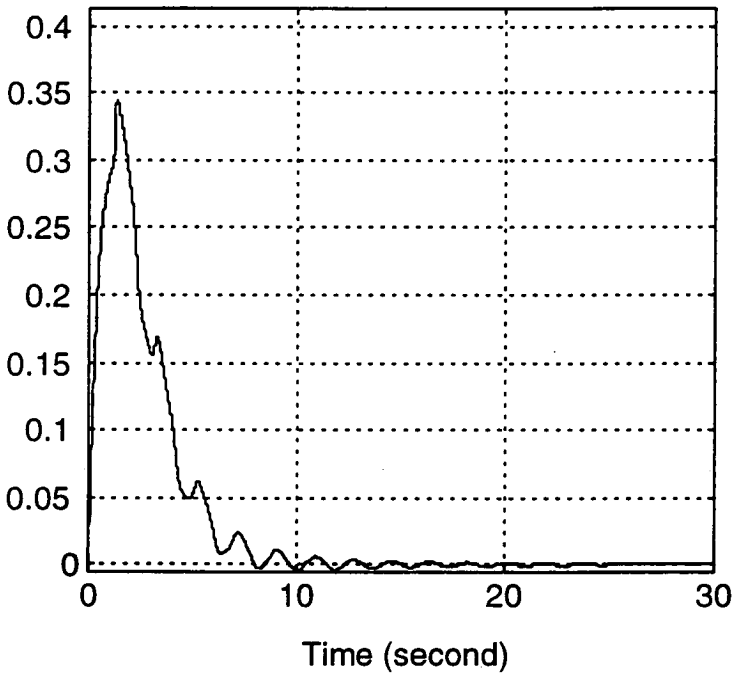


Figure 3.14

ψ:(degrees)

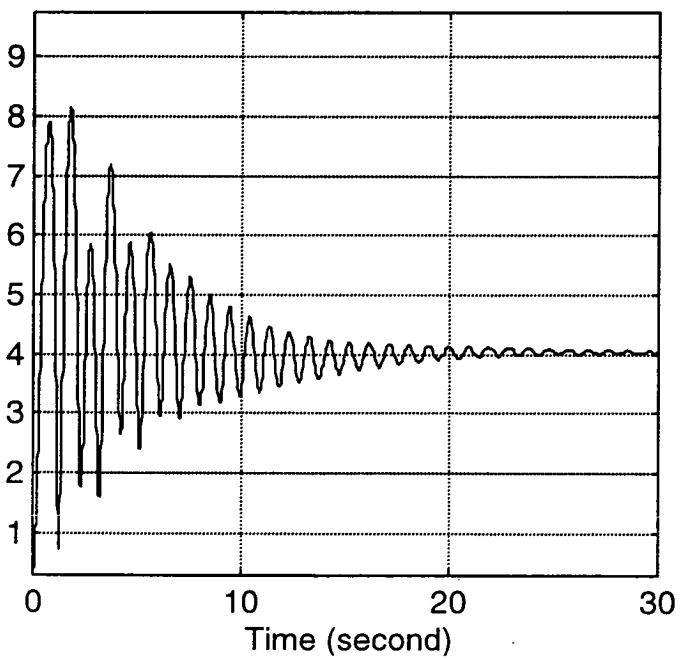


Figure 3.15

α : (degrees)

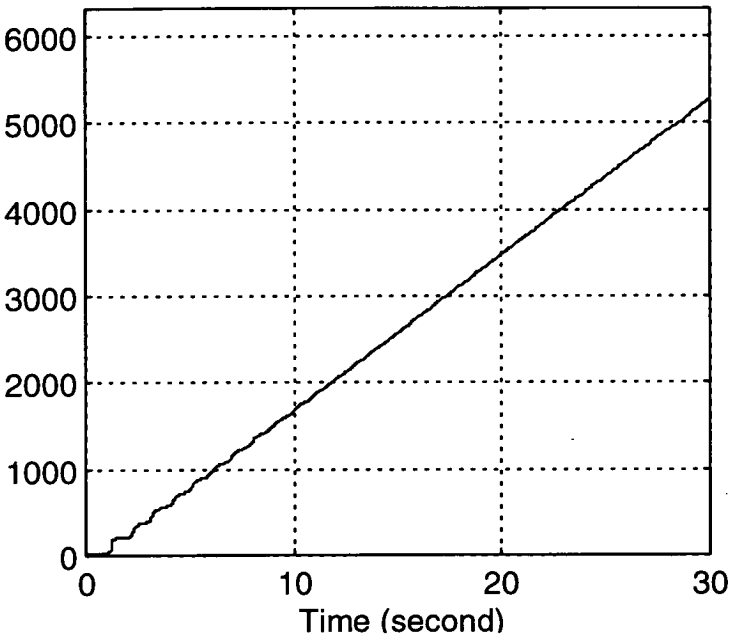


Figure 3.16 The polar angle

$\{h\}=[1.8122\ 1.3591\ 0.0\ 0\ 0]^T$, $\{g\}=[0.821\ 0.4617\ 0.0\ 0\ 0]^T$

$\mathbf{q_i=[0.0\ -0.5.0\ 1.0\ 0.0\ 0.0]^T}$, $\mathbf{q_d=[1.5\ 0.5\ 1.0\ 0.0\ 0.0]^T}$

The assumed physically realistic maximum velocity, $\dot{x}_{max}$, of the x actuator= 0.5 m/sec

The assumed physically realistic maximum velocity, $\dot{y}_{max}$, of the y actuator= 0.25 m/sec

$\dot{x}$ (max.) = 0.4783 m/sec

$\dot{y}$ (max.) = 0.2927 m/sec $\psi(\infty)=3.6°$

t_{xs} (settling time in x -direction, within 1% of error) ≈ 9.5 sec

t_{ys} (settling time in y -direction, within 1% of error) ≈ 12 sec

x:(metre)

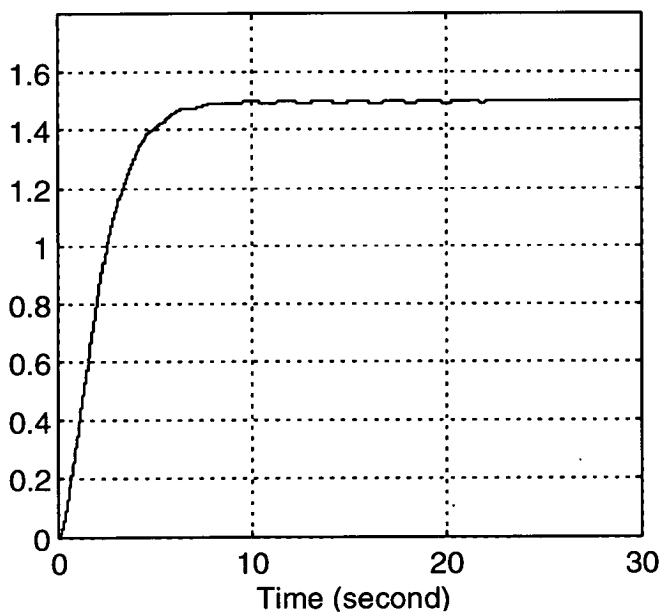


Figure 3.17 Position of the trolley in the x- direction

x-(velocity)(m/sec)

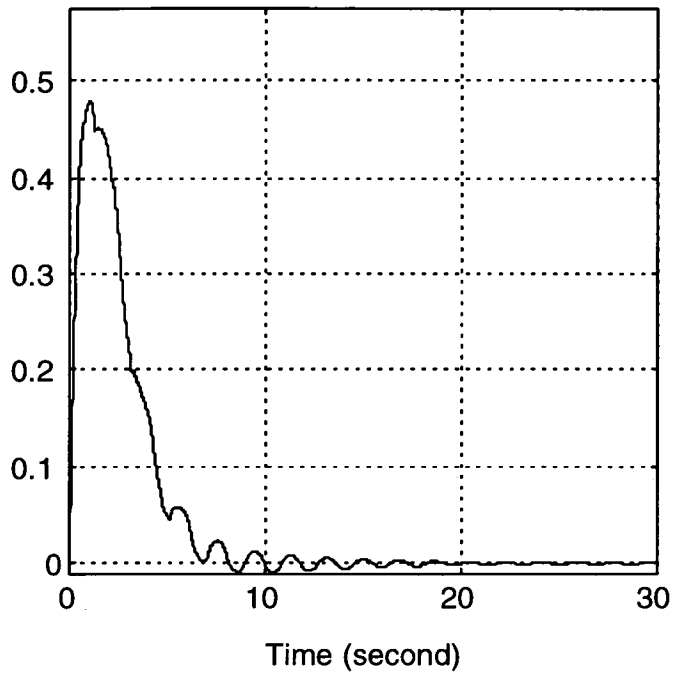


Figure 3.18 Velocity of the trolley in the x- direction

y:(metre)

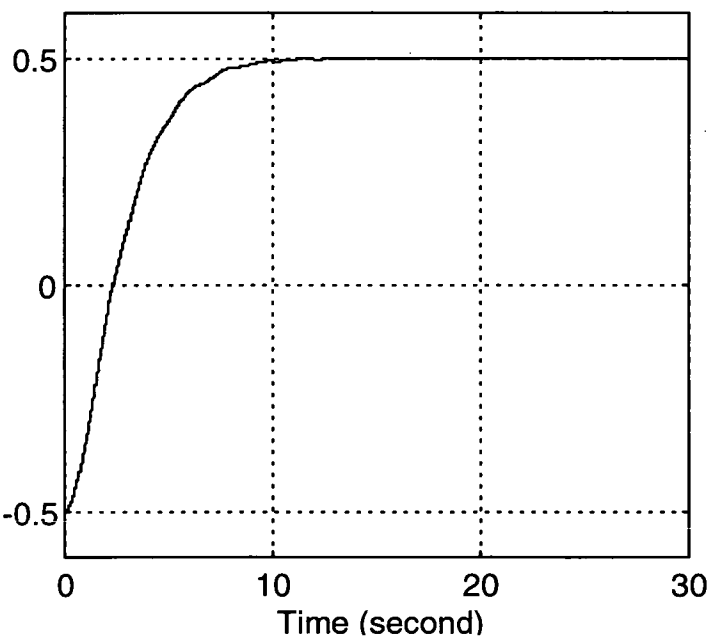


Figure 3.19 Position of the trolley in the y- direction

y-(velocity)(m/sec)

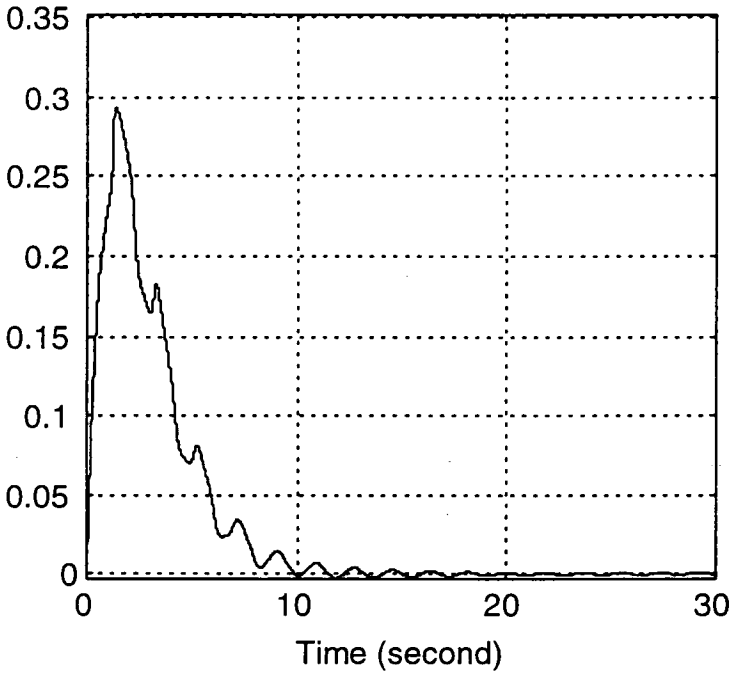


Figure 3.20

ψ:(degrees)

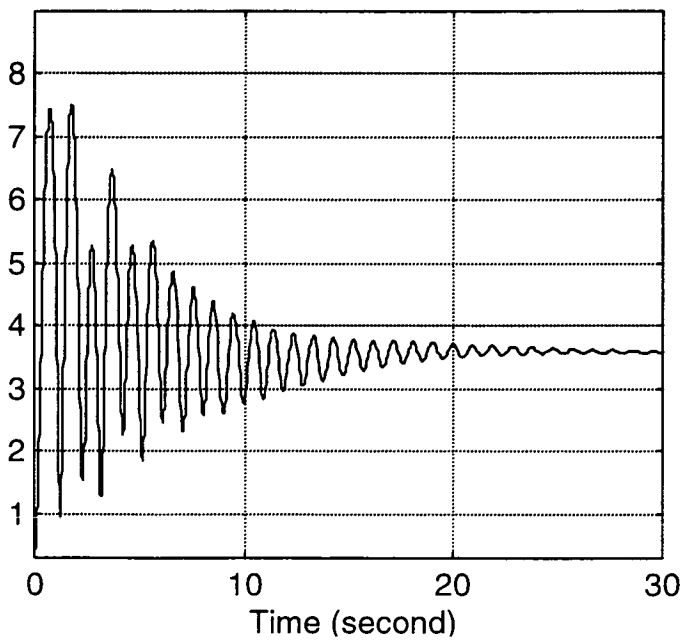


Figure 3.21

α : (degrees)

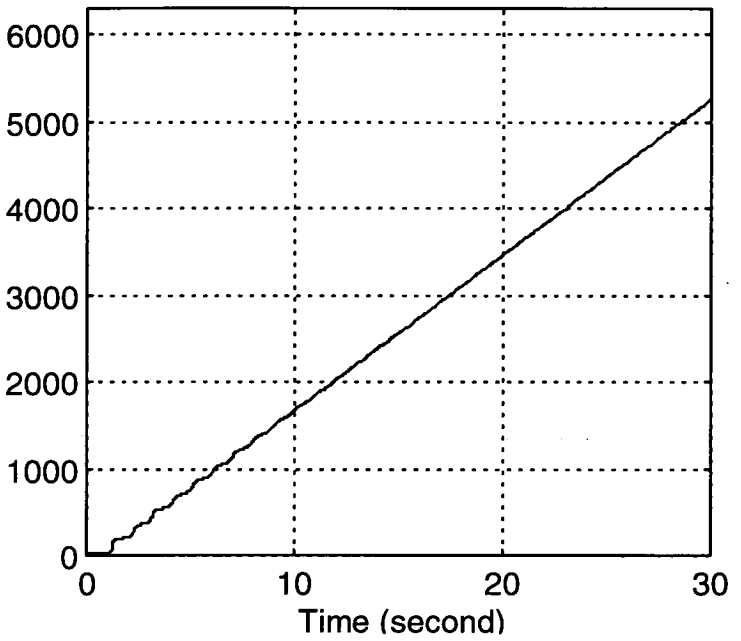


Figure 3.22 The polar angle

{h}=[1.8122 1.631 5 0 0]' , {g}=[0.821 0.665 5 0 0]'

q_i=[0.0 -0.5.0 1.2 0.006 0.0]' , q_d=[1.5 0.5 1.2 0.0 0.0]

The assumed physically realistic maximum velocity, $\dot{x}_{max}$, of the x actuator =0.5

The assumed physically realistic maximum velocity, $\dot{y}_{max}$, of the y actuator =0.25

$\dot{x}$ (max.) = 0.4783 m/sec

$\dot{y}$ (max.) = 0.2927 m/sec $\psi(\infty)$=**3.6°**

t_{xs} (settling time in x -direction, within 1% of error) ≈ 9.5 sec

t_{ys} (settling time in y -direction, within 1% of error) ≈ 12 sec

psi(ψ):(degrees)

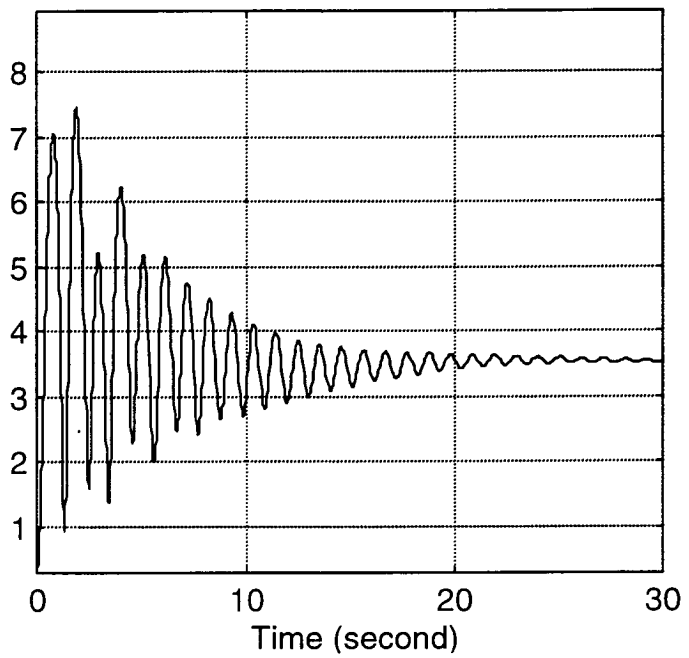


Figure 3.23

alpha(α) : (degrees)

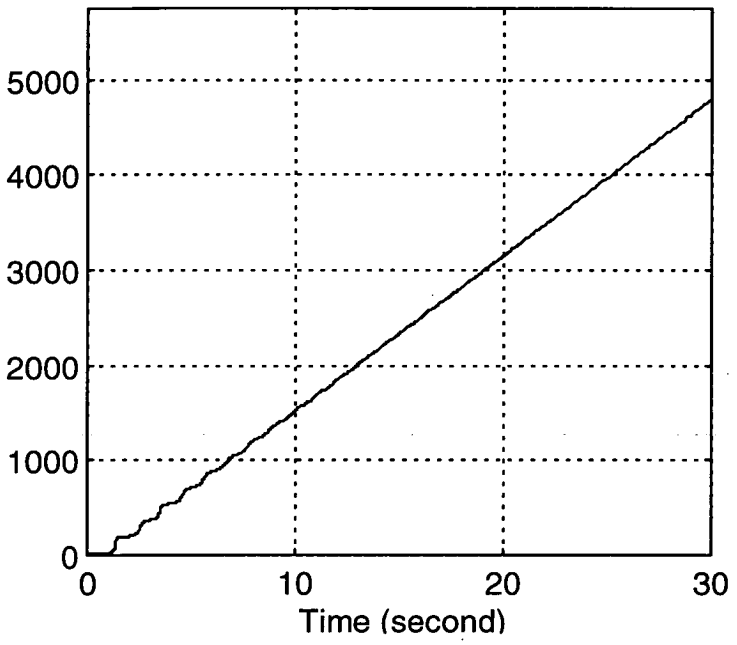


Figure 3.24 The polar angle

{h}=[1.8122 1.631 5 0 0]' , {g}=[0.821 0.665 5 0 0]'

q_i=[0.0 -0.5.0 <u>0.8</u> 0.006 0.0]' , q_d=[1.5 0.5 <u>0.8</u> 0.0 0.0]

The assumed physically realistic maximum velocity, $\dot{x}_{max}$, of the x actuator =0.5

The assumed physically realistic maximum velocity, $\dot{y}_{max}$, of the y actuator = 0.25

$\dot{x}$ (max.) = 0.4783 m/sec

$\dot{y}$ (max.) = 0.2927 m/sec $\psi(\infty)$=**3.6°**

t_{xs} (settling time in x -direction, within 1% of error) ≈ 9.5 sec

t_{ys} (settling time in y -direction, within 1% of error) ≈ 12 sec

psi(ψ):(degrees)

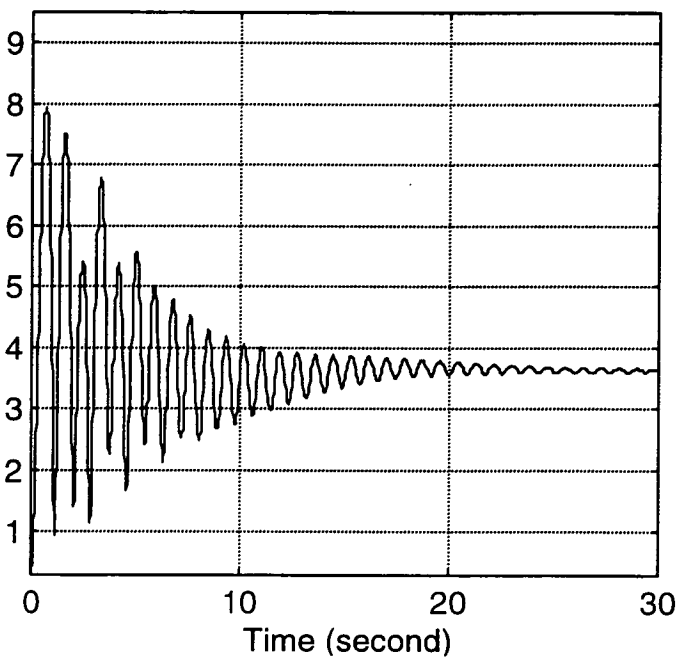


Figure 3.25

alpha(α) : (degrees)

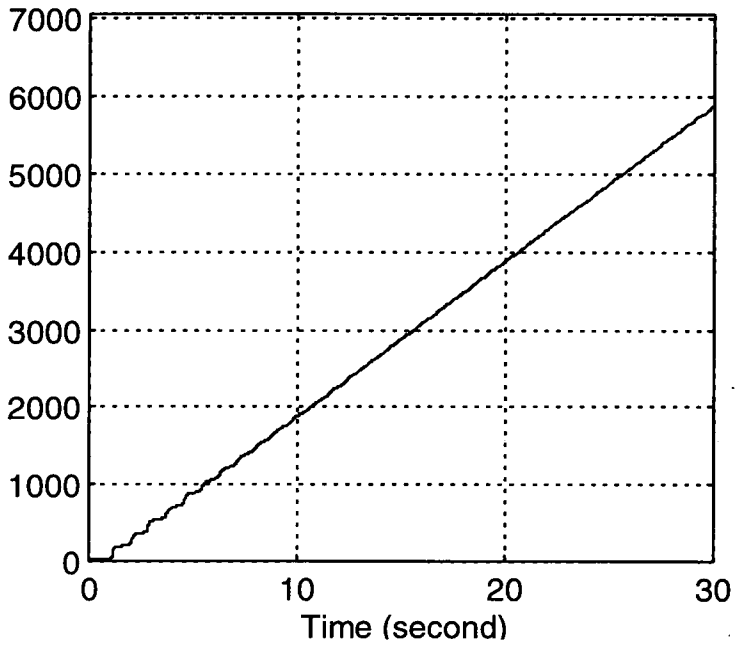


Figure 3.26 The polar angle

Simulation results for section [d] in section 3.5

$\{h\}=[1.8122\ 1.359\ 2.3918\ 0\ 0]^T$, $\{g\}=[0.821\ 0.4617\ 1.4302\ 0\ 0]^T$

$q_i=[0.0\ -0.5.0\ 0.7\ 0.0\ 0.0]^T$, $q_d=[1.5\ 0.5\ 1.2\ 0.0\ 0.0]$

The assumed physically realistic maximum velocity, $\dot{x}_{max}$, of the x actuator= 0.5 m/sec

The assumed physically realistic maximum velocity, $\dot{y}_{max}$, of the y actuator= 0.25 m/sec

The assumed physically realistic maximum velocity, $\dot{l}_{max}$, of the l actuator= 0.22 m/sec

$\dot{x}$ (max.) = 0.4783 m/sec $\dot{y}$ (max.) = 0.2927 m/sec

$\dot{l}$ (max.) = 0.2182 m/sec $\psi(\infty)=2.5°$

t_{xs} (settling time in x -direction, within 1% of error) ≈ 9.5 sec

t_{ys} (settling time in y -direction, within 1% of error) ≈ 12 sec

t_{ls} (settling time in l -direction, within 1% of error) ≈ 6.5 sec

x:(metre)

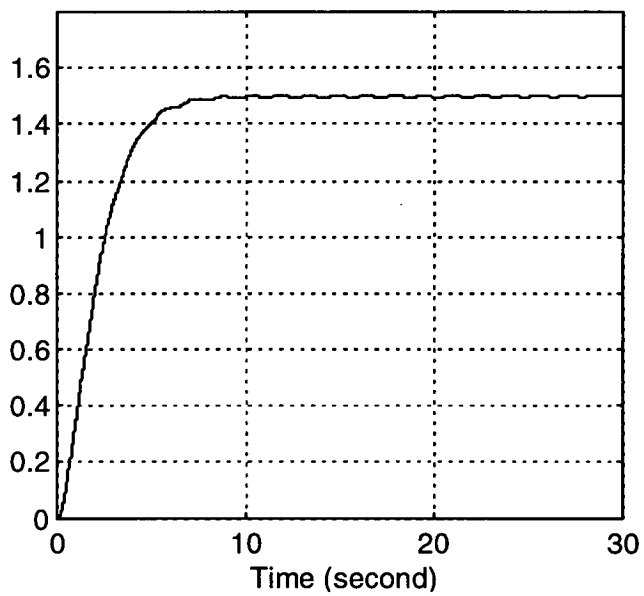


Figure 3.27 Position of the trolley in the x- direction

x-(velocity)(m/sec)

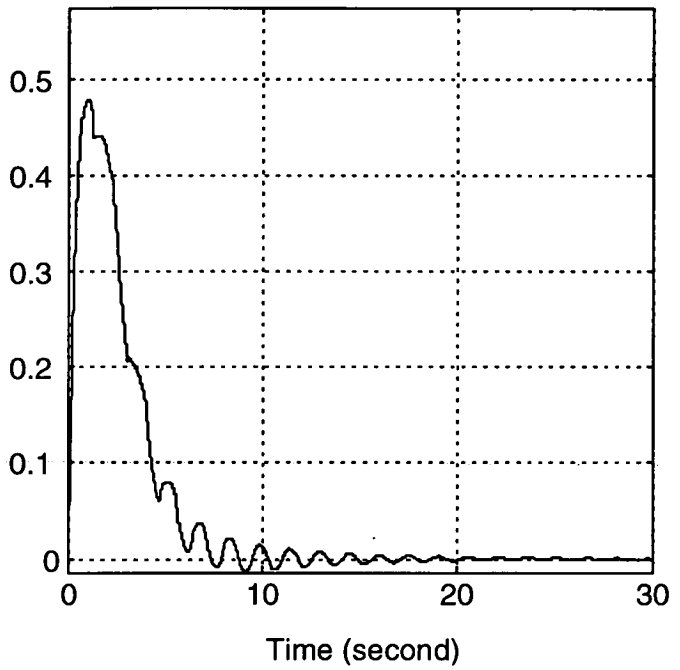


Figure 3.28 Velocity of the trolley in the x- direction

y:(metre)

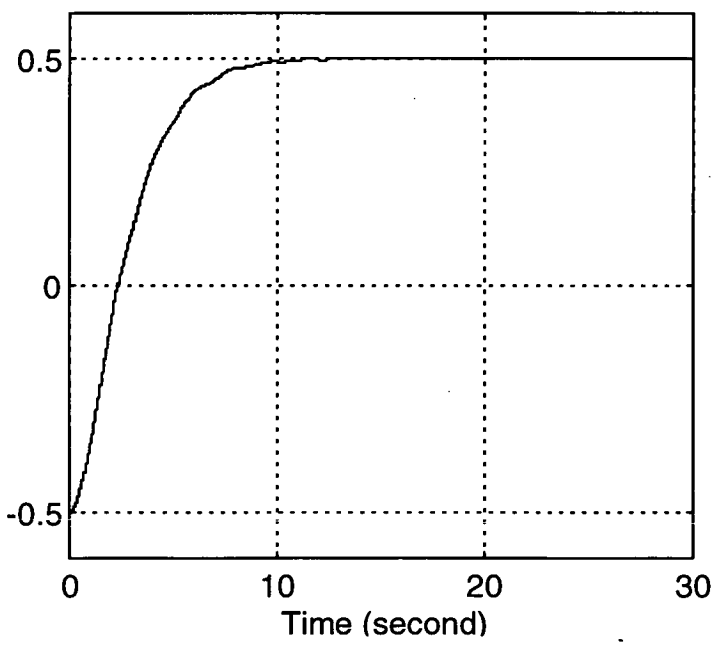


Figure 3.29 Position of the trolley in the y- direction

y-(velocity)(m/sec)

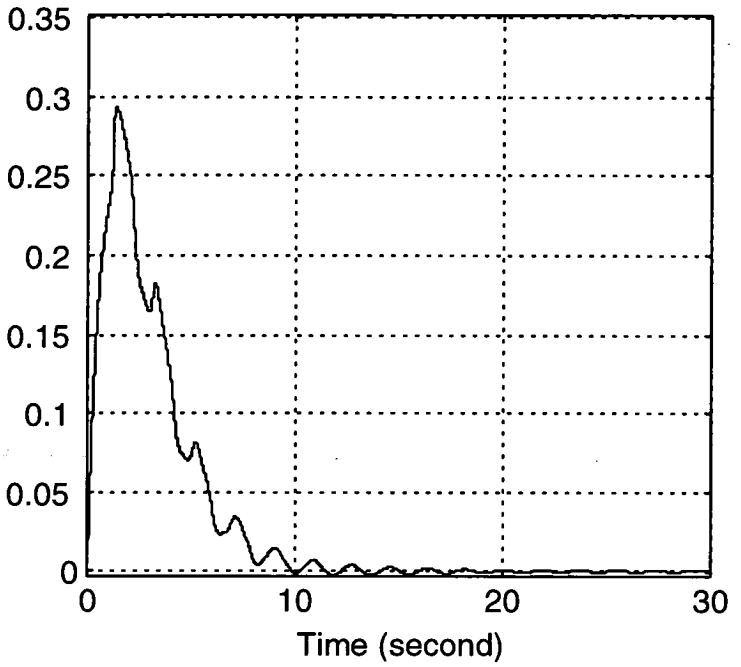


Figure 3.30 Velocity of the trolley in the y- direction

l:(metre)

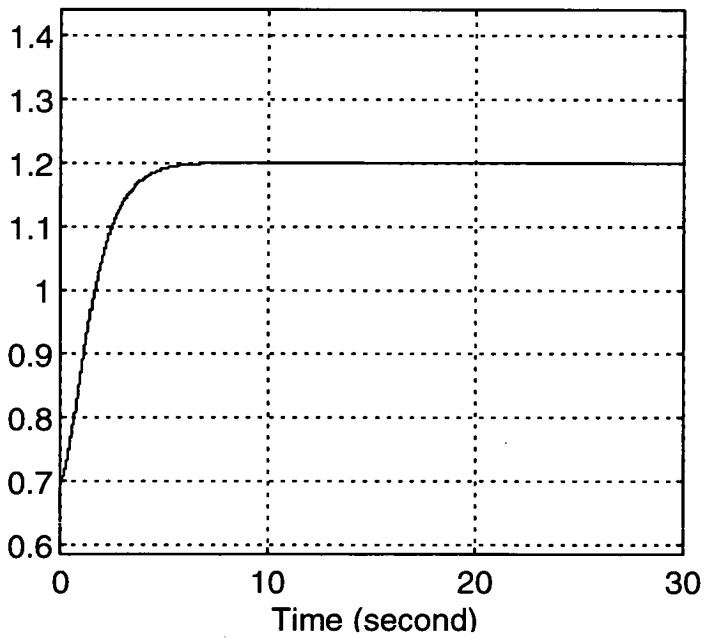


Figure 3.31 Position of the trolley in the *l*- direction

l-(velocity)(m/sec)

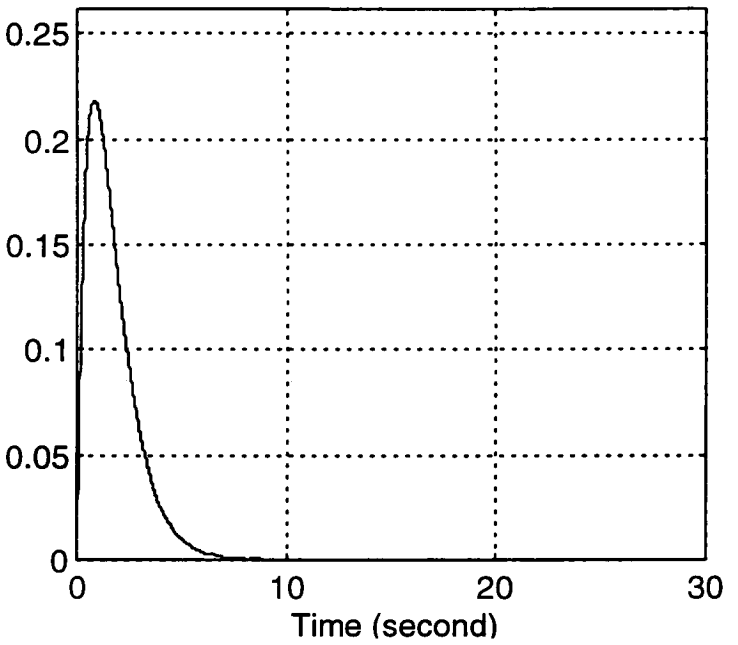


Figure 3.32

ψ:(degrees)

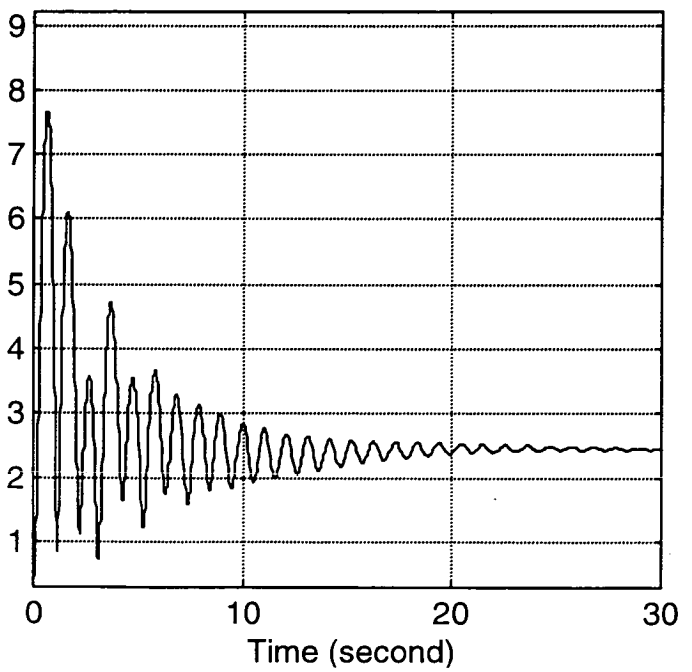


Figure 3.33

α : (degrees)

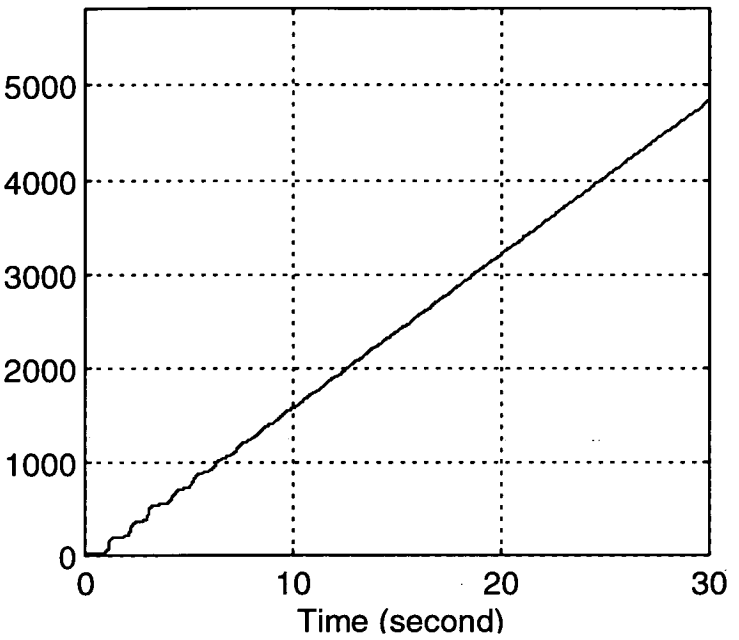


Figure 3.34 The polar angle

Chapter 4

The Design Of The Experimental RTG Crane Rig

4.1 Introduction

In spite of the existing literature mentioned in Chapters 2 & 3, much of which employs different control strategies, implementation into a real working environment has been slow compared to progress made on the development of theory. Most of the published work to date is merely based on numerical simulation. Among the papers dealing with experimental analysis, two degrees of freedom, x (the actuating direction of the trolley), and ψ (the swing angle of the cable) are commonly applied (Benhidjeb [5], Butler [10], Mason [54], Yoon [96]). In these cases the *single-cabled* payload moves only in the plane of motion and with a fixed cable length. In the work presented by Hämäläinen [35], Marttinen [51][52][53], Virkkunen [88], another actuated degree of freedom, l (the length of the cable, or *hoist* length) was made available on the experimental facility. However, the payload was still contained in the plane of motion. There appears to be no relevant work on the construction of an experimental crane capable of actuation in the directions of x (traveling motion of the trolley), y (transverse motion of the gantry), l (hoisting), together with a facility for simulating the dynamics of a payload with a multi cable model (MCM) (Cartmell [14][15][16], Morrish [57]). Furthermore, for economic reasons protracted field tests are usually impossible in practice. So it was then considered that it would be useful to design a dedicated experimental crane rig with many of the characteristics of a full-sized RTG crane, so that the theoretical development in Chapter 2 & 3 could be implemented, tested and refined.

In this chapter, the design and function of the experimental crane rig is illustrated. In sections 4.1 & 4.2, the *mechanical* and *electrical* aspects of the design for the experimental crane rig are discussed, respectively. In section 4.3, the hardware setup of the vision sensing system is described. Figure 4.12 at the end of this chapter summarizes the configuration and the relationship between the hardware and software.

4.2 The mechanical aspects of the experimental crane rig

An RTG crane is used to transfer containers between different locations around a freight yard or port. Once a container has been positioned correctly alongside the desired stack the gantry motion is disabled and the brakes applied to the rubber tyred wheels. Thereafter hoist and trolley motion can be operated at the same time to retrieve or re-stack the containers. Although simultaneous hoisting/trolley motion plus gantry travel is limited for power-supply reasons in full-sized RTG cranes, the proposed experimental crane is purposely designed to be capable of actuating concurrently in those directions; that is, traveling motion of the trolley (x), transverse motion of the gantry (y), and hoisting (l). In addition to this capability, a fourth actuator is also provided to simulate overall rotational motion of the gantry, θ_V.

The experimental crane is to **1/8** scale (as shown in Figure 4.13, 4.14 and 4.15 at the end of this chapter), based on dimensions to give the rig representative behaviour of a full-sized RTG crane. The rig is designed to be capable of operating either manually or fully automatically using the electrical drives and controls described in section 4.3. The layout of the rig and the control equipment is shown in Figure 4.1.

[a] The x- axis

The x - axis is representative of the direction of trolley travel along the top-beam of a full-sized crane. One pair of linear sliding guide-rails provides smooth linear movement with a maximum span of 1,500 mm (0 to 1,500 mm). A dc permanent magnet gearbox/motor (#1 motor) is used to drive the subassembly underneath the rails along the x -axis. This subassembly includes the y - axis sliding guide rails and the trolley as shown in Figure 4.1. A steel cable, connecting a drum mounted on the motor and a pulley at the far end of the rails, provides the drive actuation. The measured maximum speed along the x - axis is 0.543 m/sec (in Figure 4.5 at the end of the chapter).

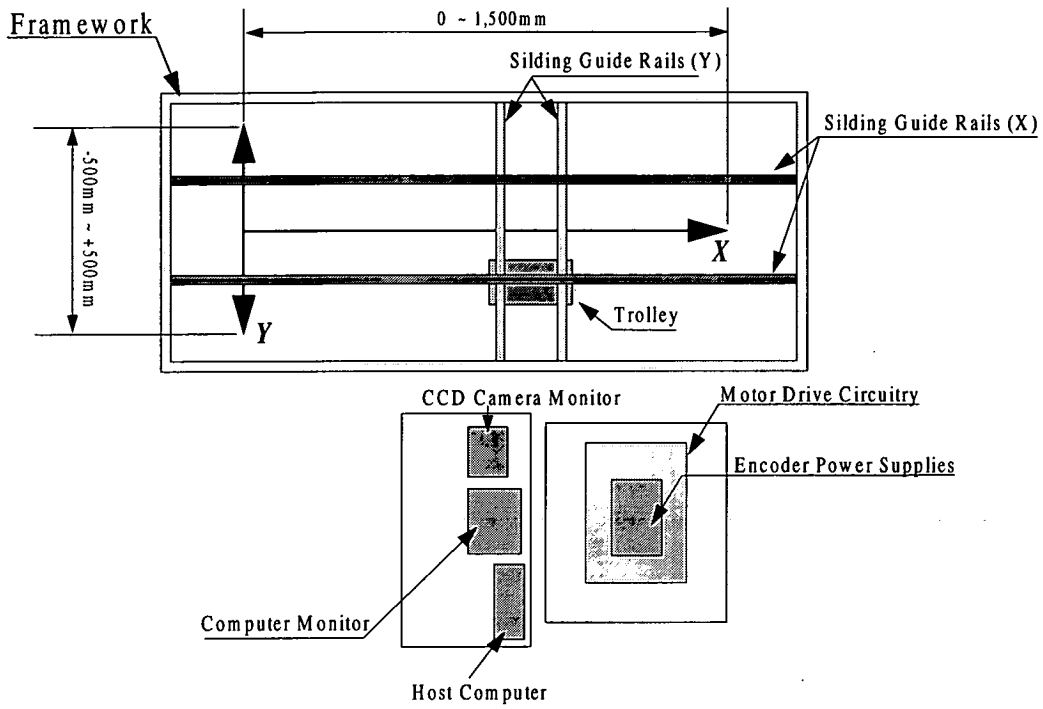


Figure 4.1 A plan view of the experimental crane rig

[b] The y- axis

The y- axis represents the direction of gantry motion in a full-sized crane. Another pair of linear sliding guides provides smooth linear movement with a maximum span of 1,000 mm (-500 mm to 500 mm). A dc permanent magnet gearbox/motor (#2 motor) is used to drive the trolley underneath the rails along the y- axis as shown in Figure 4.1. The trolley is actuated by a similar drive system (cable/drum/pulley) as for the x- axis. The measured maximum speed along the y- axis is 0.313 m/sec (in Figure 4.7 at the end of the chapter).

[c] The θ_V - axis

This subassembly provides rotational motion, simulating differential steering of a full-sized crane. Whereas the full-sized crane turns the whole gantry by pivoting about one of its four wheels, the experimental crane just rotates the trolley about its vertical (central) axis. A dc permanent magnet gearbox/motor (#3 motor) is used to rotate the hoist system underneath. The maximum span along the θ_V - axis is 180° (-90° to +90°) and 120 degrees/sec. The measured maximum angular speed is 105.6 degrees/sec (in Figure 4.9 at the end of the chapter).

[d] The *l* - axis

The hoist subassembly provides the hoist/lower motion, as in a the full-sized crane. A dc permanent magnet gearbox/motor (#4 motor) is used to rotate the hoist drum which drives the spreader underneath. The maximum span along the *l* - axis is 500 mm (700 to 1200 mm). The measured maximum speed is 0.255 m/sec (in Figure 4.11 at the end of the chapter).

The general specification for each axis is summarized in Table.4.1

	X-axis (#1)	Y-axis (#2)	Rotation(#3)	Hoist (#4)
Span	0 ~ 1,500 mm	-500 ~ +500 mm	±90°	700 ~ 1,200 mm
Measured maximum speed	0.543 m/sec	0.313 m/sec	105.6 degrees/sec	0.255 m/sec

Table. 4.1 The maximum displacement limits of the drives on the experimental rig

4.3 The Electrical aspect of the experimental crane rig

In this section, the design of the electrical part of the experimental rig is illustrated. Basically, it can be categorized into three groups as follows:

4.3.1 The Power Supply and Direction Changing Circuitry for the Motors

In order to actuate the four permanent dc motors, four proprietary control modules (RS-CUBE [64]) are employed to produce appropriate voltage and current drive. The speed of each motor is controlled in the module by the use of an internal, linear, closed-loop feedback signal based on the dc armature voltage, enabling the unit to maintain a constant motor speed for variable motor loads. The corresponding output speed of a motor is *proportional* to a range of dc voltage (0 ~ 10 volts) input to the modules. This feature is verified in Figure 4.4 ~ Figure 4.11 (lumped together at the end of this chapter) by directly applying a voltage, V_{in}, to the drive module of each motor and then measuring the frequency of the relevant shaft encoder. It is noted that there is an offset in the V_{in}- axis of each motor. This is caused by the minimum speed setting on the drive module, because the motors may overheat at low speed (RS-CUBE [64]).

The direction of the motors can be changed by altering the polarity of the voltage applied to it, via heavy duty fast acting relays. The electrical components for the above mentioned functionality are contained in the *Motor Drive Circuitry* in Figure 4.1. Figure 4.16 shows the layout of the host computer and the control circuitry.

In practice the crane driver gives reference values for velocity of hoisting and transversing by means of a joystick (Ferranti [31]). Hence, manual control is also available in the experimental crane. This is done by directly supplying 0 ~ 10 volts from an external power source using four potentiometers for each motor drive module. Clearly this facility is disabled when the computer based automatic controller is deployed.

4.3.2 The interface between the host computer and the motor drive modules

In order to correctly output 0 ~ 10 dc volts to control the motor speed, a digital to analogue converter (DAC) is placed between the host computer and the drive modules. An 8-channel analogue output board plugged into one of the ISA buses in the host computer is utilised for this purpose. It provides 12-bit resolution over the output range; that is, resolution down to $10/2^{12} = 2.44\times10^{-3}$ volts. To output a dc voltage through a channel, some basic functions can be called from a C library (Data [24], CIO-DAC08 [87], Universal [81]) supplied by the manufacturer, or values can be directly sent to the address registered for that channel. These functions are closely related to the real time implementation which is discussed further in Chapter 5 & 6.

4.3.3 The interface for feedback signals from motors

In order to obtain the actual speed of the motors, four optical shaft encoders of a resolution of 2,500 ppr (pulses per revolution) are used. There are two signals, A & B, which are output from the encoders, both of which can be used for detecting the position or the speed of the motors. The phase difference between them is also employed to detect the direction of the rotation. A proprietary computer board, is used to count the number of pulses from the encoders. There are ten counters on the computer board, and each of these has a 16-bit (65,526 counts) count register. A variety of configuration modes (Data [24], CIO-CTR10 [86], AM9513 [2]) can be setup to each counter for different types of measurement. Four of them are configured for the position measurement in x (#1), y (#2), θ_V (#3), l (#4). Four of the others are configured for frequency measurement. There are also sixteen digital inputs (D/I) on this board, four of which are used to detect the instantaneous direction of all the motors. Similarly, four of the sixteen digital outputs (D/O) on the computer board are used to control the relays (section 4.3.1) to change the direction of the motors by altering the polarity of the applied voltage from the drive modules.

4.4 The vision system for the experimental crane rig

Generally speaking, two devices are required for image processing. The first is a physical sensor that is sensitive to a required band in the electromagnetic energy spectrum (such as the x-ray, ultraviolet, visible, or infrared bands) and which produces an electrical signal output proportional to the level of energy sensed. The second is a system for converting the electrical output of the physical sensor into digital form for further processing. In the case of the vision sensing system on the experimental crane rig, a CCD camera and an image processing board (a *framegrabber*) are both employed for these two tasks, respectively.

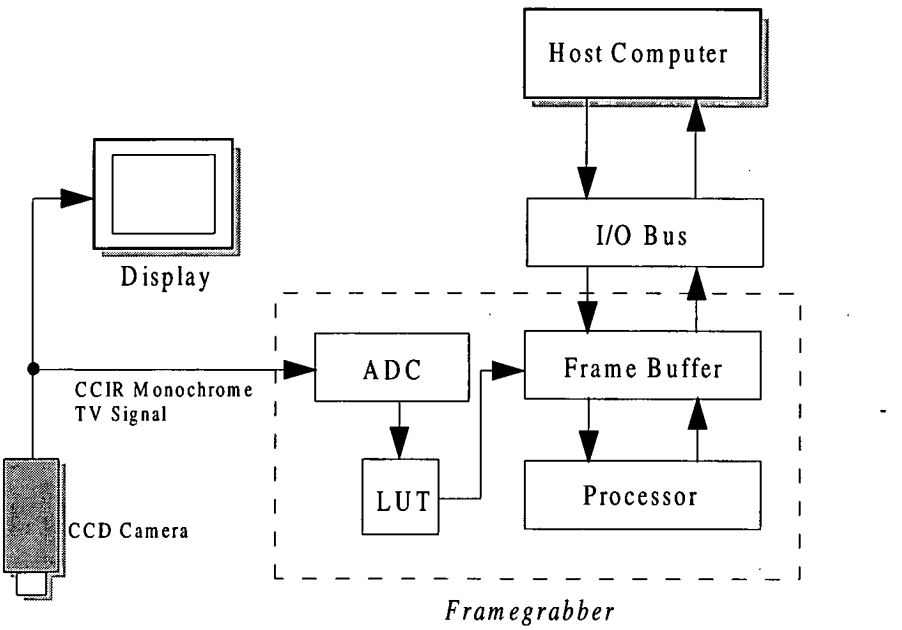


Figure 4.2 The vision system for the experimental rig

As shown in Figure 4.2, images are first retrieved by the CCD camera mounted on the trolley of the experimental crane, shown in Figure 4.17, and then the composite TV signals are passed to the framegrabber to be digitized in order to search for the coordinates of the two lighting sources on the spreader (section 3.2). The host computer, on the other hand, is only used at this stage to monitor and receive the calculated coordinates from the framegrabber through the I/O bus, once the downloaded programs on the framegrabber are initialized. A 9" monochrome TV monitor is used to 'observe' directly the motion of the spreader under the CCD camera (Figure 4.12). In the following sub-section, the relevant features of the CCD camera and framegrabber are presented.

4.4.1 The CCD camera

The video camera used in the experimental rig is a CCD (charge-coupled device) monochrome camera. A CCD array is a type of solid-state sensor which , when exposed to light, provides charge packets proportional to the incident light intensity developed in each element in the array. The stored charge packets are then sequentially shifted to the storage registers and composed as a composite signal. In this application, a standard CCIR *50 Hz* , *interlaced* TV signal is used. The camera lens has a focal length , f , of 8.5 mm, and a *field of view* of 16 degrees in the vertical direction and 21 degrees in the horizontal direction (Figure 4.3).

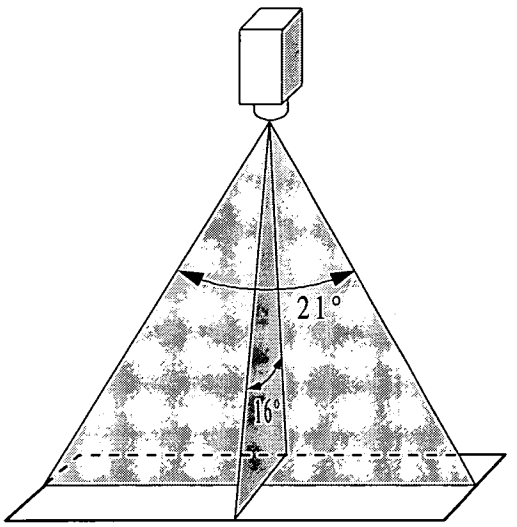


Figure 4.3 Field of View (FOV) of the camera

4.4.2 The framegrabber

Processing of digital images involves procedures that are usually expressed in an algorithmic form. Thus, with the exception of image acquisition and display, most image processing functions can be implemented in software. Although large-scale image processing systems are still being sold for massive imaging applications, such as processing of satellite images, the trend continues toward miniaturisation and the merging of general-purpose image processing boards into small personal computers. Generally, they consist of an analogue-digital-converter (ADC), a look-up table (LUT), a frame buffer and an on-board processor. At first the analogue TV signal is

digitized and passed to the look-up table, which can be set up to provide non-linear translation of the input data. The frame buffer is used to store temporarily the image data and also to keep the downloaded internal control programs sent from the host computer. The on-board processor then executes the instructions in the programs to process the image data in the frame buffer. The I/O bus is responsible for transferring the calculated results to the host computer and is also ready to receive commands for other tasks.

The framegrabber selected for the experimental crane rig is a full-sized ISA bus board. There is a 256 byte look-up table and 2 megabyte DRAM (dynamic random access memory) for the frame buffer. Additionally a programmable threshold may be set up to produce a *binary* image. The on-board processor is an Intel 960CA (i960) RISC type CPU. The framegrabber can be initialized in a *stand-alone* mode that means it will work independently from the host computer. This allows more CPU time on the host computer to be freed up for other assignments such as handling the feedback signals from the encoders.

4.5 Conclusion

For practical testing of the control strategies developed for the RTG crane, an advanced 1/8 scaled down crane has been designed and constructed. In this chapter, the functions and constraints of this experimental rig have been described. This facility not only provides a platform for implementing the feedback linearization controller given in Chapter 3, but also other control strategies which may be developed in the future. Because this experimental rig has been engineered to be as versatile as possible, the end-users do not necessarily need to spend a significant time in setting up the interfaces between the mechanical, electrically powered devices and the host computer. All that is needed is to produce a *block diagram* of the desired controller, which can then not only be simulated in the preliminary stage (Chapter 3) but can also be converted to physical control software (Chapter 6), which can then be implemented on the experimental rig. The sophisticated interfaces are *transparent* and can be considered as *black boxes* to the end user. Therefore the main effort can be focused on investigating control algorithms of interest rather than in spending a significant amount of time going through low-level sensing and data acquisition issues. Configuring the experimental system (rig and electrical and software control system) in this manner has been a major part of this research. The rig as a test-bed for future high-speed crane control research is considered to be one of the major achievements of the work.

Moreover, all the components used in the experimental crane rig, such as the motors, the encoders, the CCD camera, the framegrabber, data acquisition boards and the host computer are general purpose devices (so-called 'off-the-shelf' components). To design this way not only reduces the overall complexity of the system but also implies that such technology can readily be transferred to the practical application easily and cost-effectively.

In the next chapter, the theory of the vision sensing system for the experimental crane is discussed. Also, the details of the relationships between the hardware and software are illustrated.

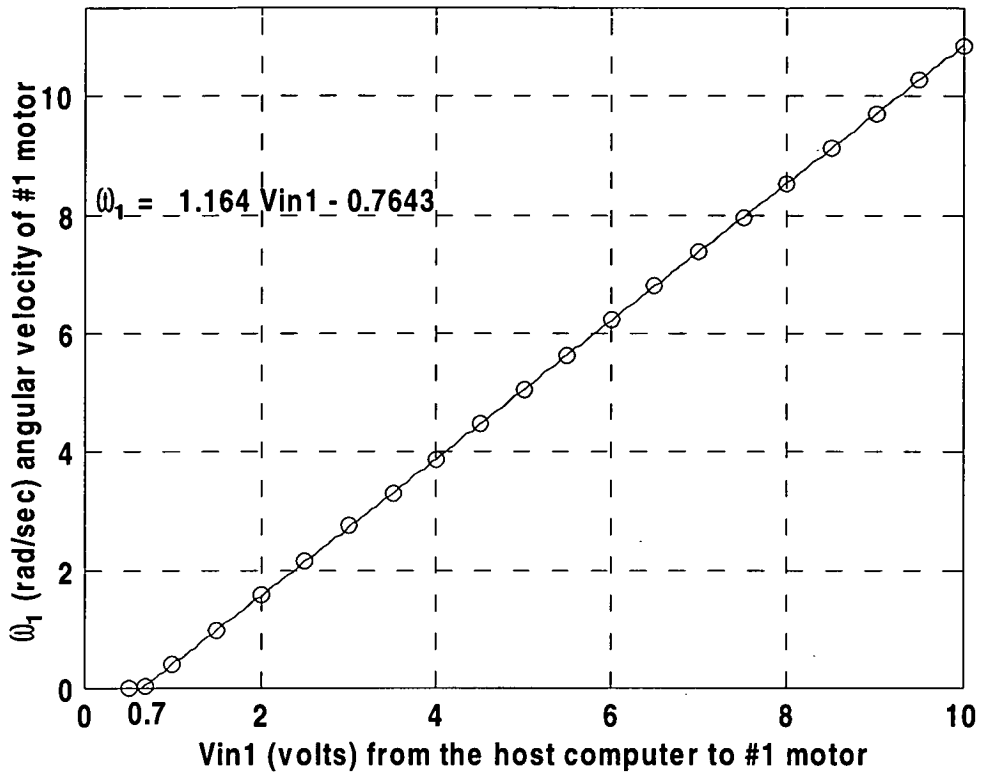


Figure 4.4 The angular speed/input voltage curve of #1 motor

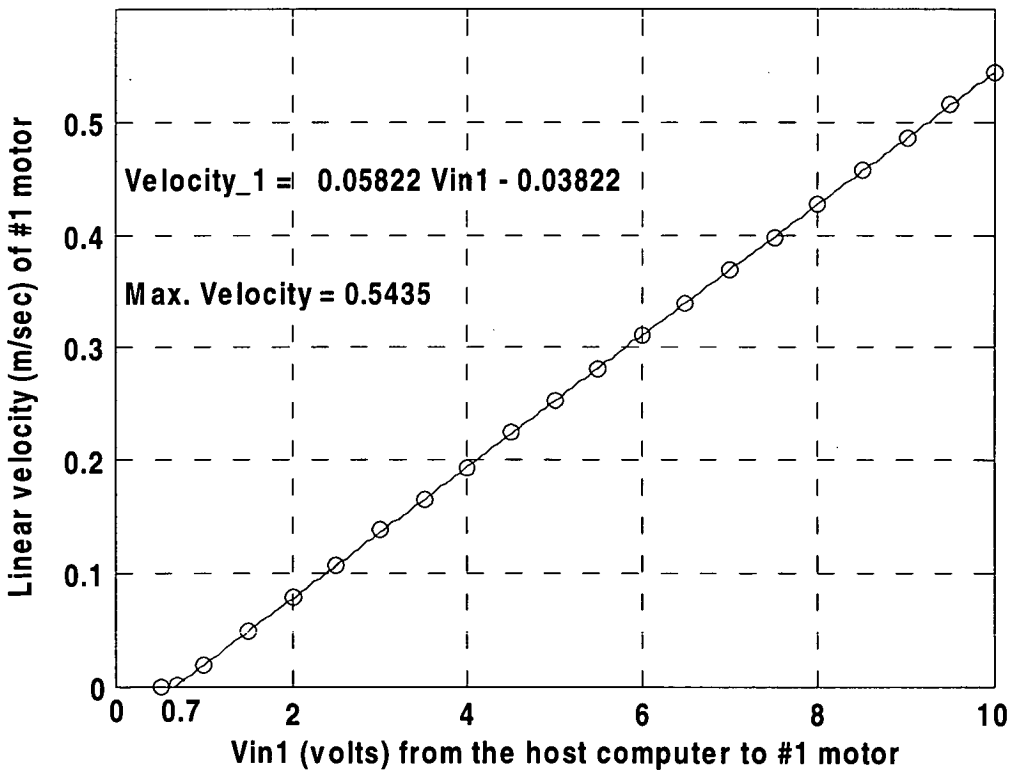


Figure 4.5 The linear velocity/input voltage curve of #1 motor

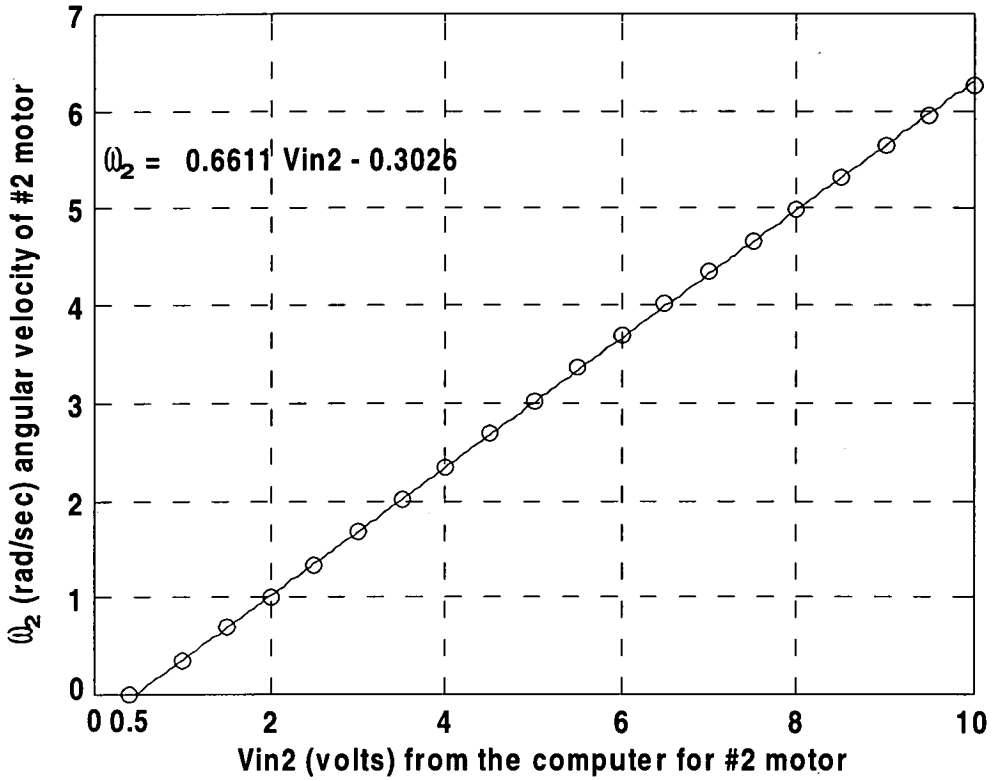


Figure 4.6 The angular speed/input voltage curve of #2 motor

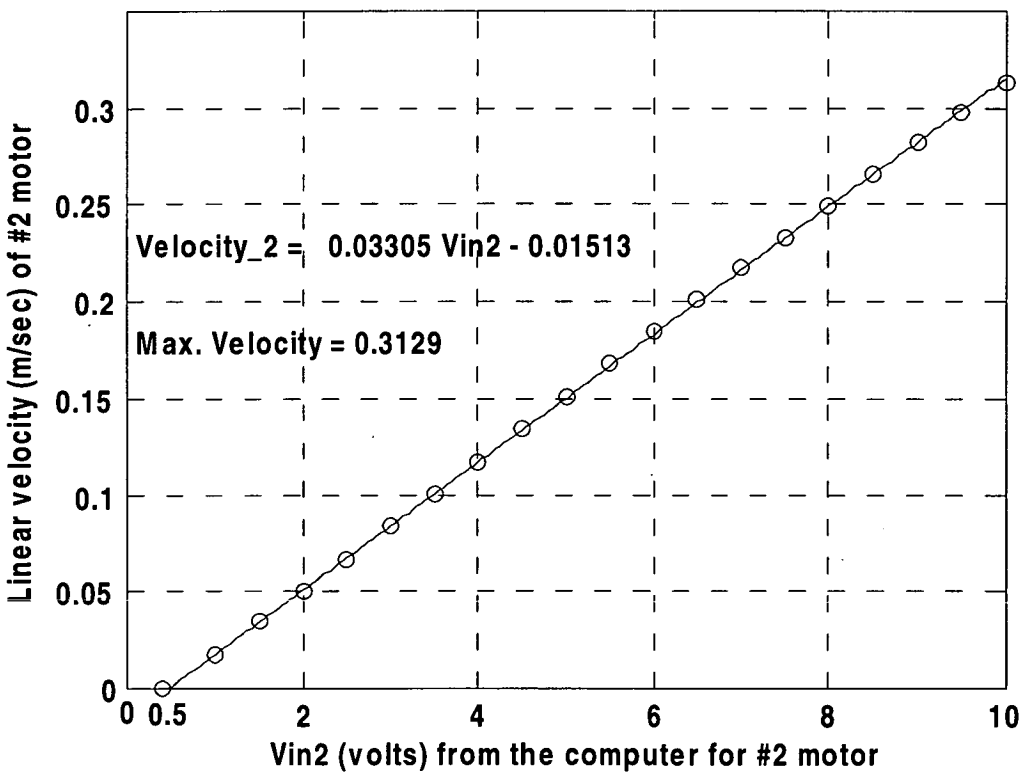


Figure 4.7 The linear velocity/input voltage curve of #2 motor

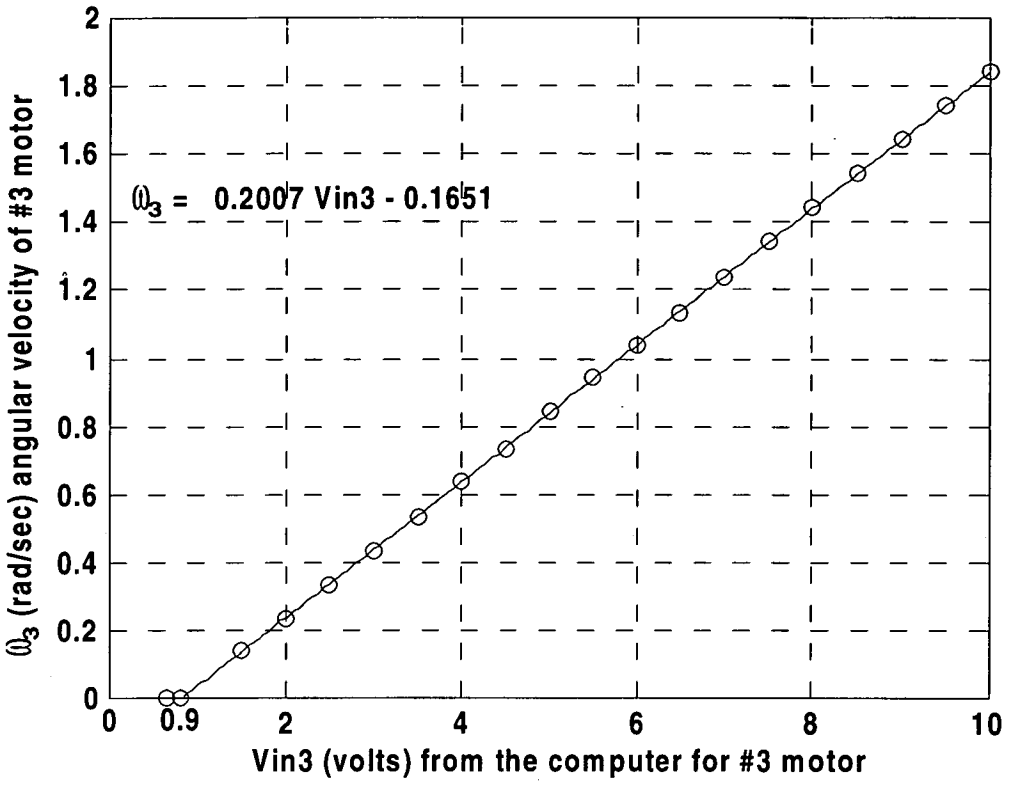


Figure 4.8 The angular speed/input voltage curve of #3 motor

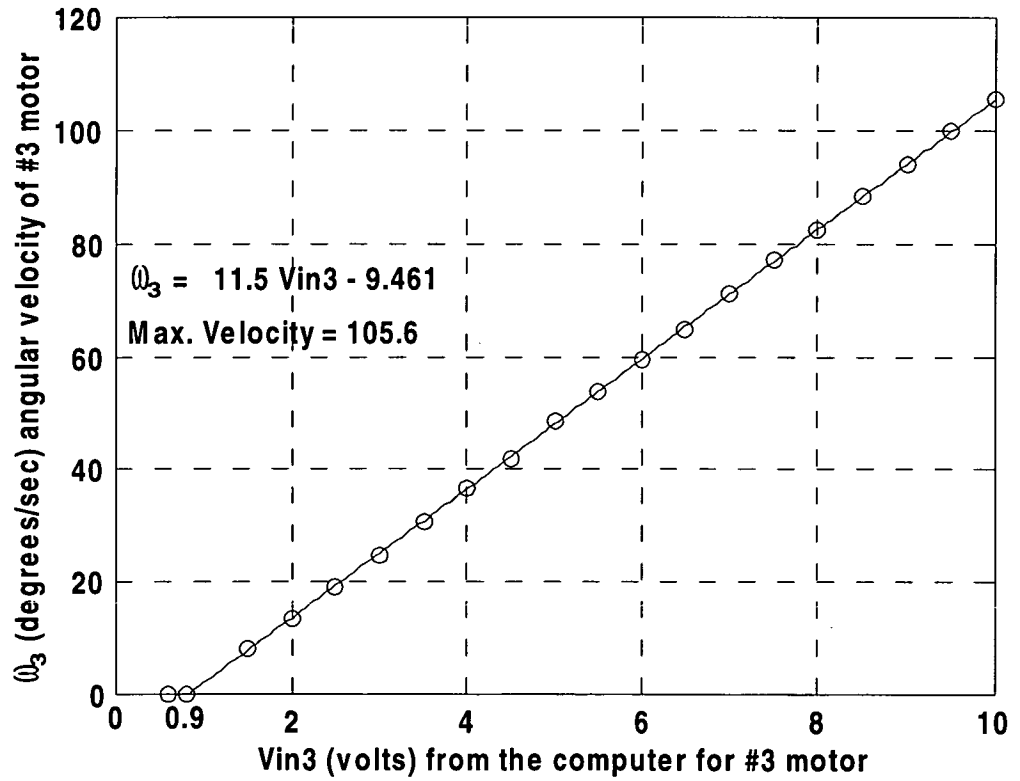


Figure 4.9 The linear velocity/input voltage curve of #3 motor

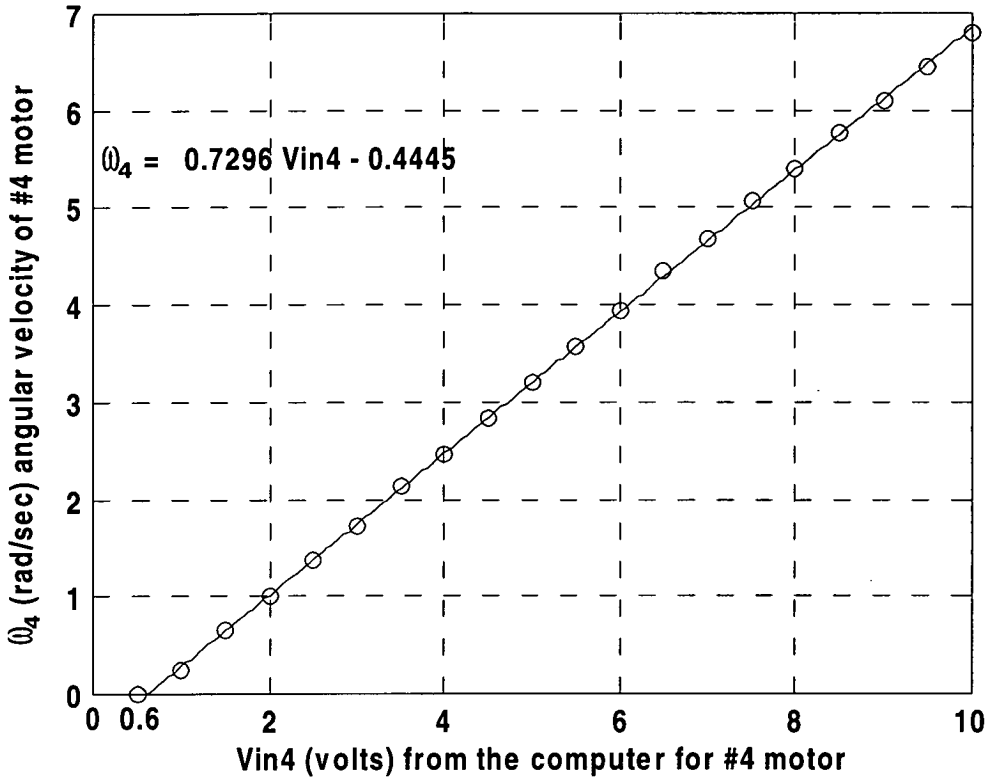


Figure 4.10 The angular speed/input voltage curve of #4 motor

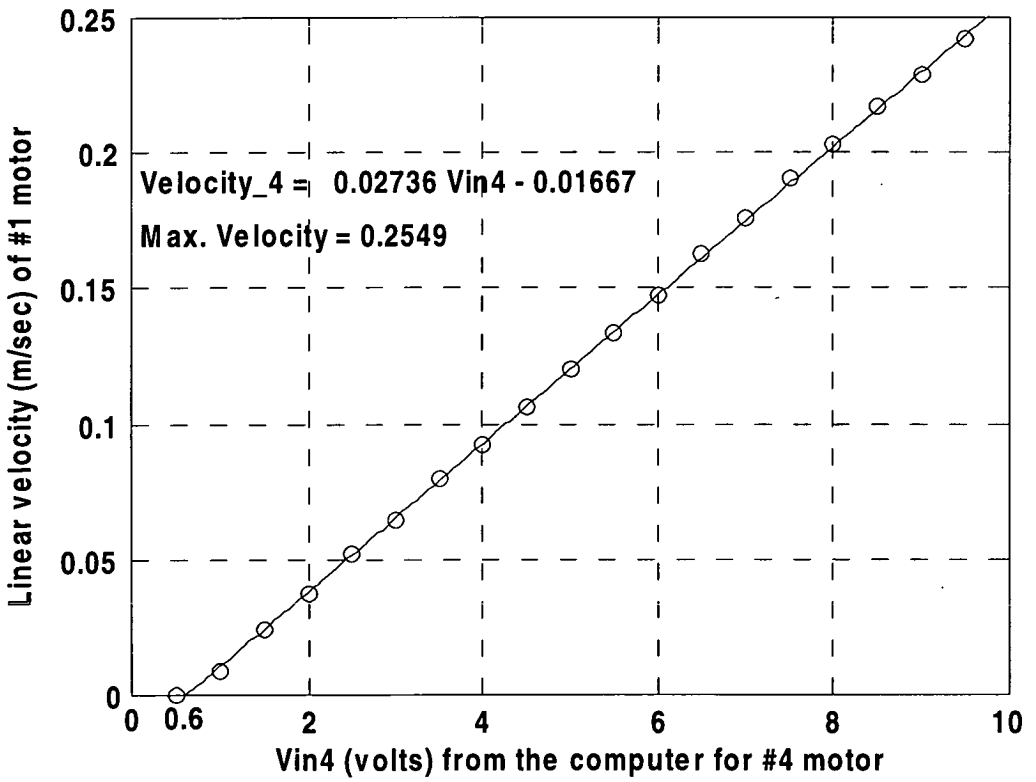


Figure 4.11 The linear velocity/input voltage curve of #4 motor

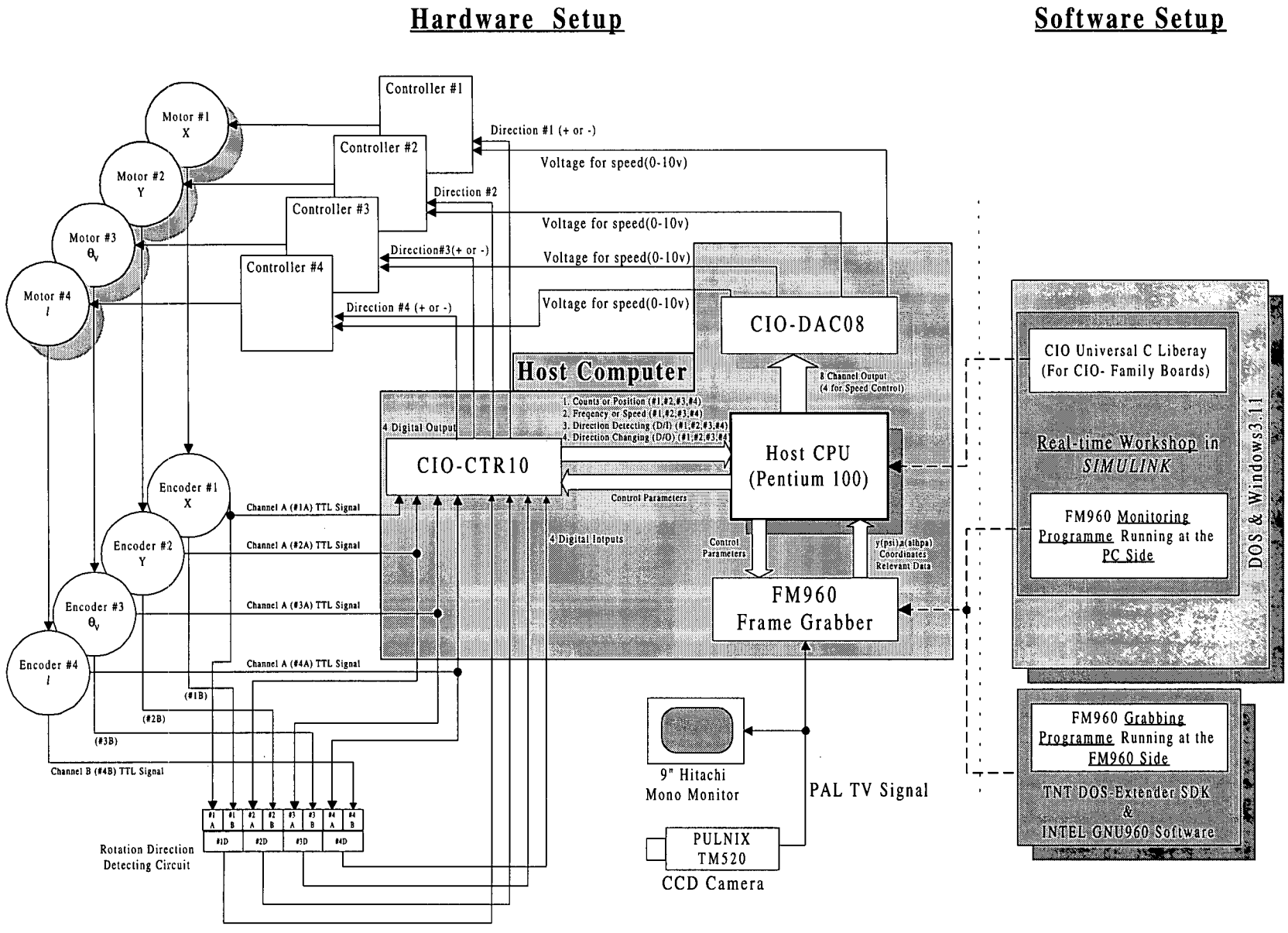


Figure 4.12 The hardware/software setup in the experimental rig

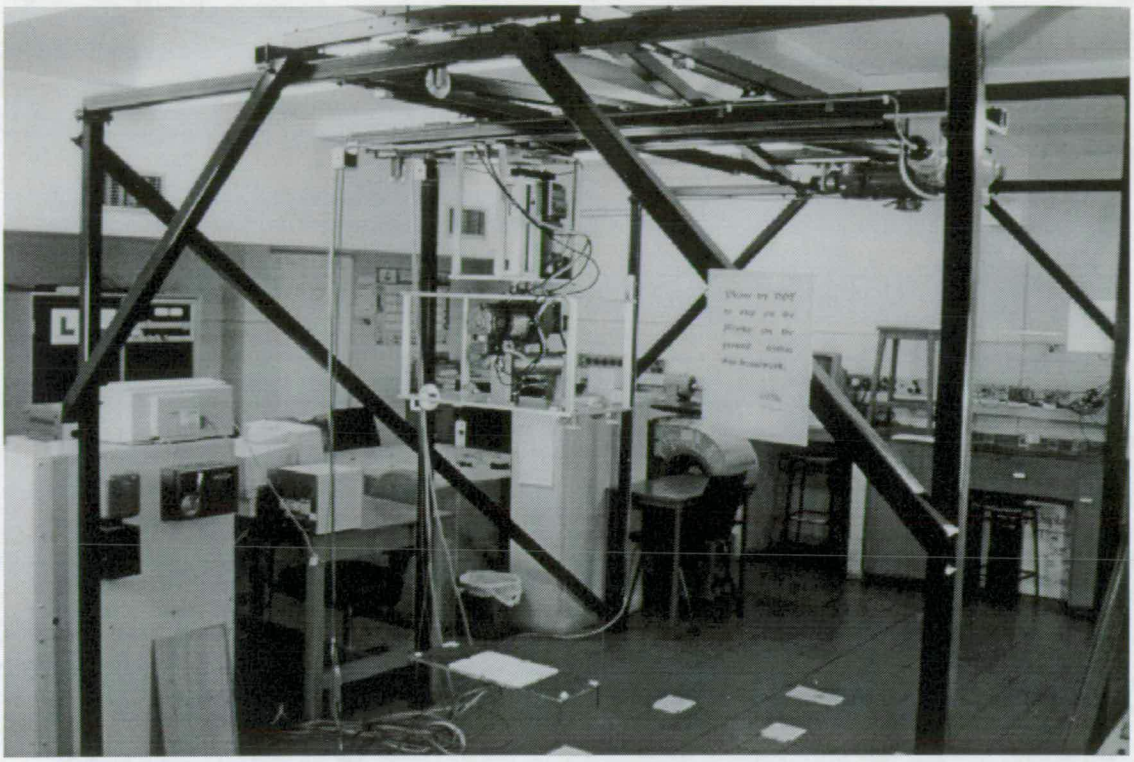


Figure 4.13 A view of the experimental rig.

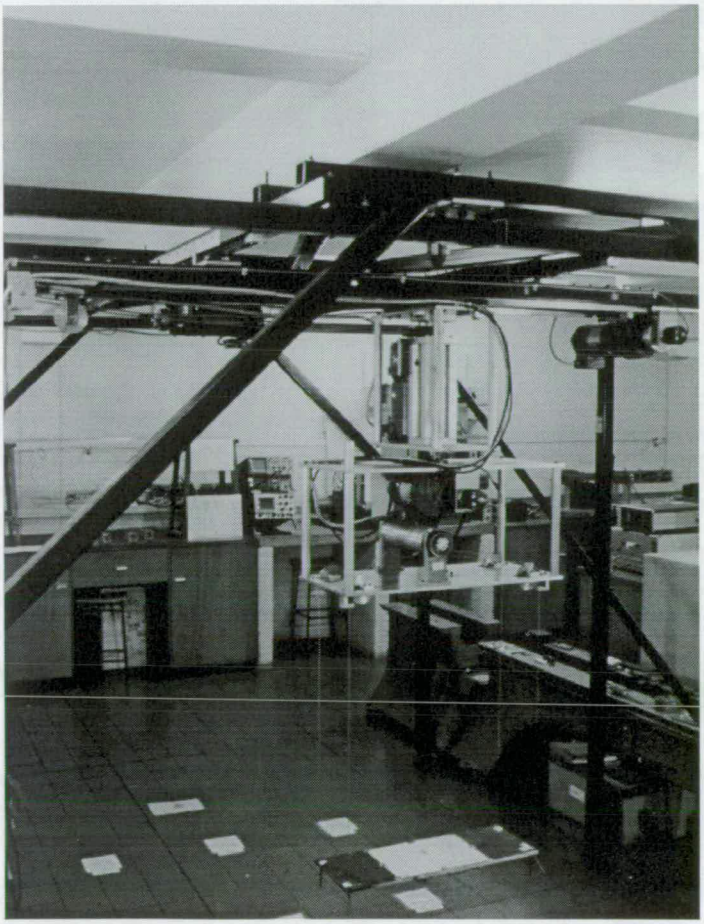


Figure 4.14 Another view of the experimental rig.

Figure 4.15 A side view of the experimental rig.

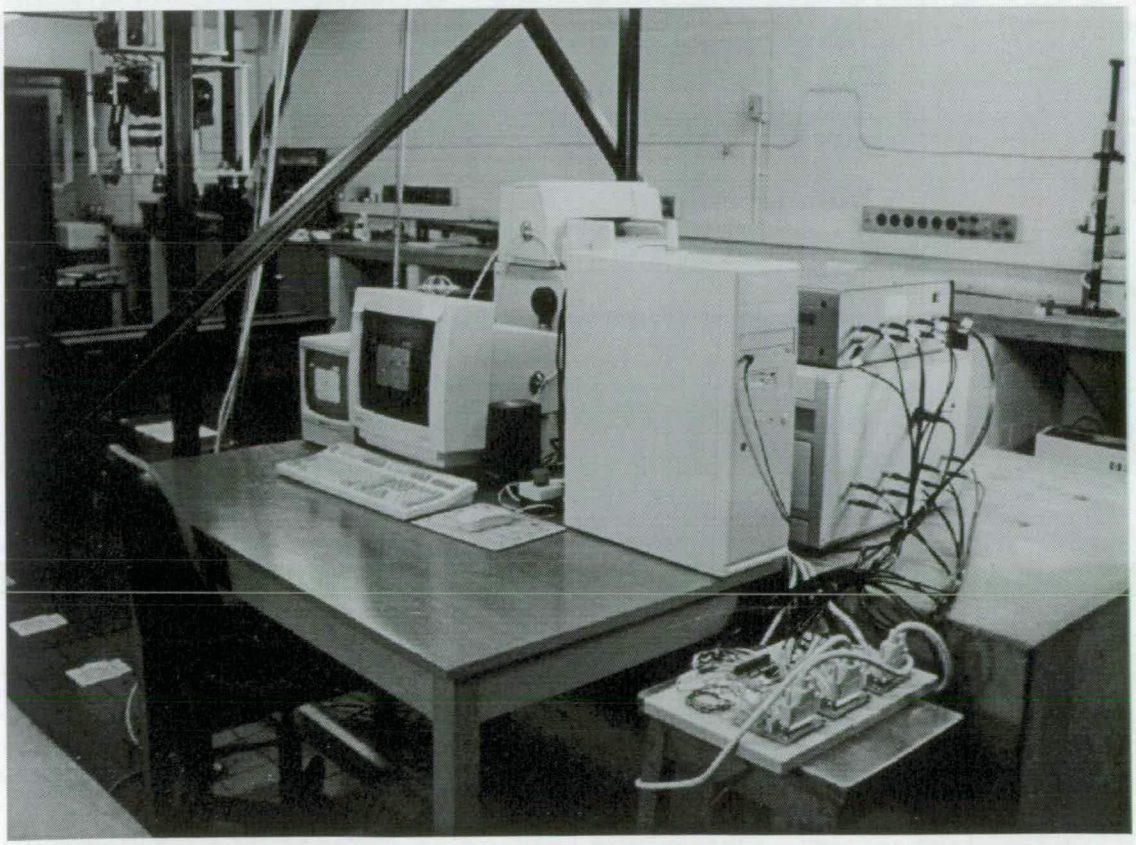


Figure 4.16 The host computer and the motor drive and sensor circuitry

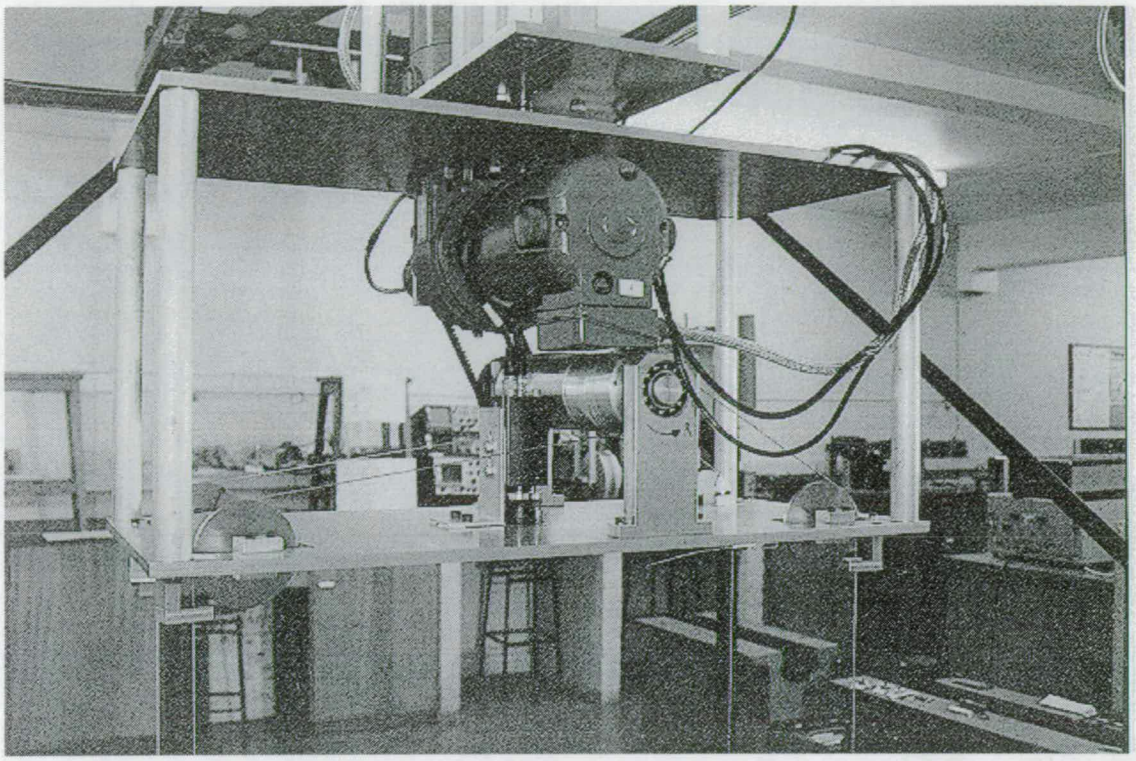


Figure 4.17 The CCD camera mounted on the trolley

Chapter 5

The Visual Sensing System Implementation

5.1 Introduction

As is true for all higher animals, a *vision* capability endows human beings with a sophisticated sensing mechanism that allows them to respond to their environment in an intelligent and flexible manner. Similarly, while proximity, tactile and force sensing play a significant role in the improvement of machine and robot performance, vision is also recognized as the most powerful among these sensing technologies (Fu [32], Harashima [36]).

Machine or *computer vision* may be defined as the process of *extracting, characterizing*, and *interpreting* information from the images of a three-dimensional world. *Extraction* is the process that yields a digitized image, in which preprocessing technologies may be employed to reduce the noise of the incoming signals. *Characterization* deals with the computation of features, such as size or shape suitable for differentiating one type of object from another, or the background. Finally, *interpretation* is the process that recognizes the object and assigns meaning to it. Research using visual sensory systems can be found in the field of control and tracking of robots and AGVs (automatic guided vehicles) (Bishop [7], Feddema [29], Maravall [49]). The camera may either be mounted on the end-effector of the manipulator (also known as the eye-in-hand configuration) or it may be statically located. In both these instances the traditional approach to visual servoing has been to de-couple the problem of obtaining information about the target (from the camera image) from that of manipulator control. This approach is known as the *look-and-move* approach. Initial work in this area has been somewhat hampered by the ready availability of high computational power, so it can be appreciated that a manipulator would be expected to look, and *then* move in a physical experiment. Recent advances in computers have improved the general availability of computational power so that slow look-and-move

responses are not now very obvious in physical experiments, even though the basic philosophy of separating the vision and manipulator control problems still persists. However, it is still considered difficult to provide the necessary operational bandwidth with current off-the shelf image processing equipment for real-time applications, such as the experimental crane rig. For example, if typical CCIR monochrome images consisting of 512 × 512 pixels in 8-bit (1 byte) gray-scale are passed to image processing hardware, there will be a data rate of (512×512×1)×25 = 6,553,600 bytes/second which need to be processed. Therefore it is critically important to seek out a solution to increase the data throughput rate and to reduce the amount of time needed to find the required information from this image data.

In section 5.2 the general hardware configuration and software setup of the visual sensing system for the experimental crane rig are described. The *binary image* format is chosen because of its advantages in the real-time application. In section 5.3 search algorithms are proposed for locating the coordinates of the two lighting sources on the spreader from the digitized images.

5.2 The hardware and software configuration for the visual sensing system

Generalized image interpretation is still one of the most challenging and important research areas in robotics and machine intelligence. To date *binary* image processing is more often used in practical computer vision for a variety of applications because it offers advantages such as compactness and simpler processing algorithms, compared to *gray-scale* image processing (Tong [79]).

As mentioned in section 4.4.2, the principal devices used for the visual sensing system on the experimental rig are the CCD camera mounted on the trolley and the framegrabber in the host computer. One single image is scanned by the CCD camera at a frequency of 25 Hz (CCIR standard in the UK) which is known as the *frame rate*. Each frame consists of 625 lines, of which 576 contain image information (Tecalp

[77]). These TV signals are then passed to the framegrabber on the host computer for further processing.

M (rows) × N (columns) pixels can be set up in a configuration file (in Appendix C) for the framegrabber to construct a format for the resolution of the digitized images, as shown in Figure 5.1. These pixels are generated sequentially from *top-left* to *bottom-right* as the TV signals scan. A corresponding coordinate-system **F,** as described in section 2.2.3, related to the position of the M×N pixels, is assigned as in Figure 5.1.

$$F = F(\text{row\_index}, \text{column\_index})$$

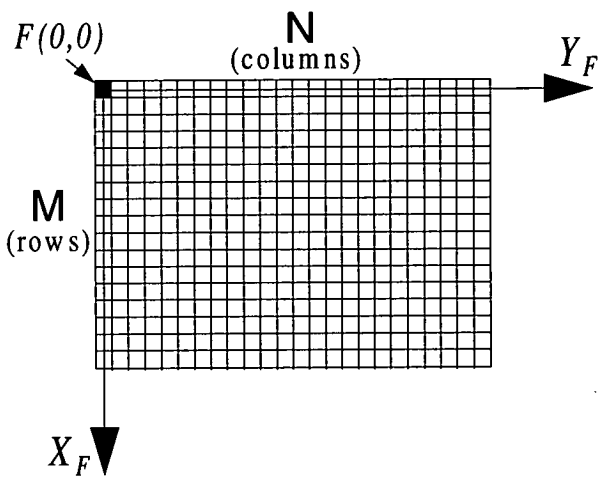


Figure 5.1 M×N pixels on the digitized images

As the analogue-to-digital converter (the ADC) on the framegrabber is an 8-bit device the output value of each pixel can range from 0 (black) to 255 (white). Through a programmable *threshold* (0 ~ 255) binary images can be produced from these digitized pixels. If the value of a pixel is larger than the threshold the final output is **1** (bright). On the contrary, if the value of a pixel is smaller than the threshold the final output is **0** (dark). The reasons for choosing this approach are:

(1) *Compactness*: Because the value of the pixels is either 1 or 0 a *byte* which contains 8 *bits* can store up to 8 pixels. Therefore the memory on the framegrabber can be freed up to store more image processing programs.

(2) *Faster and simpler processing*: The binary value of the pixels presents a simpler form for locating them because all unwanted background objects, such as the floor, the spreader etc., can be filtered out by carefully adjusting the threshold. This avoids complicated image processing algorithms, (such as a Laplacian filter for edge extraction), which are required if higher resolution other than binary mode is needed. The compact size of the stored images also implies a faster throughput rate between different stages of the hardware.

Basically, the structure of the software setup is as in Figure 5.55 at the end of the chapter. First the host computer downloads the on-board programs to the framegrabber. Once it is initialised TV signals from the CCD camera are digitised and then a search algorithm is implemented to find the co-ordinates of the lighting sources (refer to Figure 5.56 at the end of the chapter). The host computer only monitors and retrieves the data stream from the framegrabber, so otherwise it is freed up to execute different tasks.

5.3 The search algorithms for locating the lighting sources on the spreader

As shown in Figure 5.56, two lighting sources appear as two distinct *clusters* (or *groups*) of pixels on the binary images. Therefore the *geometric* centre of the pixels forming the cluster represents the coordinates of the lighting sources. In Figure 5.2 the maximum values of rows and columns in an image, or a *frame,* are termed **FrameHeight** and **FrameWidth**, respectively, and these can only be set up as 32 increments (for example, 480, 512, 544). At present, they are set up for **544** (FrameHeight) × **736** (FrameHeight), this being the maximum resolution that the framegrabber can manage.

There are two ways to adjust the threshold of the visual sensing system: (1) the iris setting on the CCD camera, (2) the software setup in the configuration file downloaded to the framegrabber. It was decided to opt for the second option and to use a fixed iris magnitude. This offers the following advantages:

(1) Increasing the depth of view: The smaller the iris setting, the less light passes through the lens. This reduces the blur of the images and also increases the depth of field (Seaway [69]).

(2) It is easier to use the software setup because there are 256 (0 ~ 255) values which can be precisely chosen in the configuration file. Compared to this, there are only 9 rough scales found on the CCD camera; therefore more accurate adjustment is obtained this way.

Based on these two advantages the iris is set to f/16 which is the minimum magnitude on the CCD camera. At this setting the software threshold can range from *95 ~ 225*. The ambient illumination can produce too much unnecessary reflection if the threshold is set to less than 95. On the other hand there will not be enough light for the visual sensing system if the threshold is above this range.

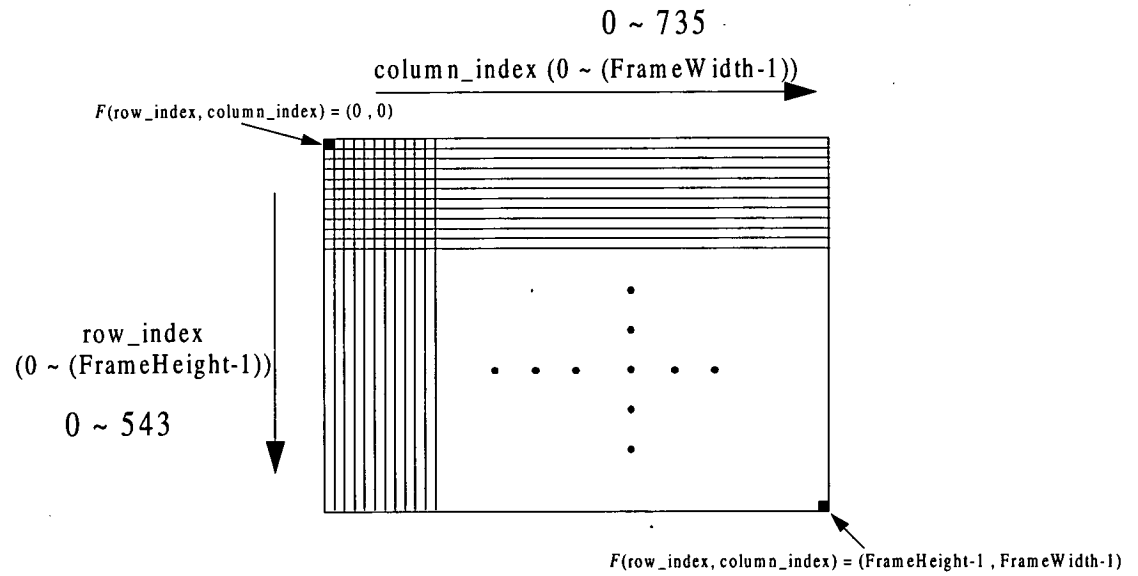


Figure 5.2 The convention of a digitized image (frame)

As shown in Figure 5.57, a mesh map with 544 × 736 grids, similar to Figure 5.2, is affixed on the spreader. This map is then adjusted to coincidence with the axes and the centre of the spreader. With a program capable of displaying 256 gray-scale images on the computer monitor, the spreader is hoisted until the map nearly fills the entire area

(in Figure 5.58). The CCD camera can then be calibrated by adjusting the gaps on the opposite sides equally, as shown in Figure 5.3.

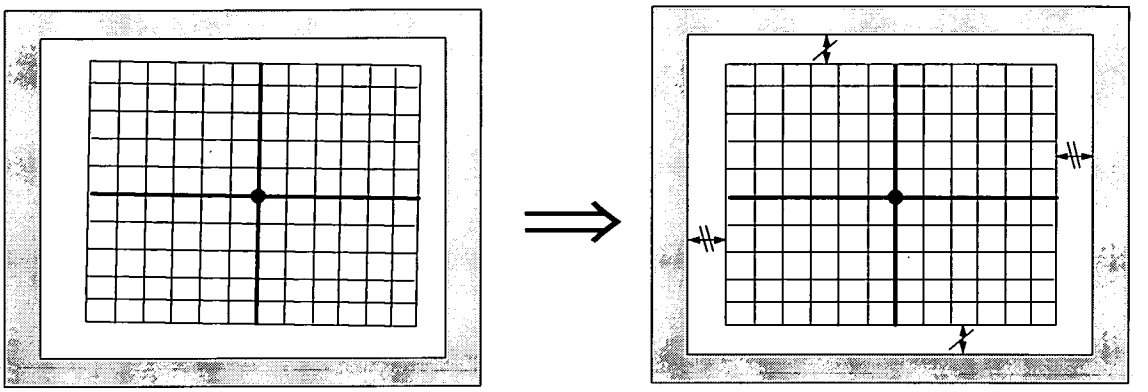


Figure 5.3 Calibrating the CCD camera by adjusting the gaps on the opposite sides equally

5.3.1 The Exhaustive Search Algorithm (ESA)

Firstly only *one* lighting source, located at the centre of the spreader, is activated. The *Exhaustive Search Algorithm (ESA)* works as follows:

(1) Searching a captured frame from top-left to bottom-right pixel by pixel for those whose binary value is **1** (bright).
(2) Summing up their individual coordinates and dividing by the total number counted, to acquire the geometric centre of the lighting sources.

The average processing speed of this algorithm is measured as **4.1** frames/sec. To analyze the performance, the cable length is set at 950 mm, this being the middle point of the range of the *l*- span (700 mm ~ 1200 mm), and the threshold is set at 150. A C program is written to record the results of the tests of the ESA. These results are passed to *MATLAB* for further analysis.

In the first test, the spreader is kept stationary and 30 images are recorded. Their geometric centres (by using the ESA) are plotted in Figure 5.4 at the end of this

section. It can be seen that they are not concentrated on the *reference centre* (271.5, 367.5), which is the central point of the image plane. By plotting a histogram with the number of pixels of these 30 images against the row_index and the column_index (in Figure 5.5 and Figure 5.6, respectively), it appears that in certain *odd* rows, 135, 339, 341, 443 etc. some pixels emerge randomly. If the same algorithm is applied to tests with the cable length equal to 700 mm and 1,200 mm, respectively, similar results can be seen in Figure 5.8 and 5.11. The other test results are summarized in Table 5.1.

The cable length (l)	The average coordinate (row_index, column_index)	Average number of detected pixels
700 mm	(270.7, 369.3)	61.6
950 mm	(263.9, 357.7)	44.5
1,200 mm	(266.4, 355.8)	37.0

Table 5.1 The test results of measuring the central point before noise reduction

These noisy rows appears to be *systematic errors* (Morris [56]) from the framegrabber and consistently emerge during measurement. So, it was then decided to reduce their influence by skipping the *odd* rows in the grabbed images. There are two advantages of doing this,

(1) Most of the noisy rows are odd.
(2) The overall processing speed is increased.

The results are shown in Figure 5.13 to Figure 5.21. The precision has clearly been improved with most of central points of the lighting source concentrated on the same coordinate. The overall processing speed increases from 4.1 frames/second to **4.9** frames/second. The test results are summarized in Table 5.2.

From the coordinates stated in Table 5.2, the position of the CCD camera can be further calibrated by moving the CCD camera along the column axis. The current coordinate of the central lighting source on the spreader is (273.5, 372.3) after adjusting the position of the camera. Also by detecting lighting sources on the axes

(two on the horizontal axis and two on the vertical axis as shown in Figure 5.22.) the alignment of the axes can also be checked and verified.

The cable length (l)	The average coordinate (row_index, column_index)	Average number of detected pixels
700 mm	(272.9, 380.3)	26. 6
950 mm	(273.4, 380.3)	17.0
1,200 mm	(273.7, 380.5)	12.4

Table 5.2 The test results of measuring the central point after noise reduction

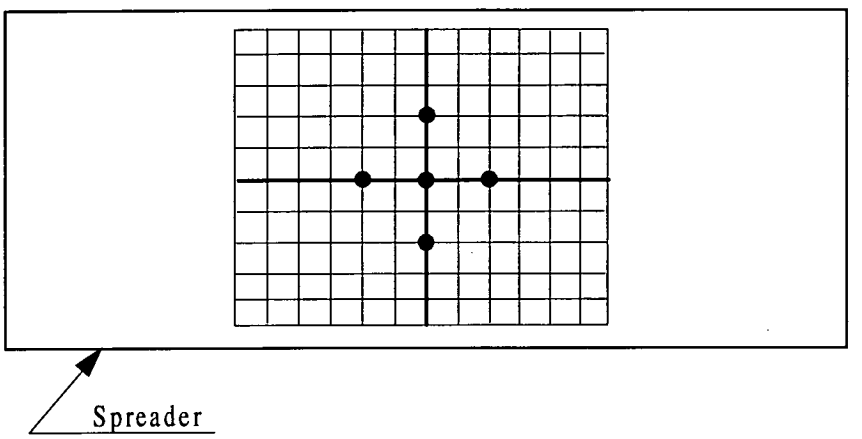


Figure 5.22 The position of lighting sources for further calibration of the CCD camera

Now, if the spreader is moved along the row- axis manually (↓) and released (the coordinates of the central lighting source on the spreader are set to an initial condition of (464, 372)), then the test results are as shown in Figure 5.22, 5.26. 5.28, with cable lengths 700 mm, 950 mm and 1,200 mm, respectively. It can been seen that the performance of the ESA is reasonably good, because the coordinates in the vertical direction (column_index) of most of the pixels stay within a small band of distribution, as shown in Figure 5.23, 5.27. 5.29.

Similarly, Figures 5.24, 5.28, and 5.32 show test results obtained by pulling the spreader along the column- axis (→) (with the coordinates of the central lighting

source set to an initial condition of (272, 564)). The results of these tests are shown in Figure 5.25, 5.29, and 5.33.

Generally speaking, there are three functions carried out the ESA:

(1) To identify noisy lines due to hardware imperfections (systematic errors).
(2) To calibrate the camera.
(3) To test the performance of the framegrabber. The processing speed is 4.9 frames/second with the odds row skipped.

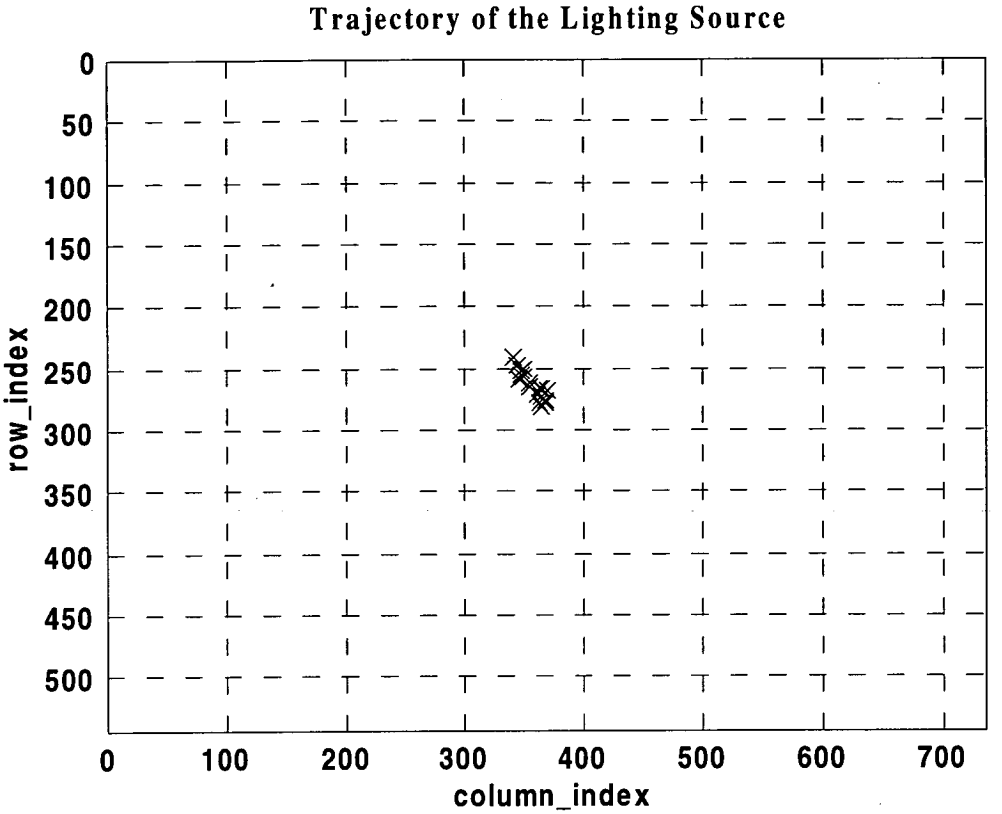


Figure 5.4 (Threshold =150, Cable Length = 950 mm)

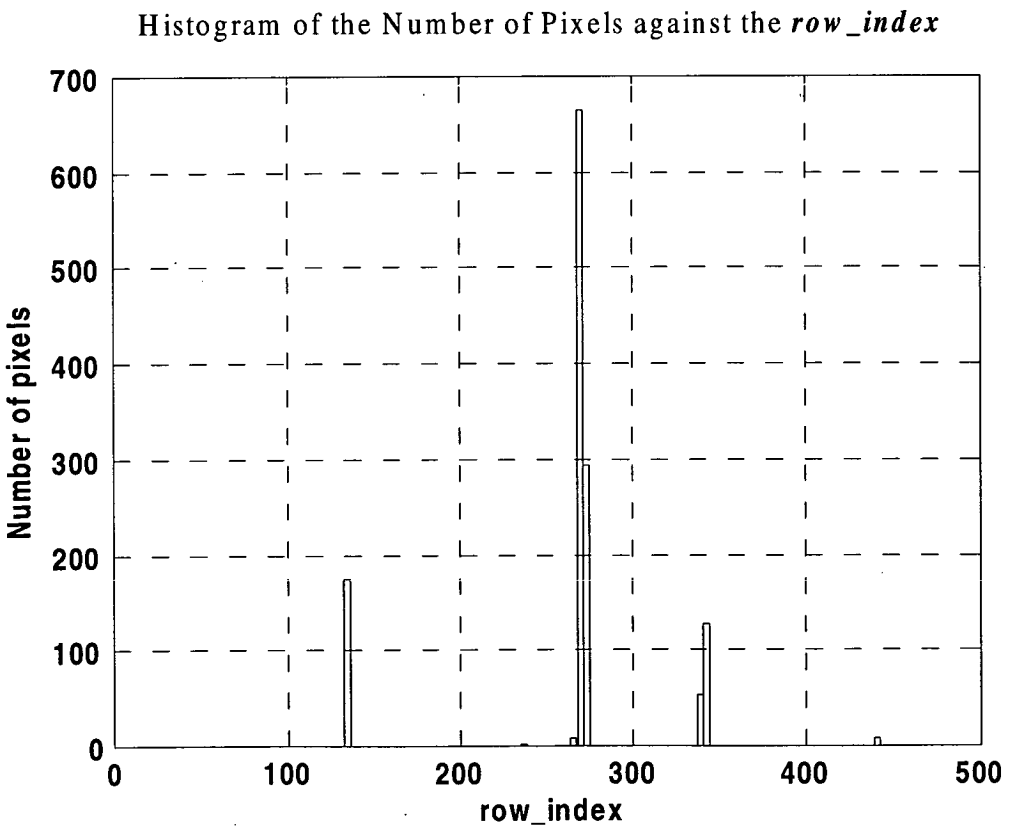


Figure 5.5 (Threshold =150, Cable Length = 950 mm)

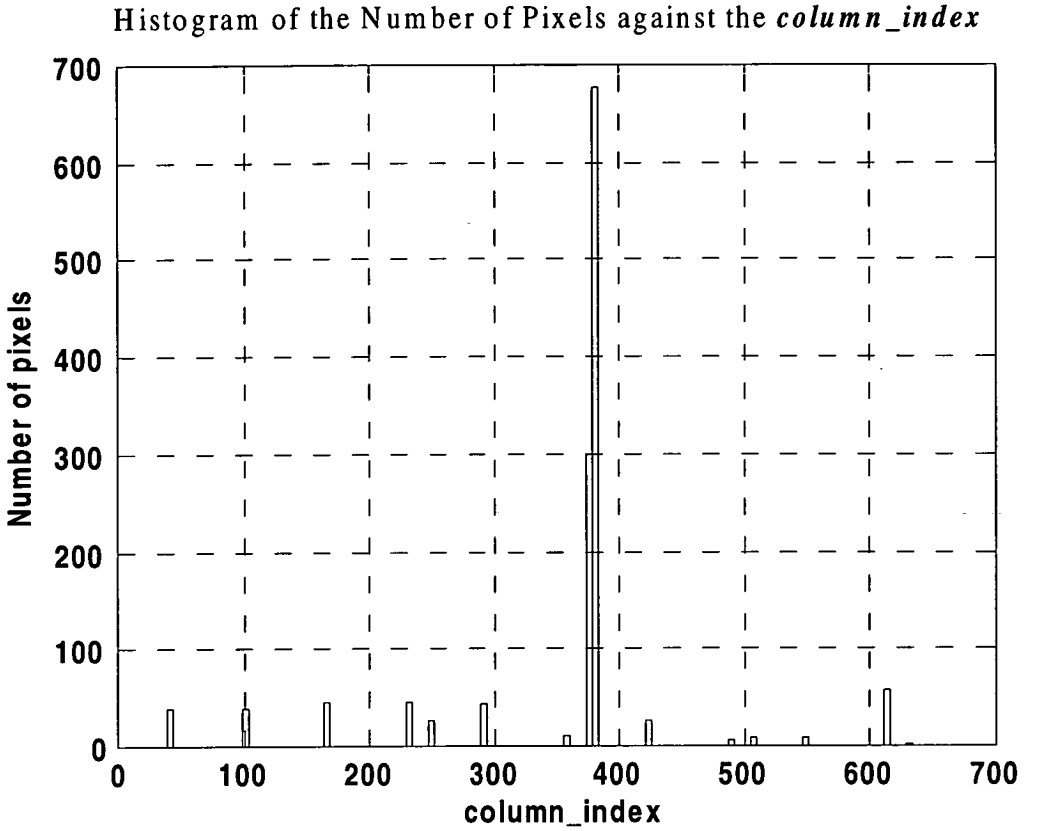


Figure 5.6 (Threshold =150, Cable Length = 950 mm)

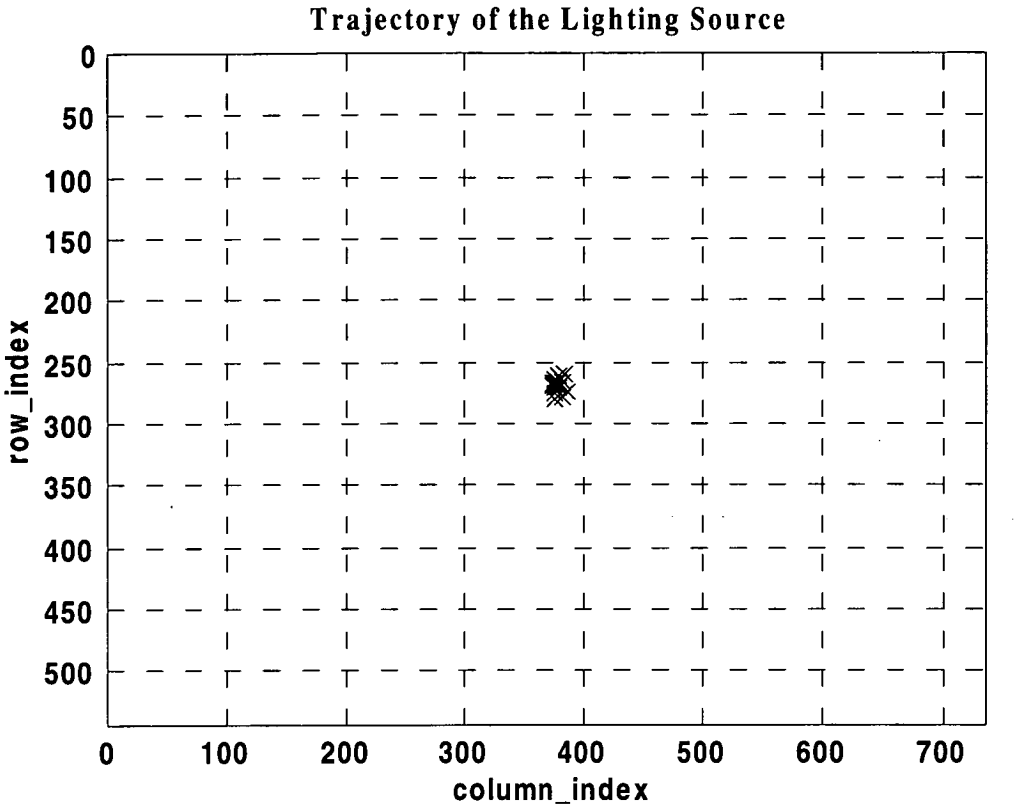


Figure 5.7 (Threshold =150, Cable Length = 700 mm)

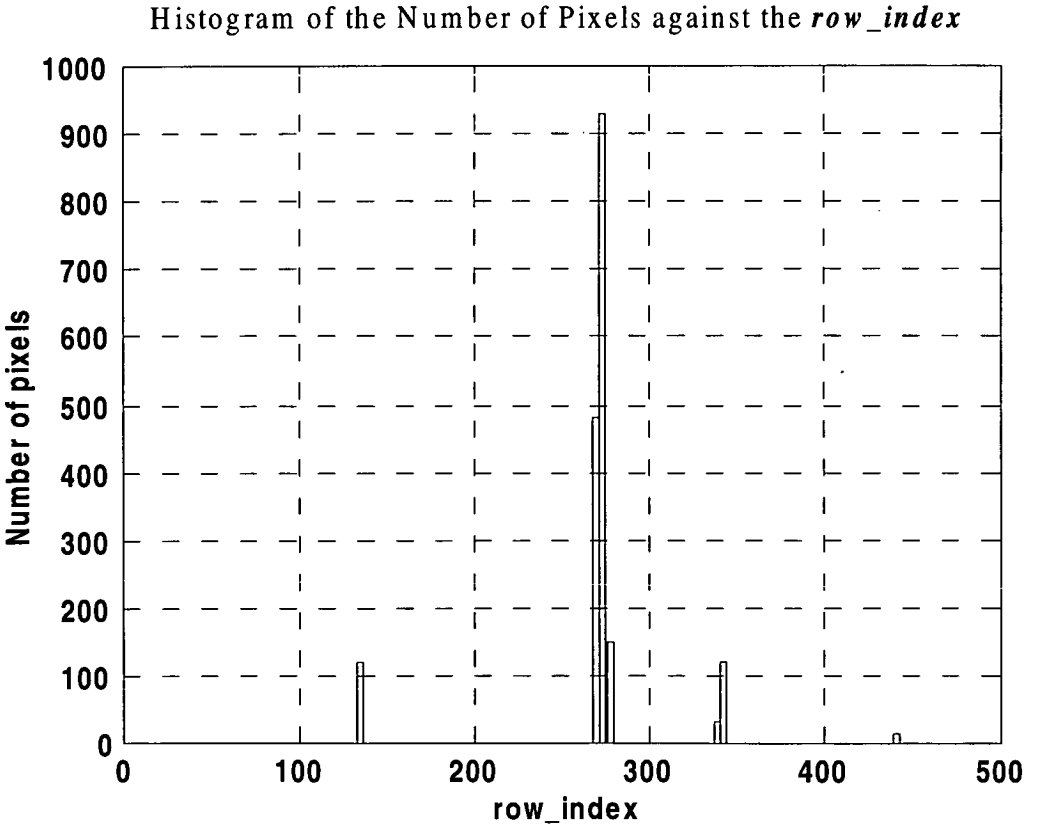


Figure 5.8 (Threshold =150, Cable Length = 700 mm)

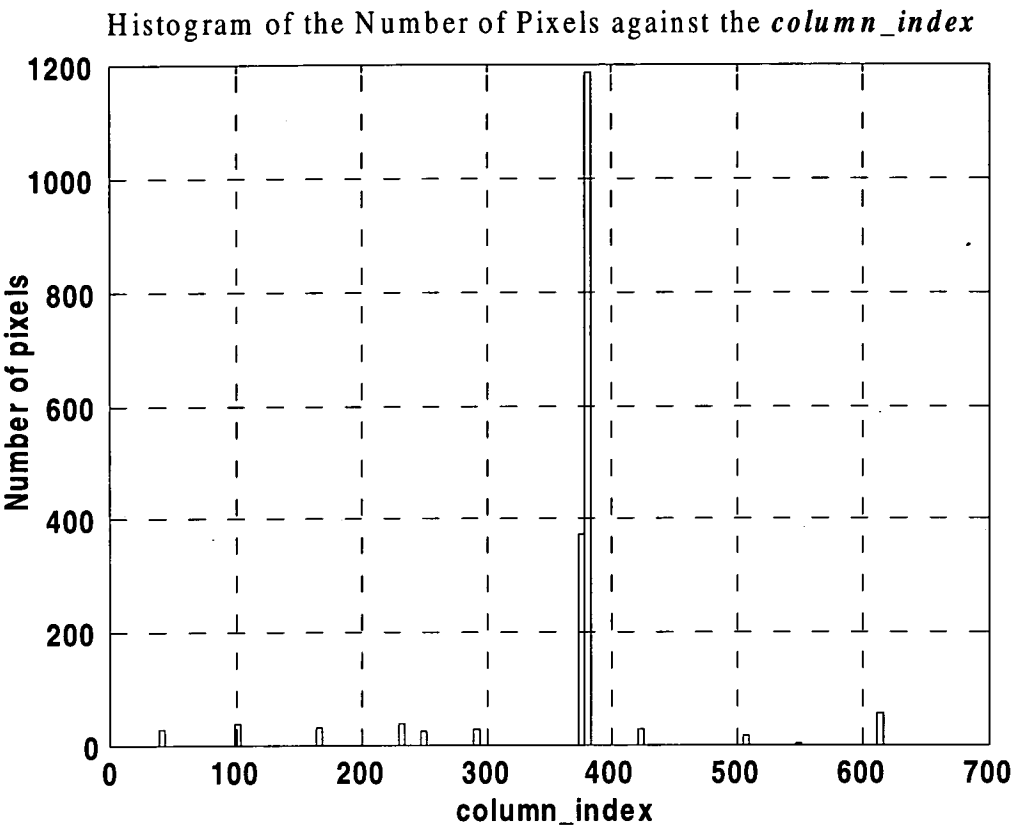


Figure 5.9 (Threshold =150, Cable Length = 700 mm)

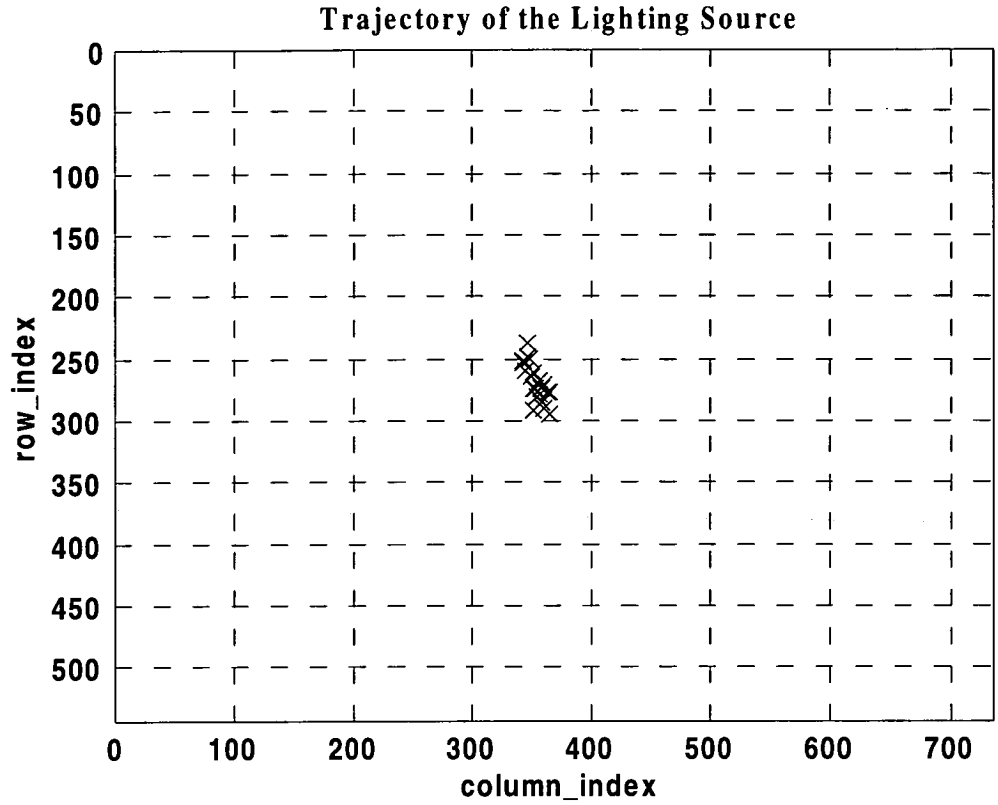


Figure 5.10 (Threshold =150, Cable Length = 1,200 mm)

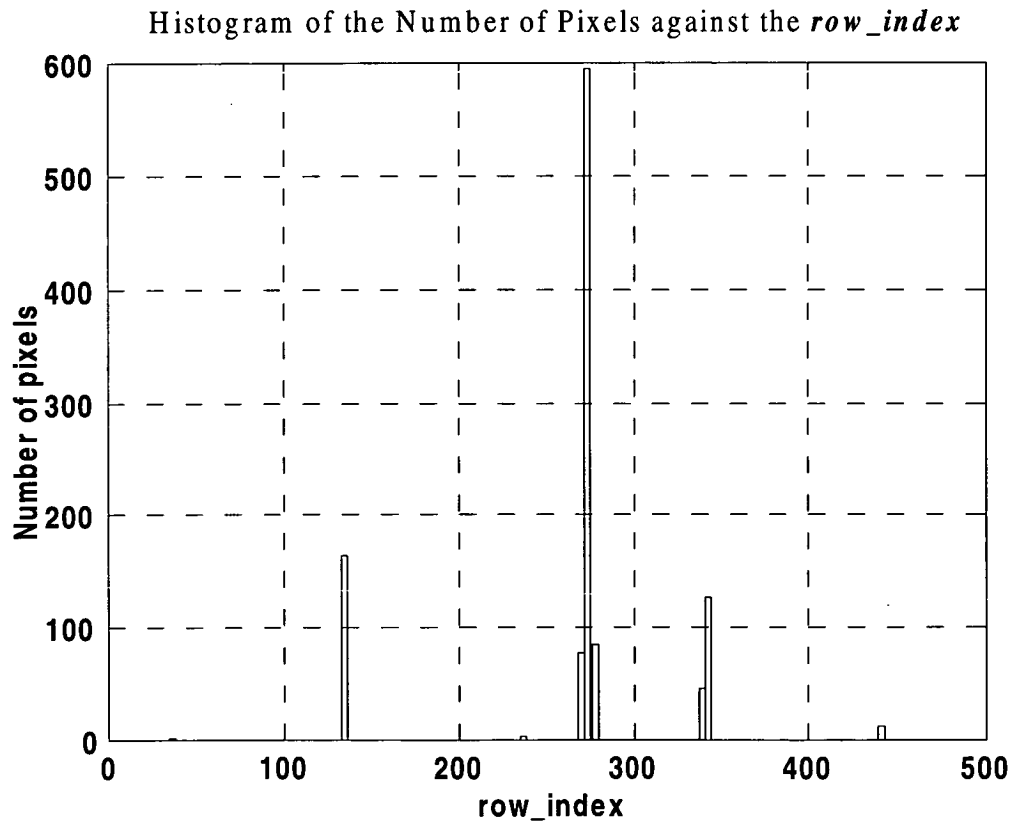


Figure 5.11 (Threshold =150, Cable Length = 1,200 mm)

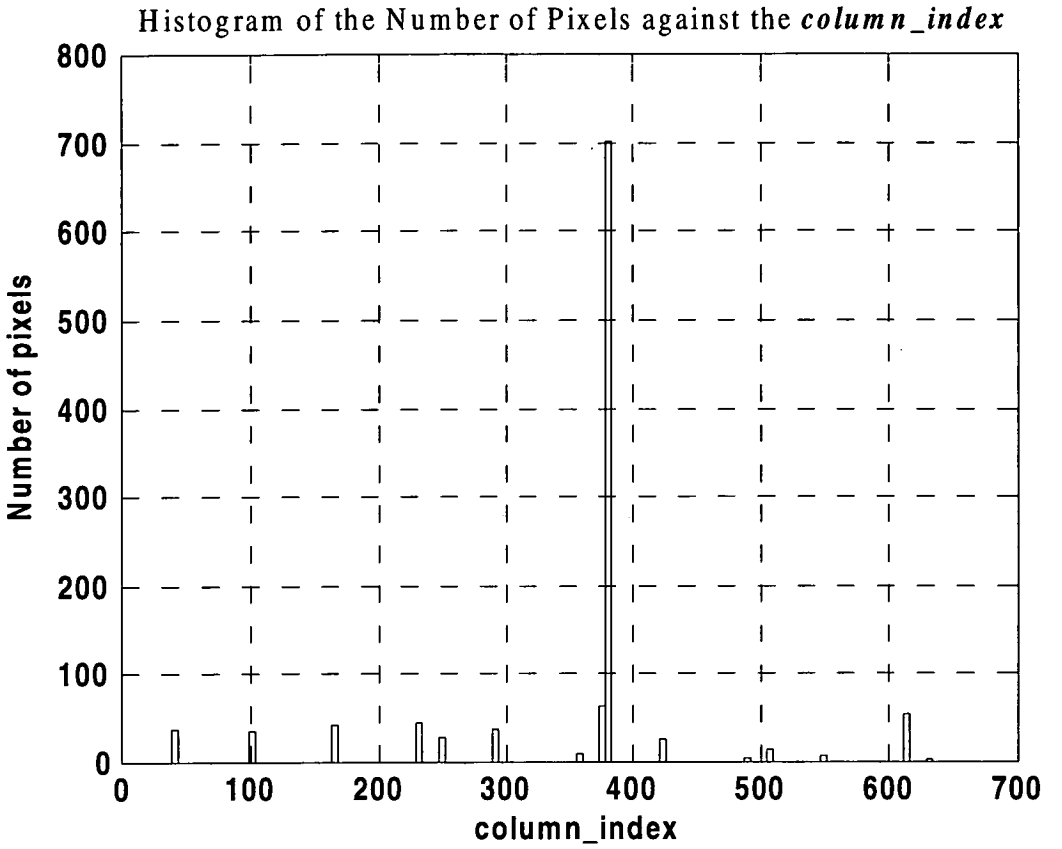


Figure 5.12 (Threshold =150, Cable Length = 1,200 mm)

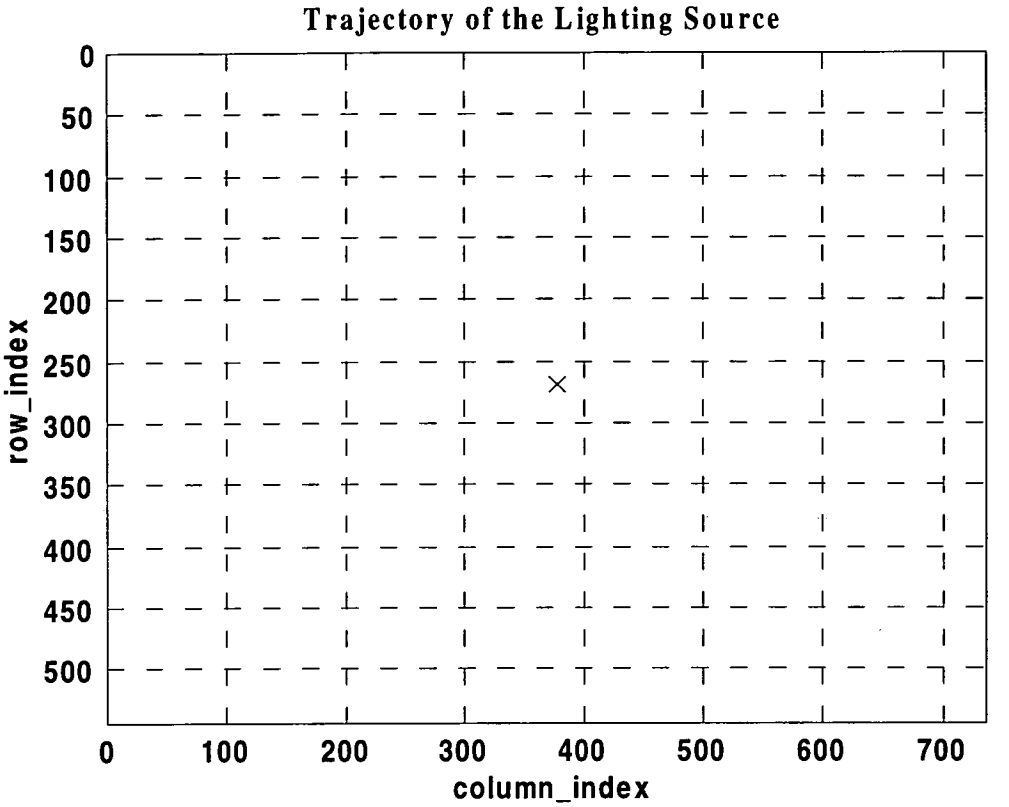


Figure 5.13 (Threshold =150, Cable Length = 700 mm)

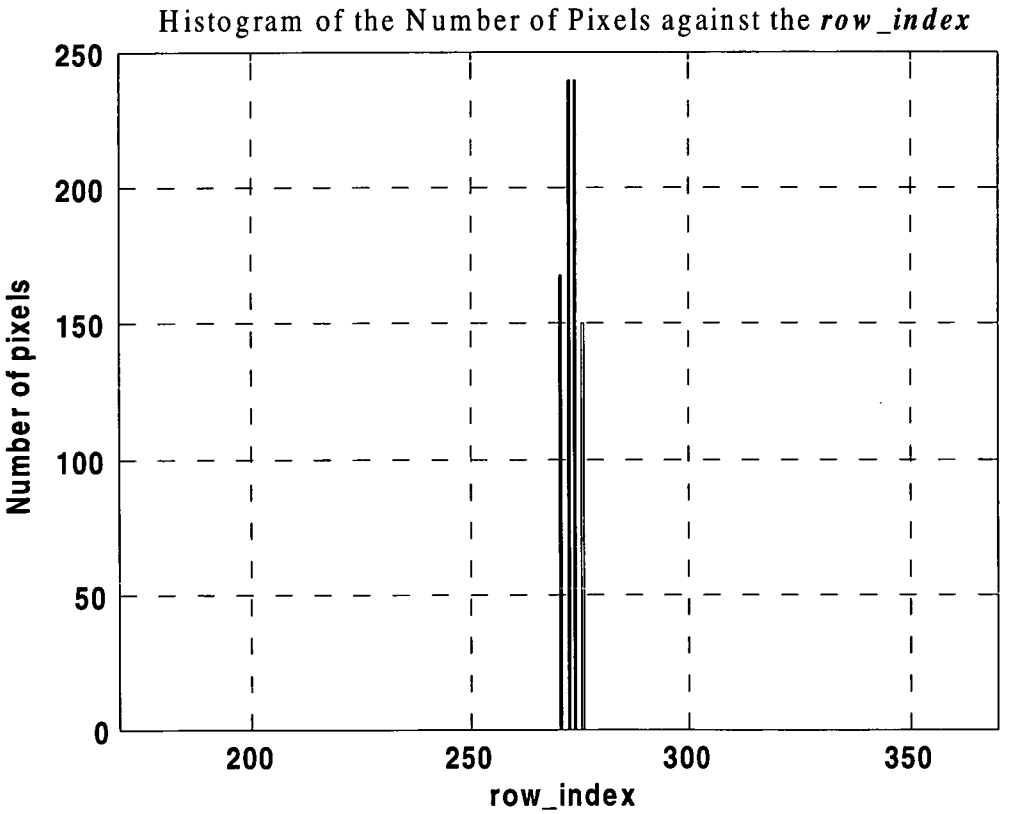


Figure 5.14 (Threshold =150, Cable Length = 700 mm)

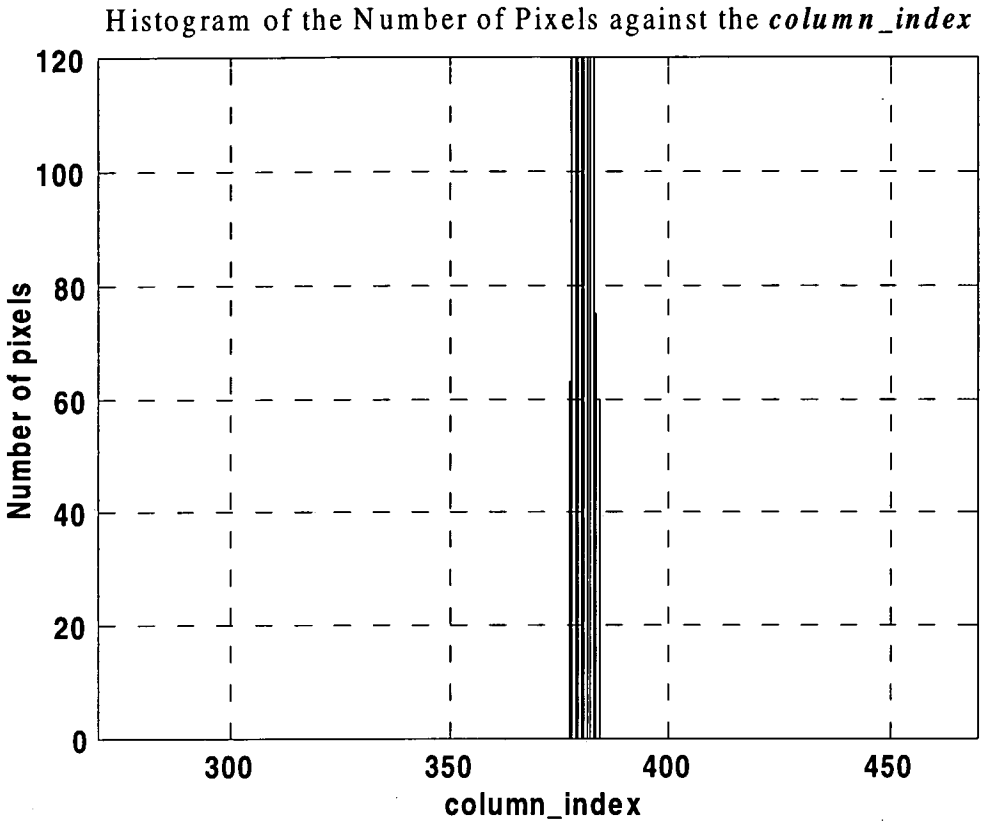


Figure 5.15 (Threshold =150, Cable Length = 700 mm)

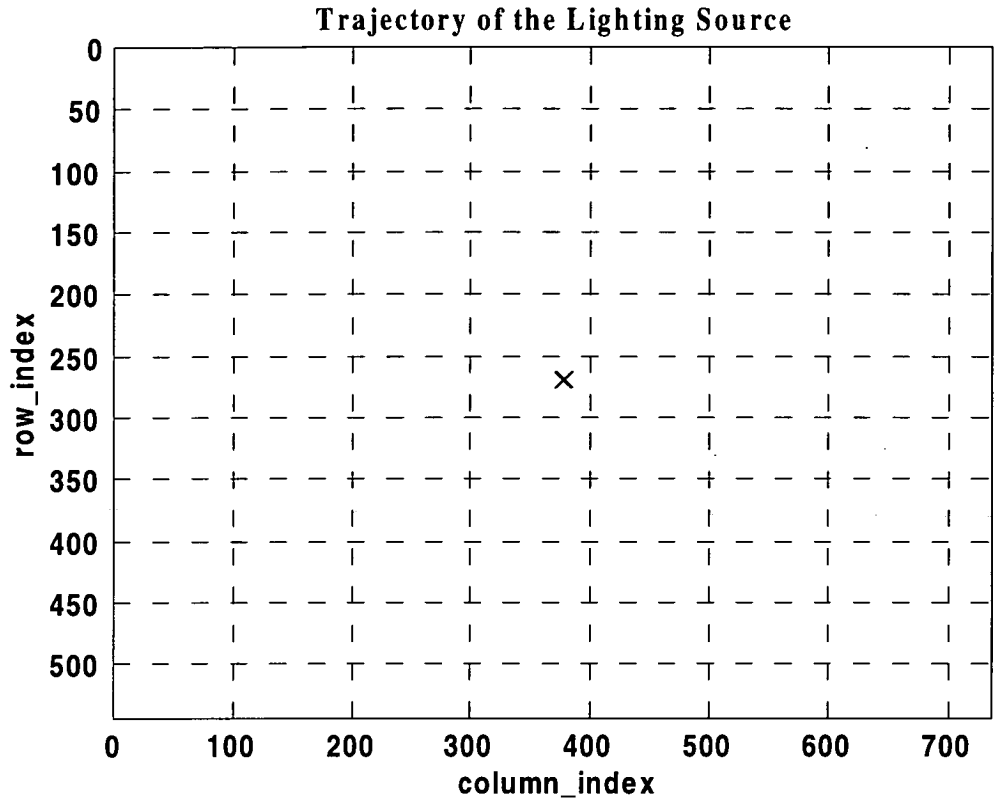


Figure 5.16 (Threshold =150, Cable Length = 950 mm)

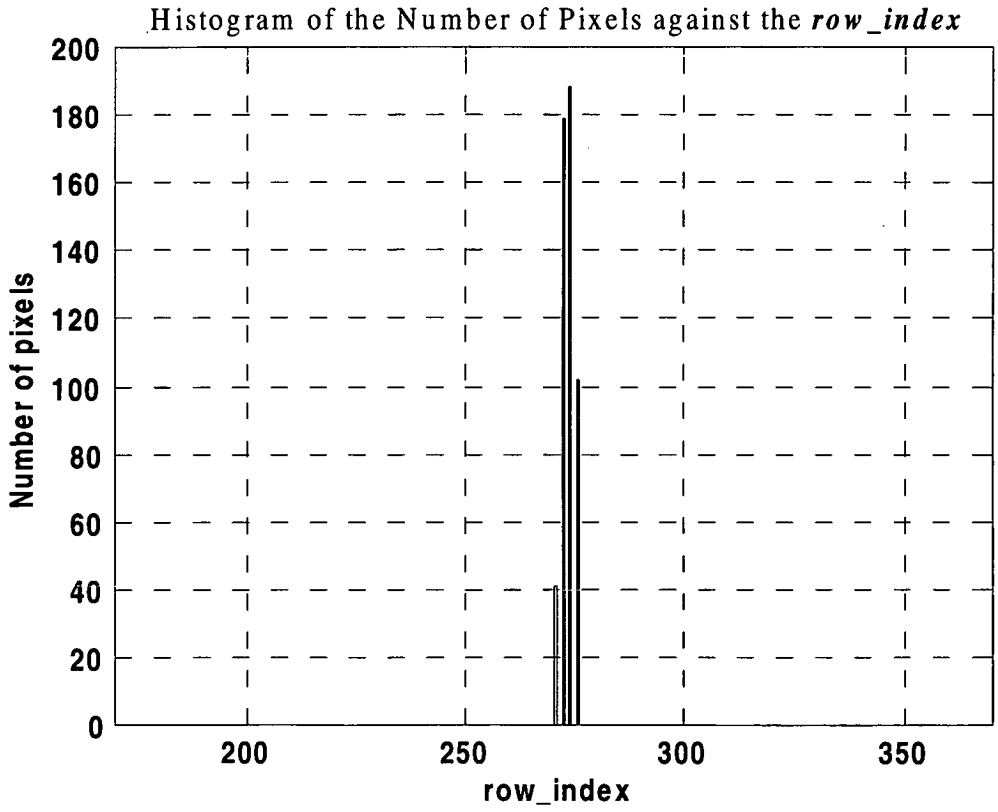


Figure 5.17 (Threshold =150, Cable Length = 950 mm)

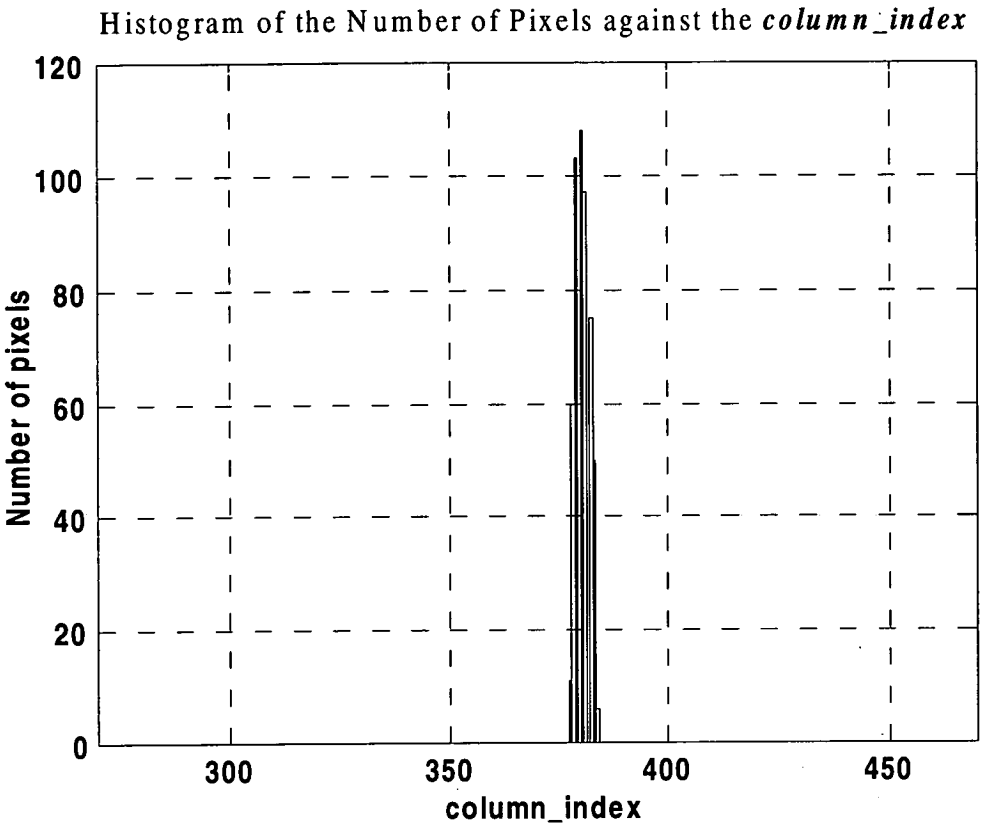


Figure 5.18 (Threshold =150, Cable Length = 950 mm)

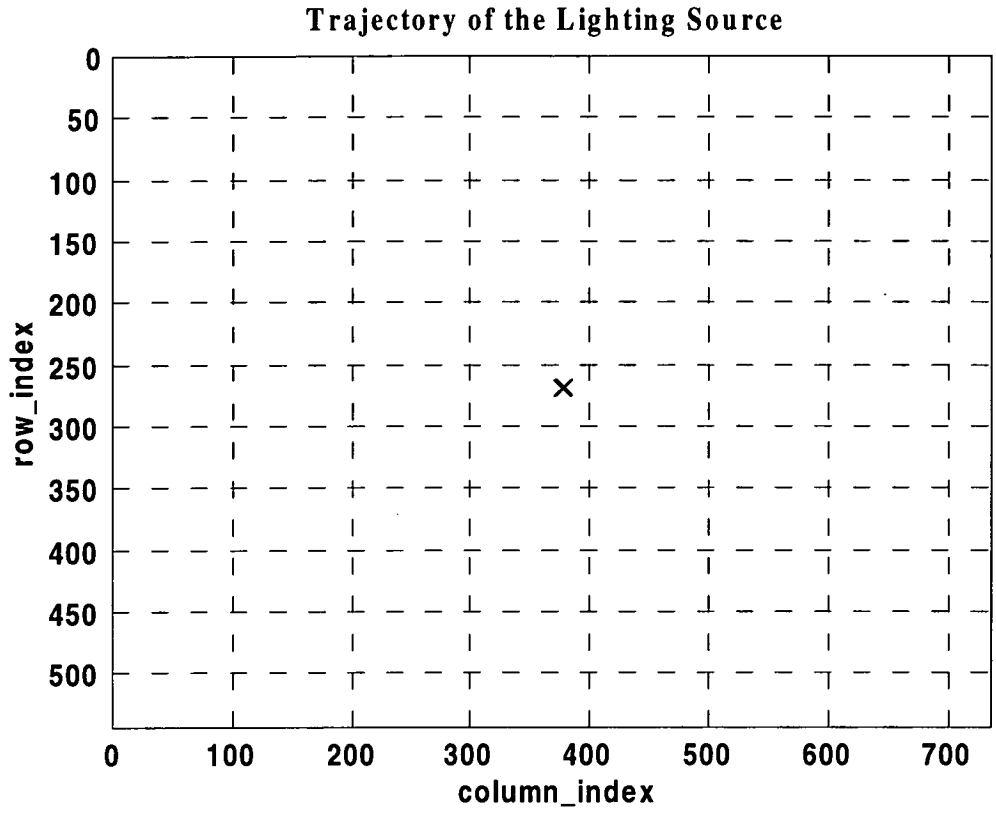


Figure 5.19 (Threshold =150, Cable Length = 1,200 mm)

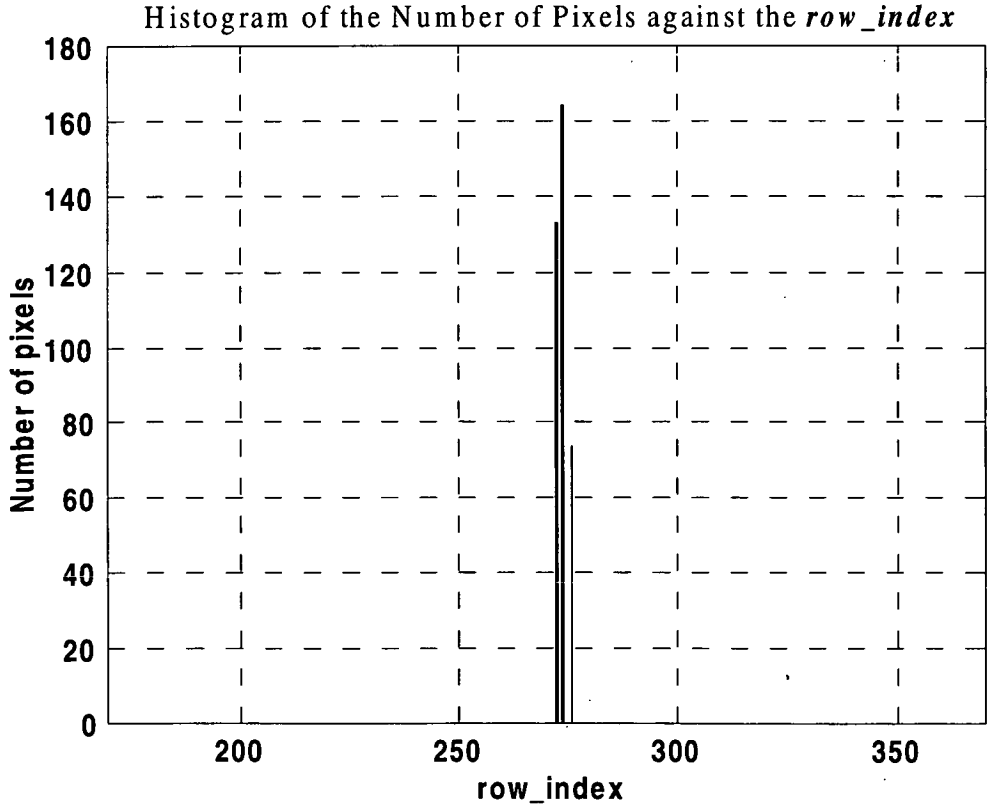


Figure 5.20 (Threshold =150, Cable Length = 1,200 mm)

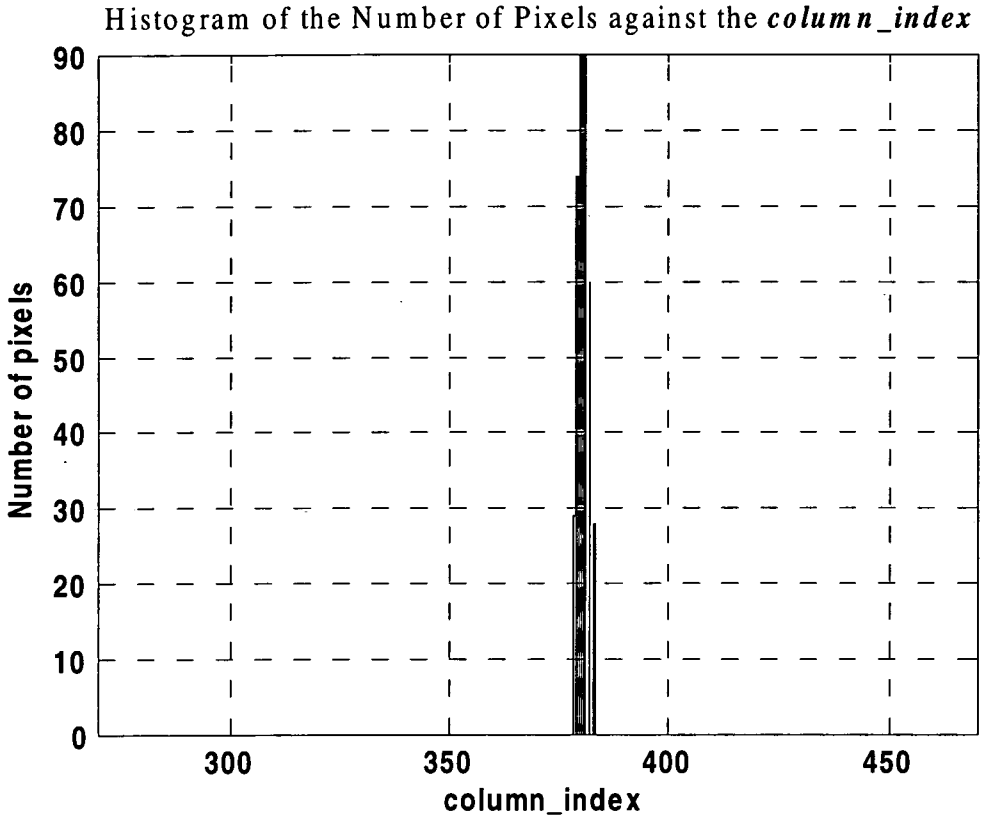


Figure 5.21 (Threshold =150, Cable Length = 1,200 mm)

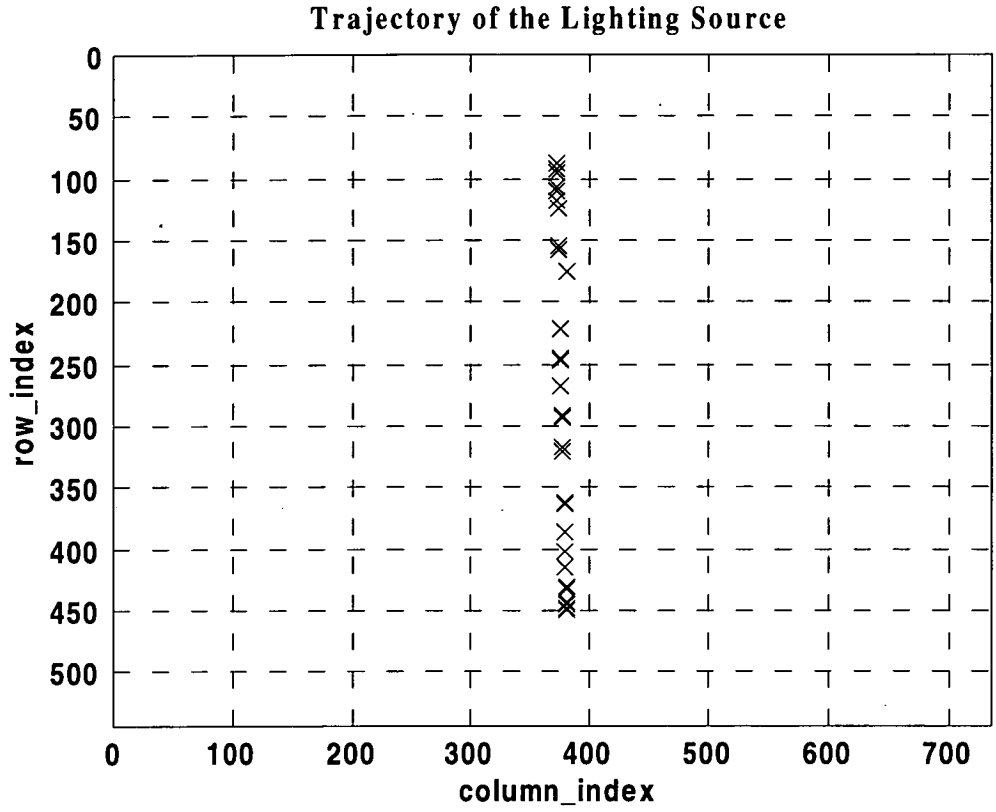


Figure 5.22 (The average number of pixels in each grabbed image: 38.1)
(Threshold =150, Cable Length = 700 mm)

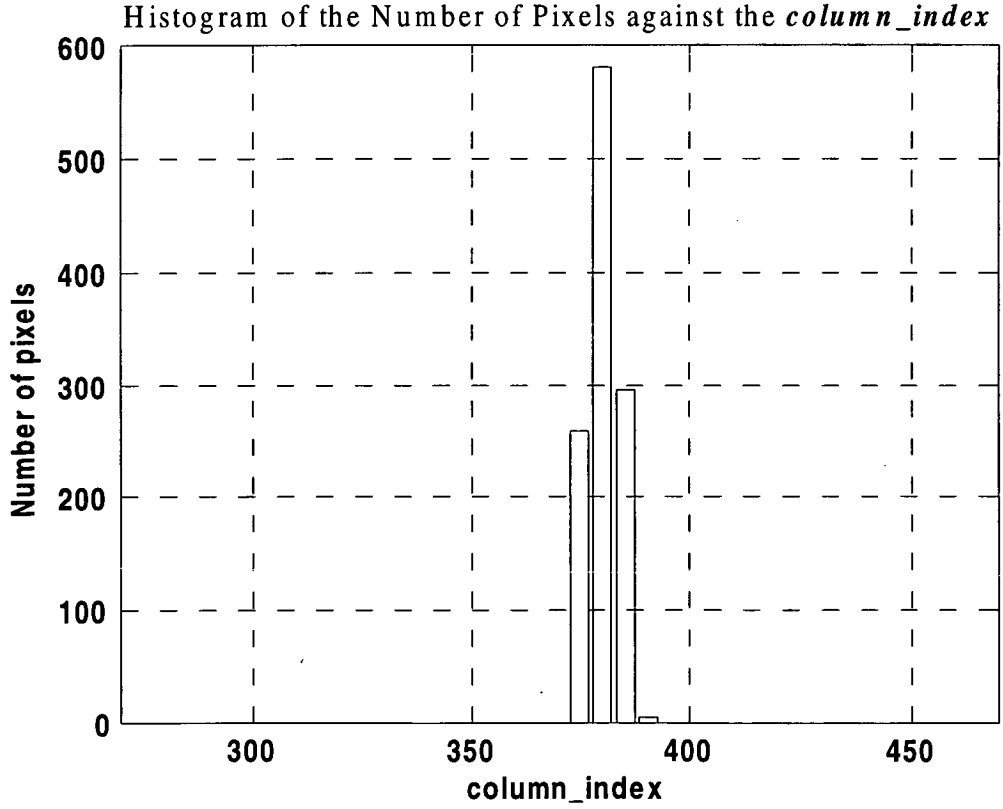


Figure 5.23 (Threshold =150, Cable Length = 700 mm)

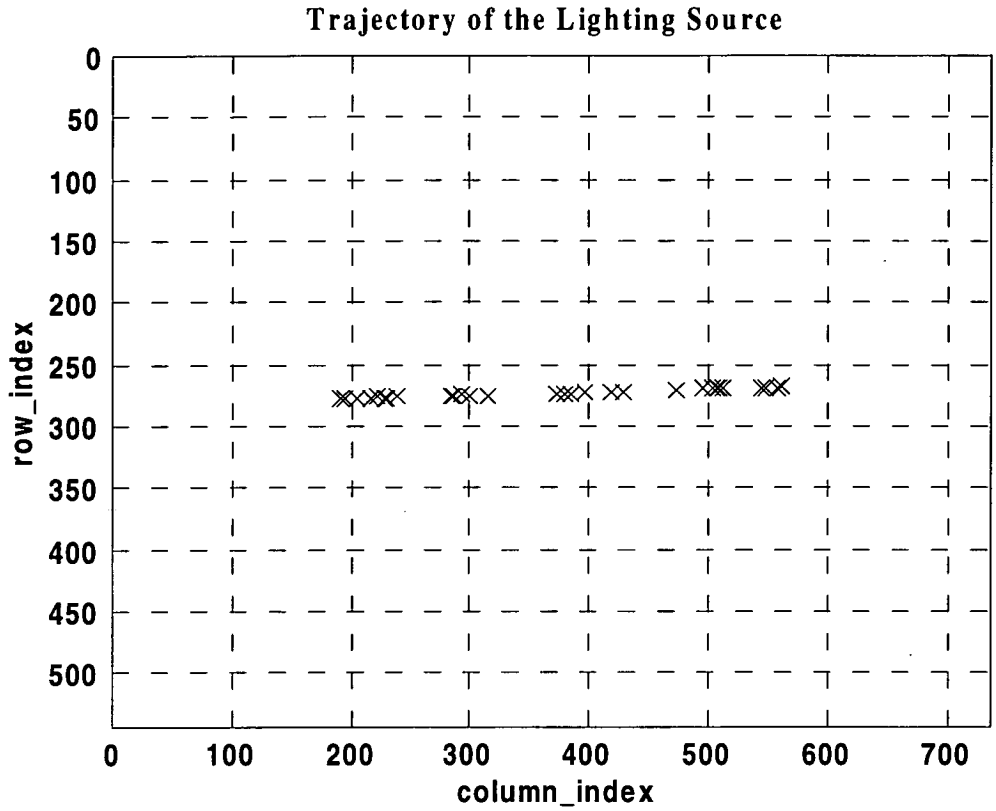


Figure 5.24 (The average number of pixels in each grabbed image: 38.5)
(Threshold =150, Cable Length = 700 mm)

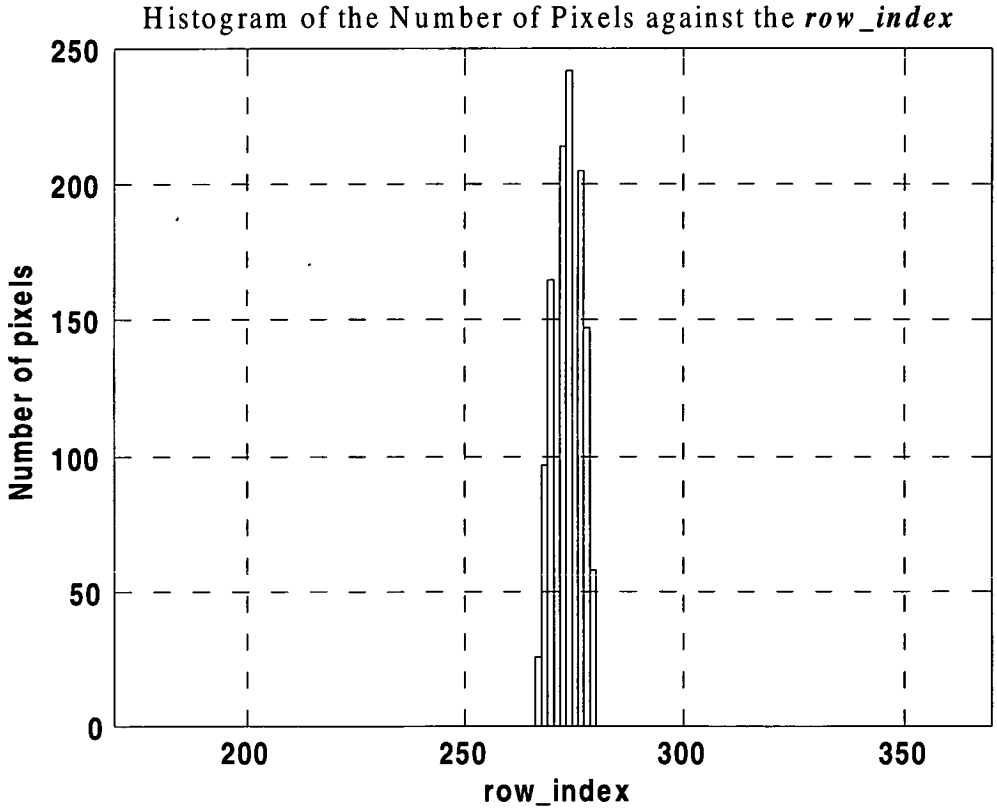


Figure 5.25 (Threshold =150, Cable Length = 700 mm)

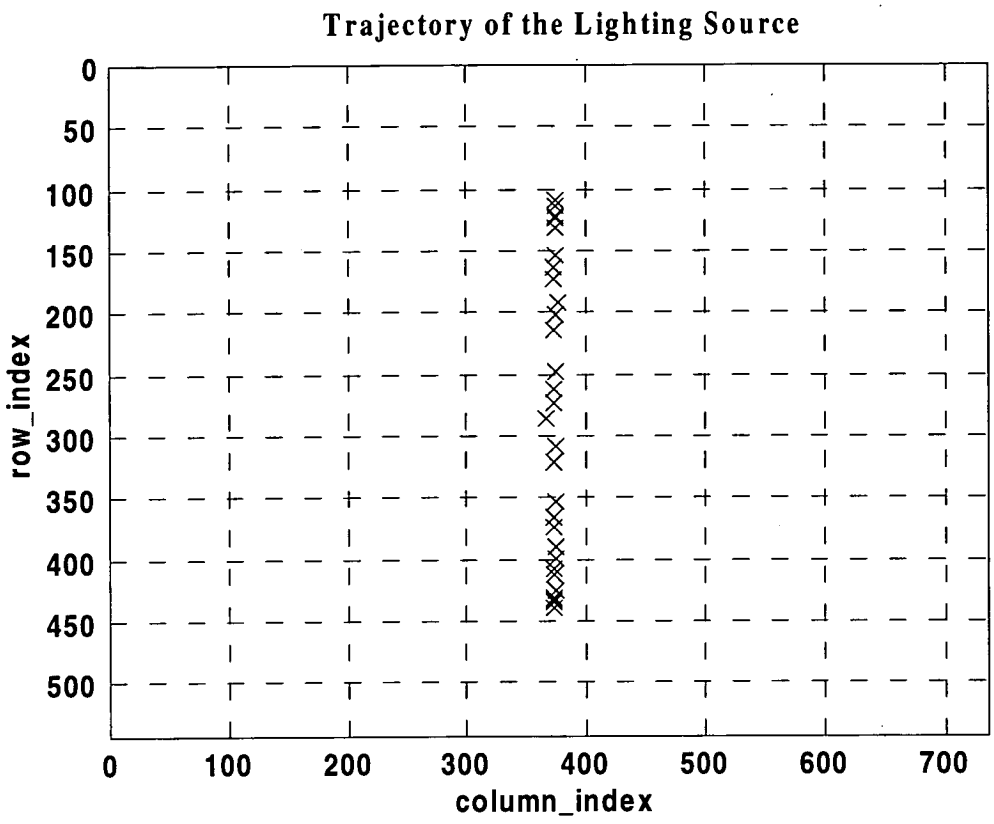


Figure 5.26 (The average number of pixels in each grabbed image: 22.7)
(Threshold =150, Cable Length = 950 mm)

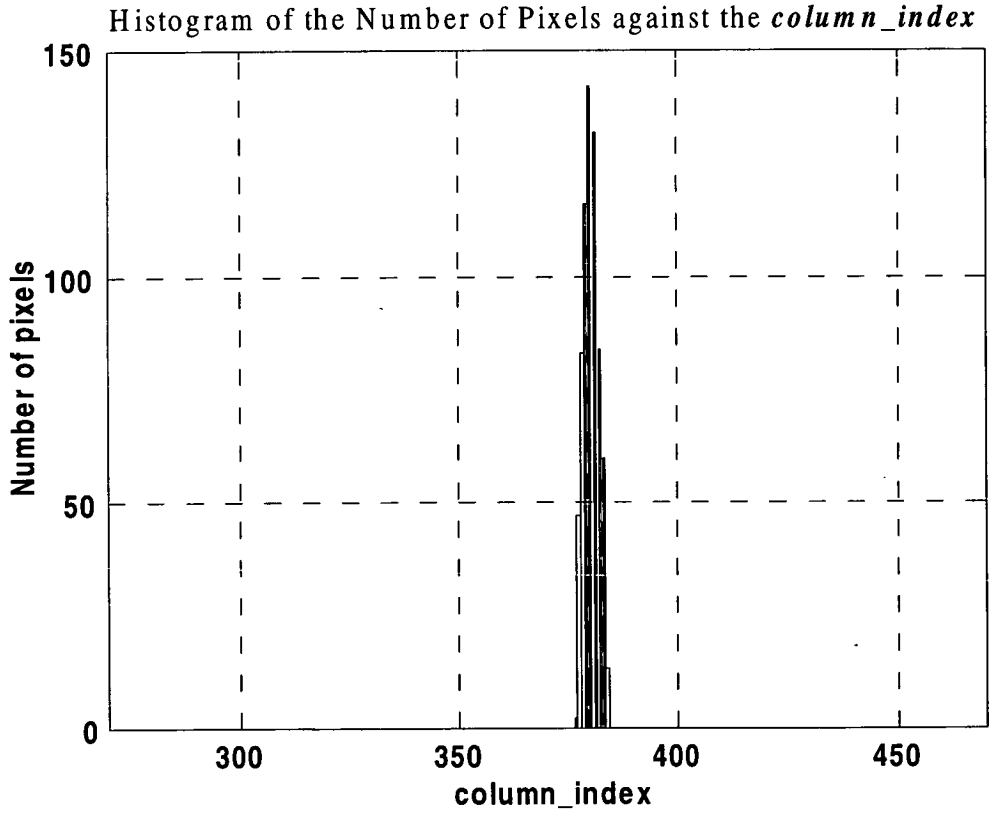


Figure 5.27 (Threshold =150, Cable Length = 950 mm)

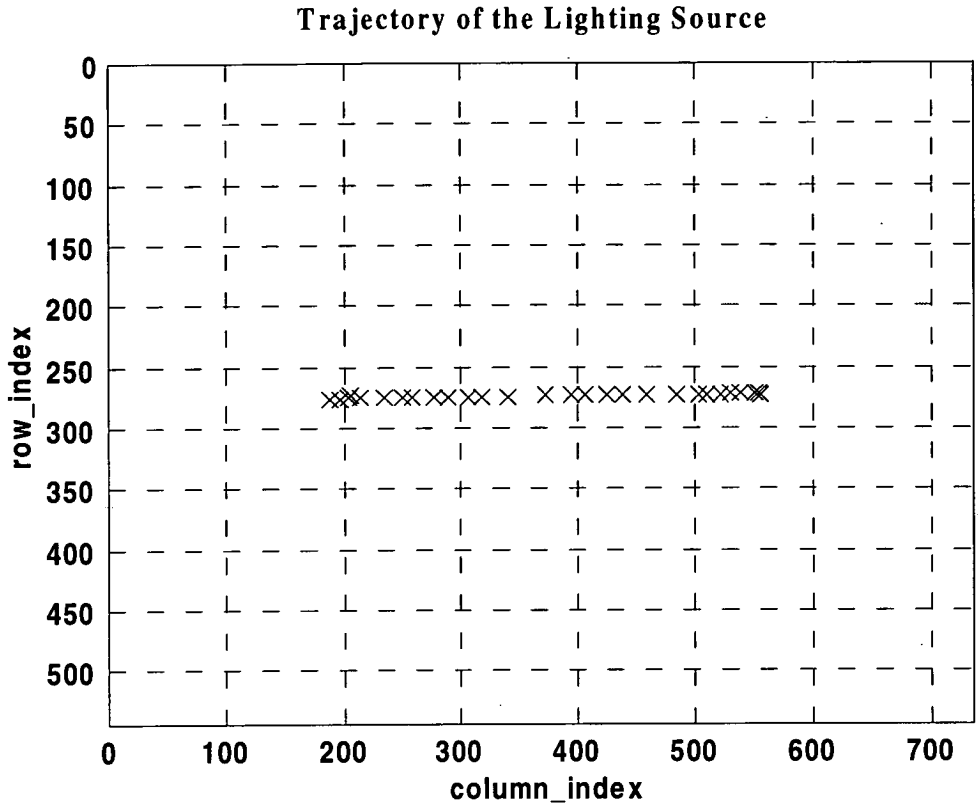


Figure 5.28 (The average number of pixels in each grabbed image: 24.0)
(Threshold =150, Cable Length = 950 mm)

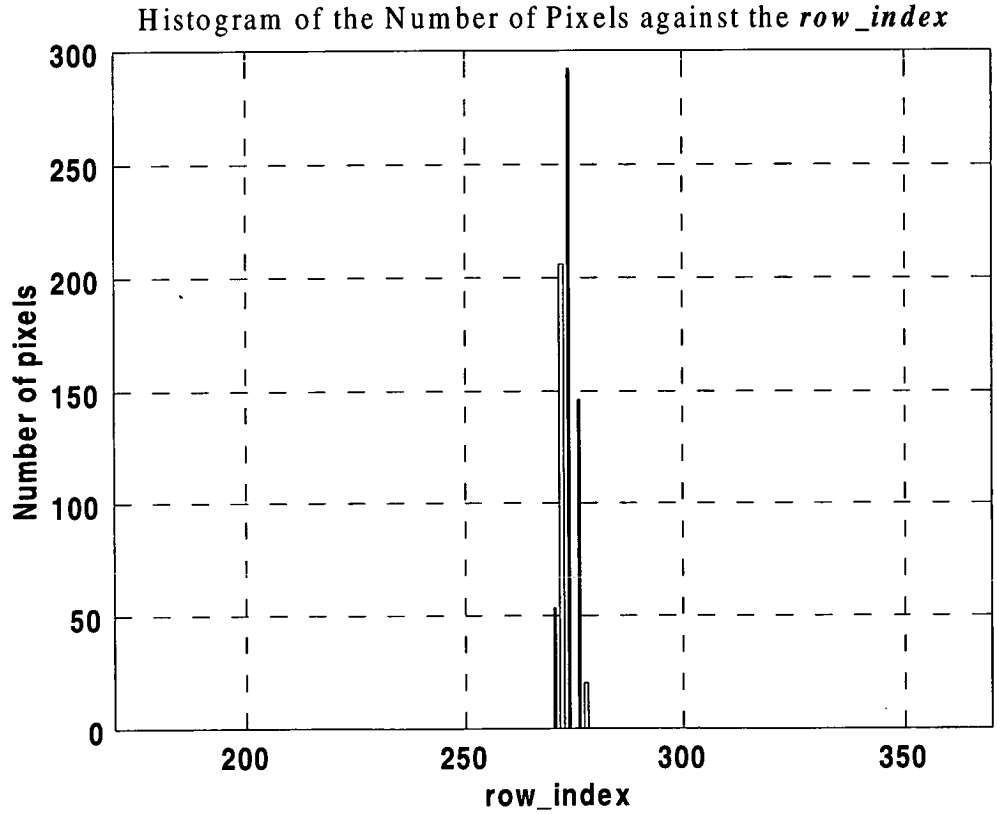


Figure 5.29 (Threshold =150, Cable Length = 950 mm)

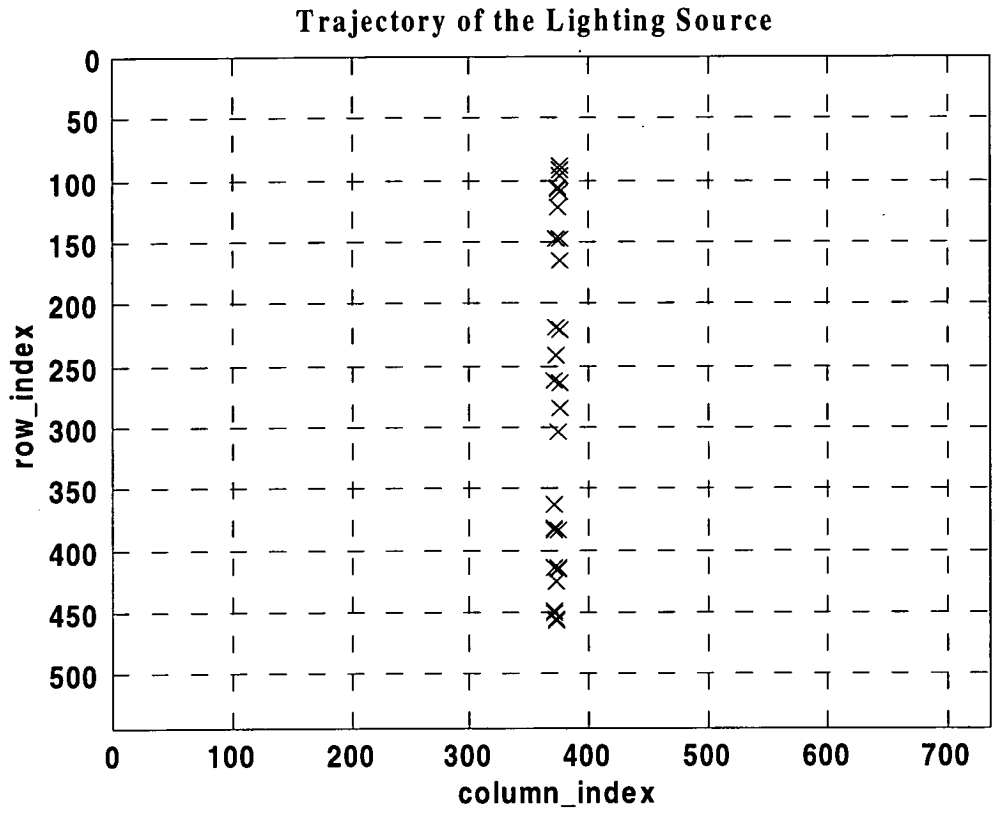


Figure 5.30 (The average number of pixels in each grabbed image: 16.7)
(Threshold =150, Cable Length = 1,200 mm)

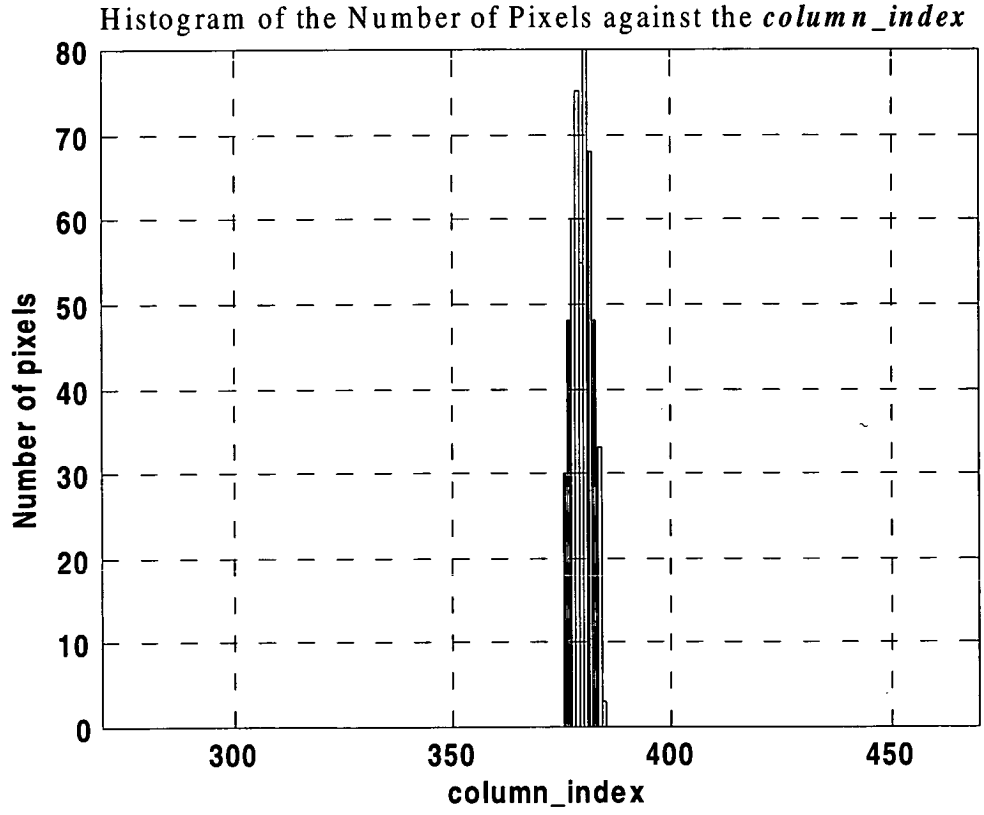


Figure 5.31 (Threshold =150, Cable Length = 1,200 mm)

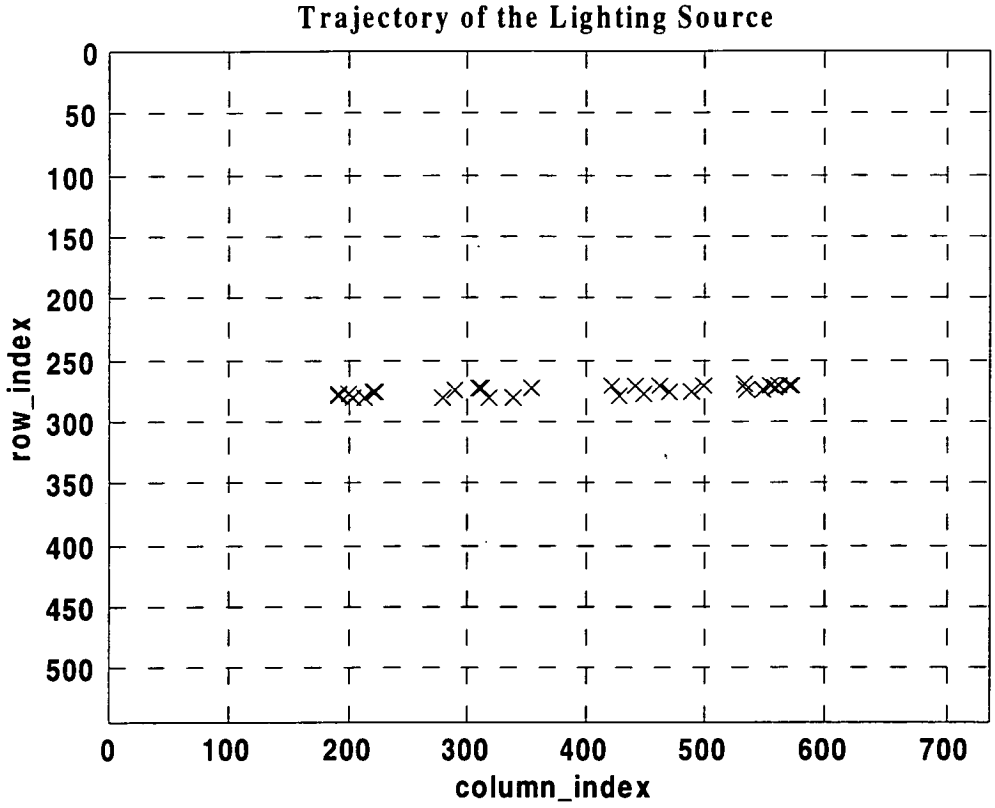


Figure 5.32 (The average number of pixels in each grabbed image: 16.5)
(Threshold =150, Cable Length = 1,200 mm)

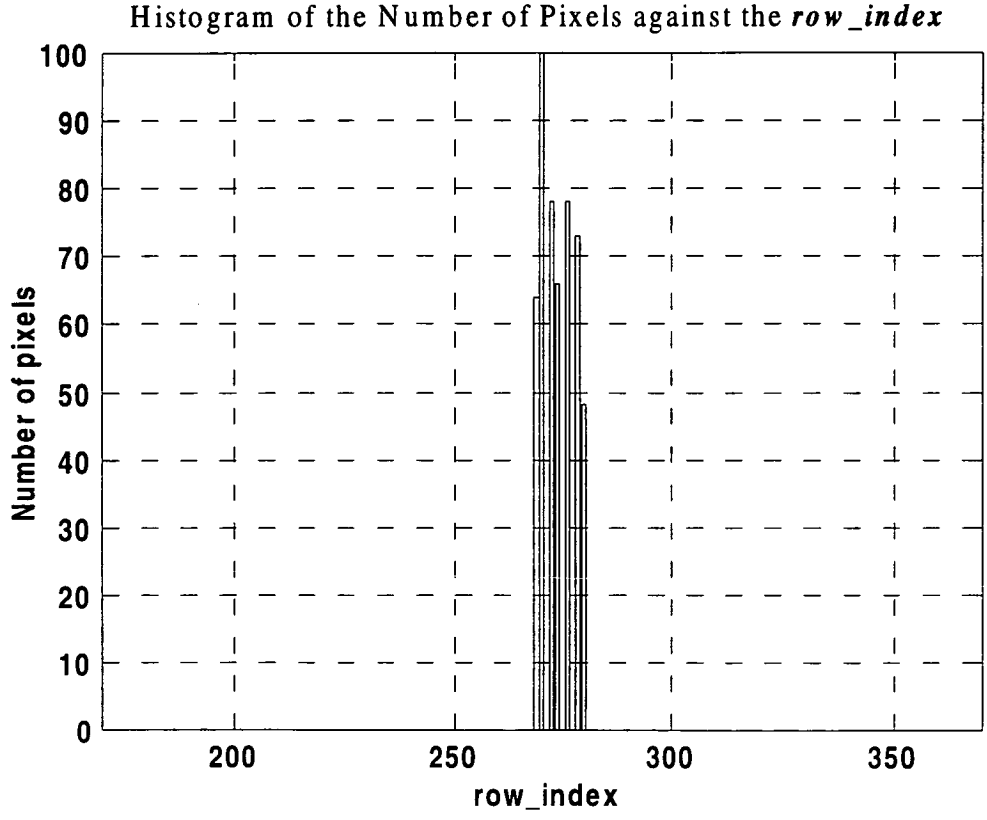


Figure 5.33 (Threshold =150, Cable Length = 1,200 mm)

5.3.2 The Partial Search Algorithm (PSA) (for a single lighting source)

Although the Exhaustive Search Algorithm (ESA) discussed in the last section is shown to be usable, there is still some room for the improvement. The ESA is designed to look into each point of the entire image from top left to bottom right, so time is wasted in checking (say) the empty space of the bottom half of the captured images if the cluster is actually located in the upper half. Secondly the ESA can only be applied to a configuration in which *only* one lighting source is present. With two lighting sources present on the spreader (necessary for obtaining the rotational angle, for example) it is no longer a feasible method.

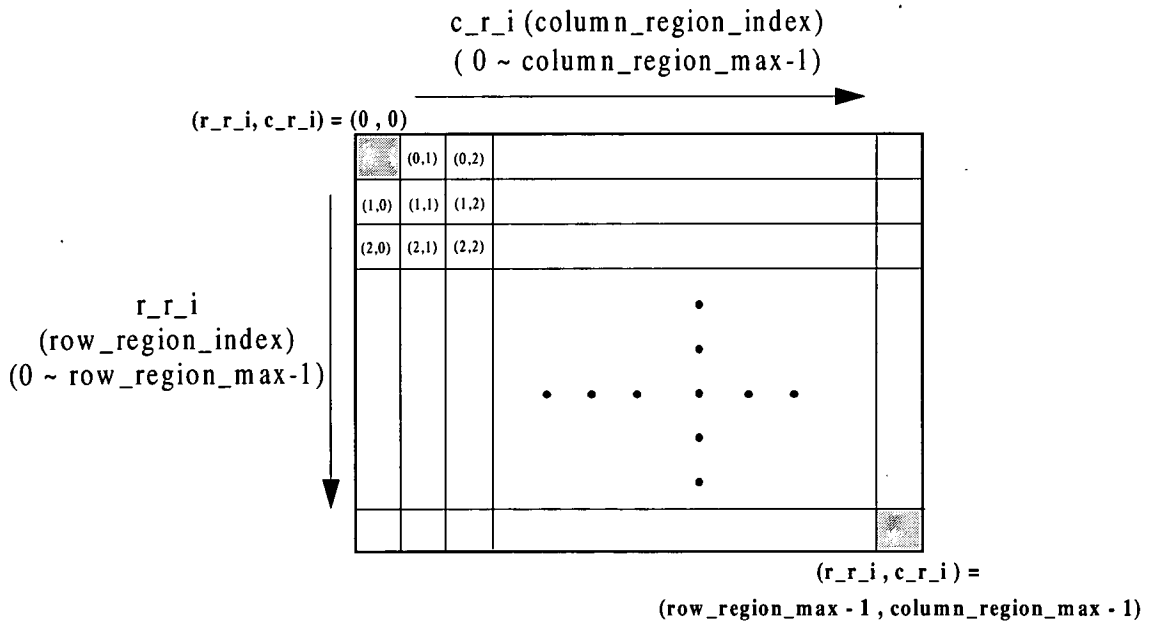


Figure 5.34 A captured image divided into grids (regions)

On assuming that *bright* pixels in a cluster are *continuous*, a *Partial Search Algorithm (PSA)* is described in the following steps:

(1) From Figure 5.34 the captured images are divided into grids (or *regions*) with size set up as (*row_div*) × (*column_div*). Currently, row_div and column_div are set to be equal to 32. Letting,

row_region_max = FrameHeight/row_div ------- (5.1)

column_region_max = FrameWidth/column_div ------- (5.2)

(2) As shown in Figure 5.35, each region is given another set of indices: r_i_i (row_internal_index), c_i_i (column_internal_index)).

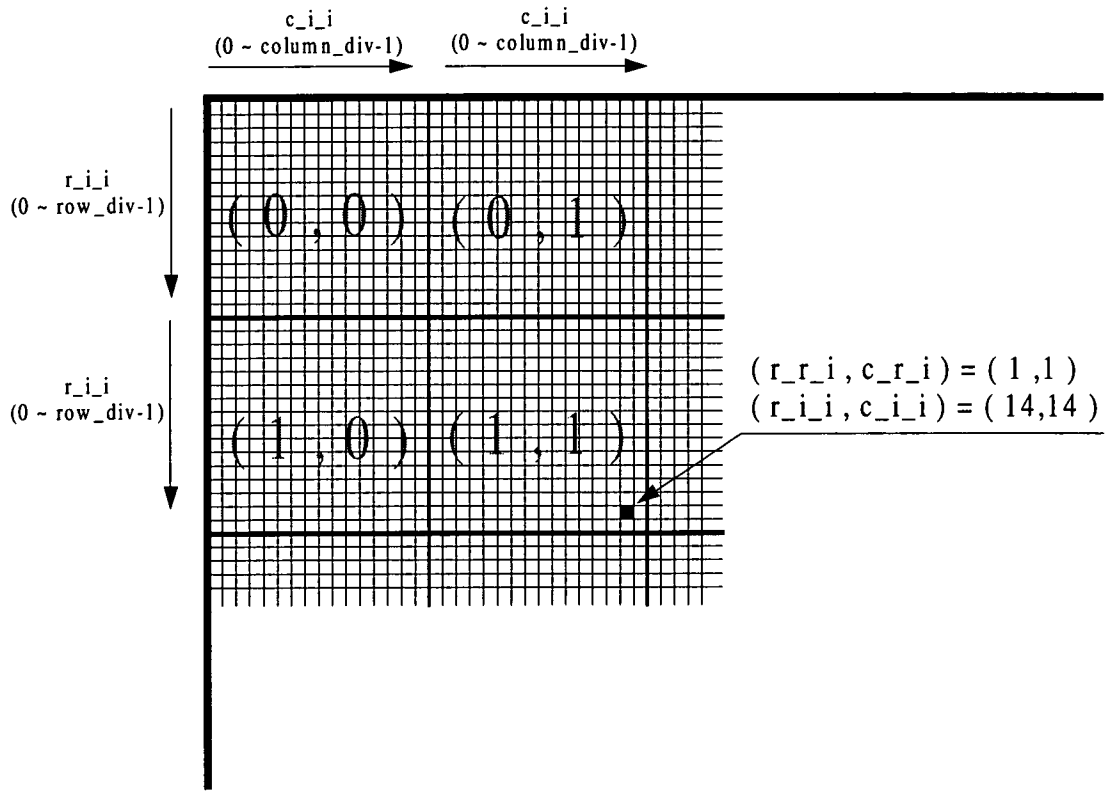


Figure 5.35 Each region is given a set of indices

(3) The algorithm searches each region from the top-left to bottom right, according to the region indices r_r_i, c_r_i, and in each region the pixels are searched from top left to bottom right as well, according to r_i_i, c_i_i. Therefore,

$$row\_index = r\_r\_i * row\_index\_max + r\_i\_i \quad \text{-------} \quad (5.3)$$

$$column\_index = c\_r\_i * column\_index\_max + c\_i\_i \quad \text{-------} \quad (5.4)$$

(4) On encountering the first bright pixel, $C_{start}(row\_index_{start}, column\_index_{start})$ in the region ($r\_r\_i_{start}$, $r\_i\_i_{start}$), then from equations (5.3) and (5.4) it is seen that,

$$row\_index_{start} = r\_r\_i_{start} * row\_index\_max + r\_i\_i_{start} \quad \text{-------} \quad (5.5)$$

$$column\_index_{start} = c\_r\_i_{start} * column\_index\_max + c\_i\_i_{start} \quad \text{-------} \quad (5.6)$$

Then the indices, $r\_r\_i_{start}$, $c\_r\_i_{start}$, $r\_i\_i_{start}$, $c\_i\_i_{start}$, for the first encountered bright pixel are passed to a subroutine, *calc()*, which looks for all the other bright pixels.

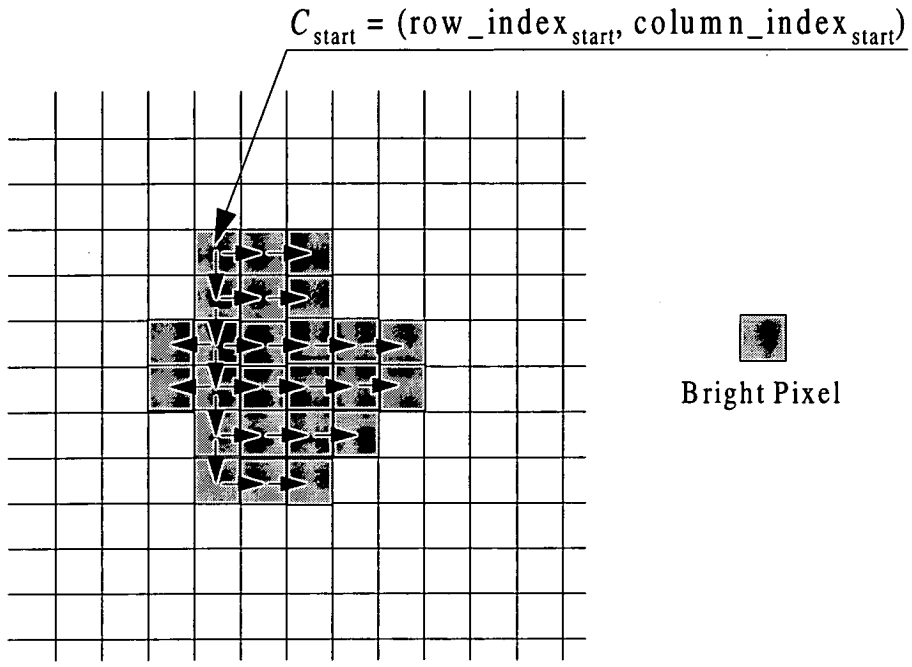


Figure 5.36 An example of the search sequence of the subroutine *calc()*

In addition to the logic of Figure 5.36 the operation of the subroutine, *calc()*, can also be illustrated as follows:

(1) Because the PSA searches from top left to bottom right the first bright pixel encountered, C_{start}, should be the top left pixel of the cluster.

(2) It then searches to the right, until the end of the positive stream (with pixel values equal to **1**).

(3) It goes to the next row and searches to the left of row_index_{start} until the end of the positive stream, and then searches to the right of row_index_{start} until it reaches the other end of the positive stream.

(4) It repeats step (3) until the row_index_{start}++ is of value zero.

(5) To ensure that the detected cluster is not just a noise pixel a cluster is disregarded if its number is less than **3**. It then moves on to look for another C_{start}.

(Note: The number, 3, in step (5) is based on the test results of the noise pixels from the ESA in the last section.)

To test the performance of this algorithm it is sensible to start from the longest cable length, 1,200 mm, because the least number of pixels can be detected at this length. As before three types of tests are taken in which the spreader is stationary, then moving vertically and then horizontally, respectively. 100 frames are sampled in each test and the trajectories of the central lighting source are plotted in Figures 5.37, 5.38, and 5.39 at the end of this section. An analysis of their results is summarized in Table 5.3. First, as shown in Figure 5.37, the PSA performs reasonably well , because the cluster in each frame concentrates on the same coordinate (273.2±0.1, 373.4±0.1) where the *standard error of the mean* (Morris [56]) is applied. Also, the overall processing speed increases to 8.1 frames/second compared to 4.9 frames/second in the ESA.

The moving direction	Stationary	Vertical ↑ ↓	Horizontal ← →
Percentage of failures (%)	0	**23.0**	0
Average number of detected pixels in one frame (excluding the frames of failure)	21.0	9.1	10.9
The average processing speed (frames/second)	8.1	6.8	7.7

Table 5.3 The test result with threshold = 150, l = 1,200 mm.

However, the result in the second test, in which the spreader moves in the vertical direction, has shown a 23 percent failure rate. As indicated in Figure 5.38, 23 frames have not registered anything, so the coordinates of the lighting source in these frames are (0, 0). Also these failures tend to occur when the lighting source passes through the central area (row_index = FrameHeight/2 = 272), where the spreader moves with maximum velocity. Since the TV signals always scan vertically from top to bottom (↓), the lighting source appears to be elongated along the row-axis if the spreader moves in

the same direction (↓). Therefore, it is possible in some instances that the pixels of the lighting source are too *sparse* to form a cluster with a sufficient number of pixels (3) to be identifiable from the background noise. On the contrary if the spreader moves from bottom to top (↑), opposite to the scanning direction of the TV signals (↓), the lighting source appears to be *shrunk* in shape. This phenomenon, which could almost be thought of as a sort of visual *Doppler effect*, causes the average number of pixels detected to come down to 9.1, compared with 20.1 when the spreader is kept stationary. Nevertheless, because the horizontal scanning frequency of the TV signals, 15.625 KHz, is comparatively much higher than the vertical scanning frequency, 50 Hz, this algorithm does not suffer from not detecting any clusters with sufficient number of pixels when the spreader moves horizontally, as shown in Figure 5.39.

Since the *number* of the pixels accumulated by the moving lighting source directly influences the performance and accuracy when searching for the lighting source, the easiest way to increase performance is to reduce the *threshold* in the configuration file for the framegrabber, which is currently set to 150. First the threshold is set to 125 and, as before, the tests in those three conditions are undertaken. As shown in Table 5.4 and Figure 5.41 although there are still some failures taking place in the test (with the spreader moving vertically) it can be seen that the percentage failure rate has dropped down to 7.0 %.

The moving direction	Stationary	Vertical ↑ ↓	Horizontal ← →
Percentage of failure (%)	0	**7.0**	0
Average number of detected pixels in one frame (excluding the frames of failure)	25.3	9.5	16.4
The average processing speed (frames/second)	8.1	7.1	12.6

Table 5.4 The test with threshold = 125, l = 1,200 mm.

By further reducing the threshold to 100 the percentage failure in the test in Figure 5.43 decreases to 4.0 %. The test results are shown in Figure 5.43 ~ 5.45. The analysis of their results is summarized in Table 5.5.

The moving direction	Stationary	Vertical ↑ ↓	Horizontal ← →
Percentage of failure (%)	0	**4.0**	0
Average number of detected pixels in one frame (excluding the frames containing failures)	34.7	13.9	24.2
The average processing speed (frames/second)	8.0	7.0	7.8

Table 5.5 The test result with threshold = 100, l = 1,200 mm.

It is considered unwise to reduce the threshold below the minimum value, 95, because the background noise can be found to increase too much. Also, to ensure that the final setup, threshold = 100, is applicable to shorter cable lengths, in Figure 5.46 ~ 5.48 and Table 5.6 the test results are shown for the cable length equal to 700 mm. It can be seen that the accuracy of the partial search algorithm improves as the number of bright pixels increases, as one would expect.

In summary a new algorithm has been derived to locate the lighting source and has been detailed in this section. The overall processing speed has been increased by *50* percent compared to that of the exhaustive search algorithm described in the previous section. In order to increase the accuracy of this algorithm, tests with different setups, involving variations in the threshold and cable length, are performed to find an optimum configuration. In the next section this algorithm is extended further in order to locate *two* lighting sources on the spreader.

The moving direction	Stationary	Vertical ↑ ↓	Horizontal ← →
Percentage of failure (%)	0	**1.0**	0
Average number of detected pixels in one frame (excluding the frames of failure)	78.4	27.4	47.8
The average processing speed (frames/second)	8.0	6.9	7.9

Table 5.6 The test result with threshold = 100, l = 700 mm.

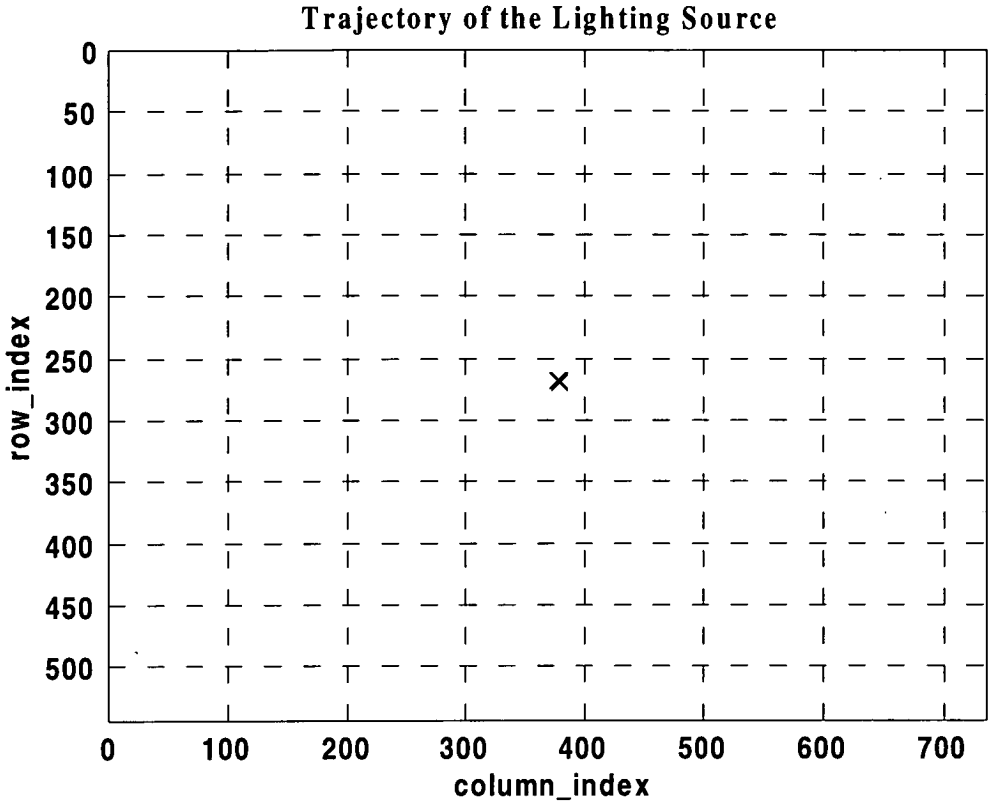


Figure 5.37 (Threshold =150, Cable Length = 1,200 mm)

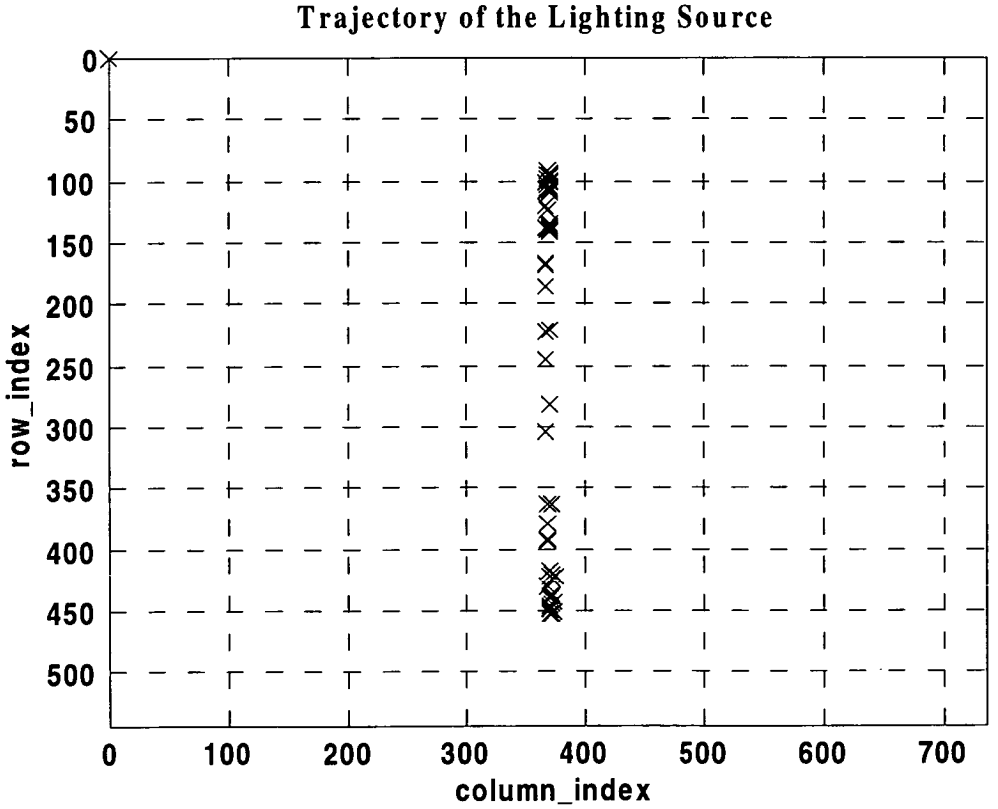


Figure 5.38 (Threshold =150, Cable Length = 1,200 mm)

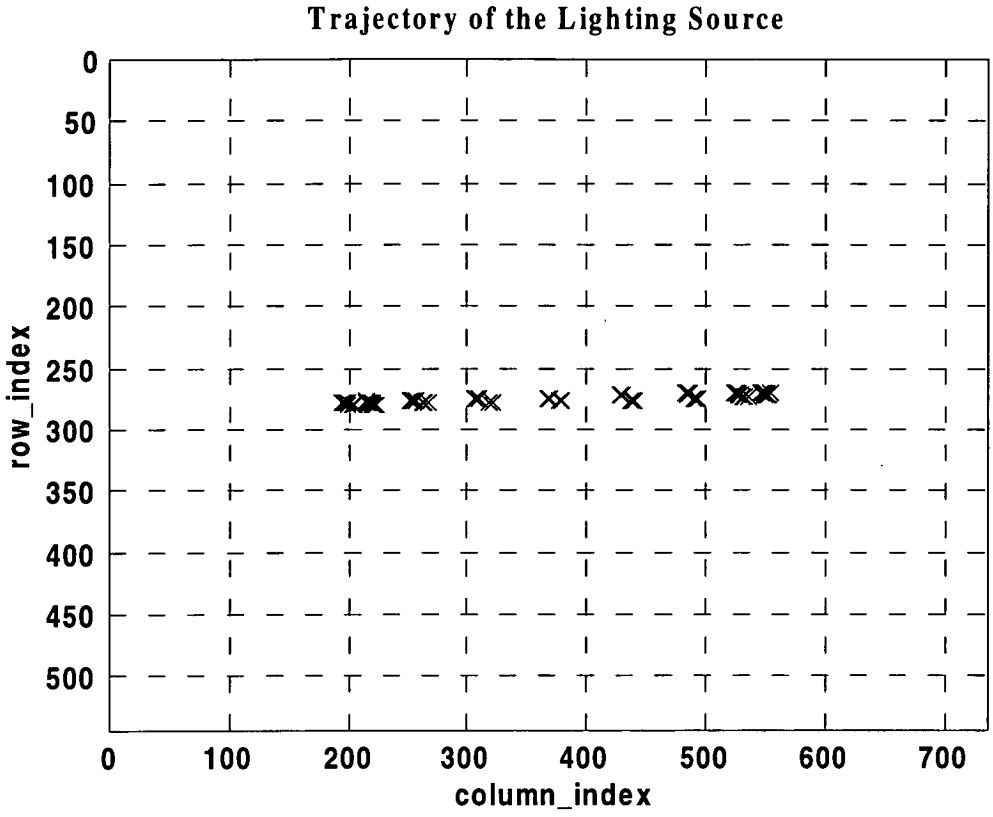


Figure 5.39 (Threshold =150, Cable Length = 1,200 mm)

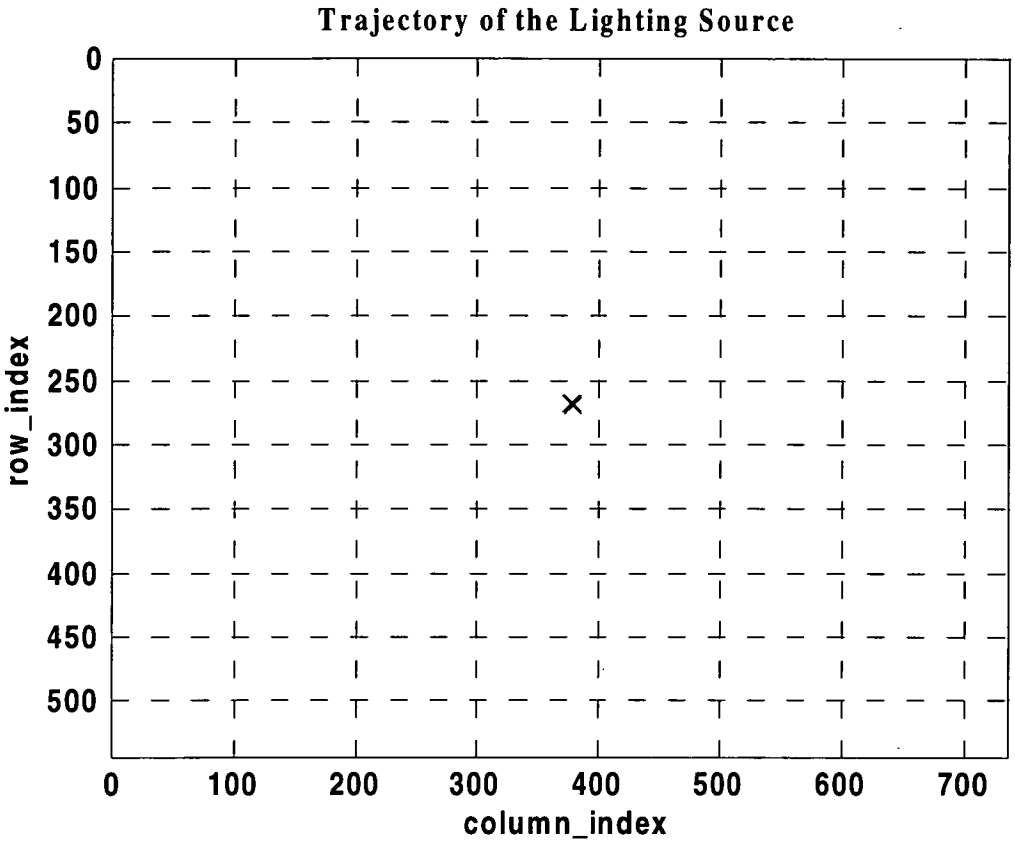


Figure 5.40 (Threshold =125, Cable Length = 1,200 mm)

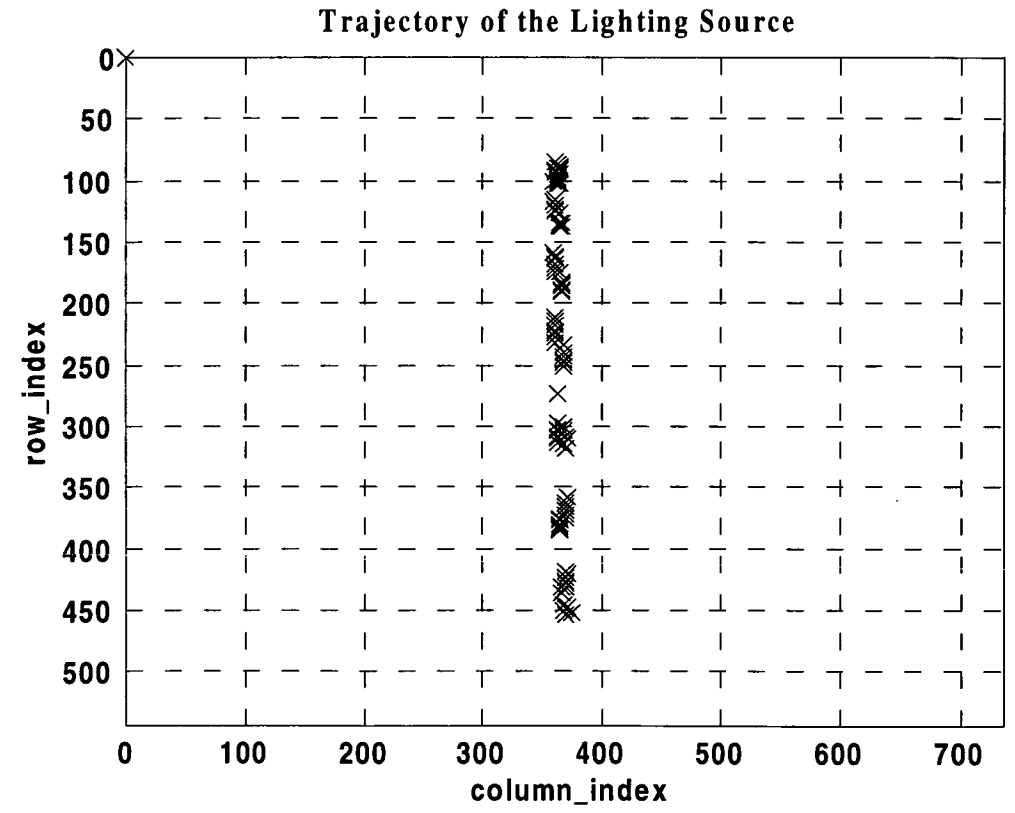


Figure 5.41 (Threshold =125, Cable Length = 1,200 mm)

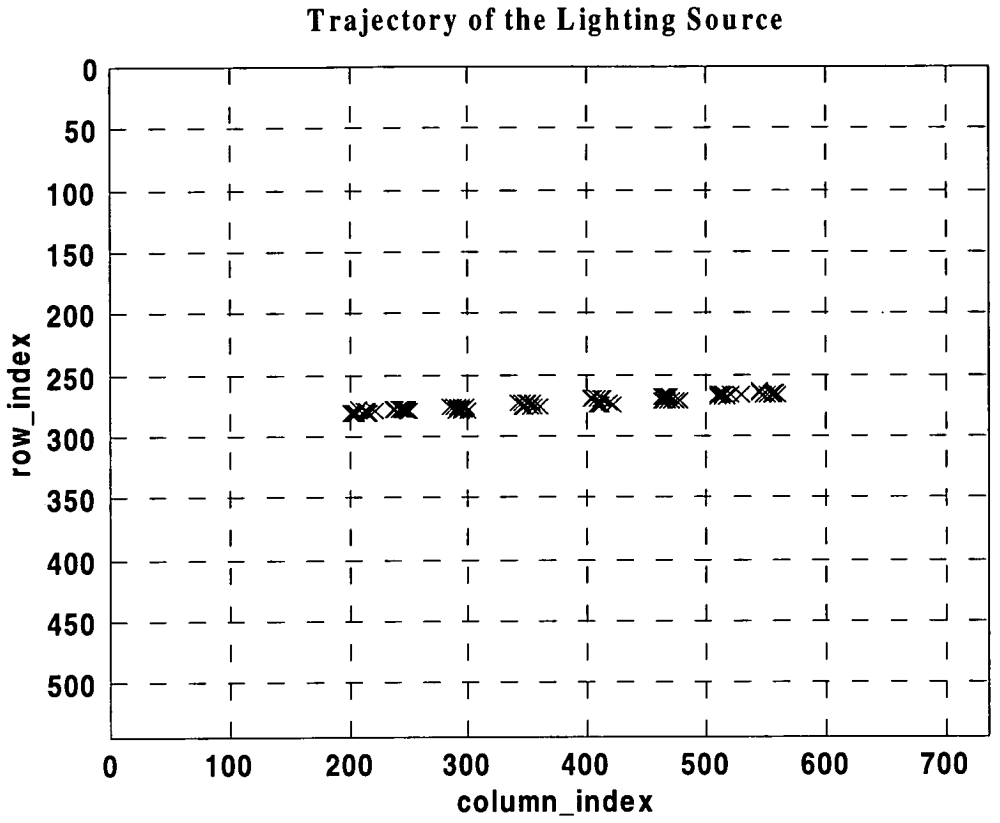


Figure 5.42 (Threshold =125, Cable Length = 1,200 mm)

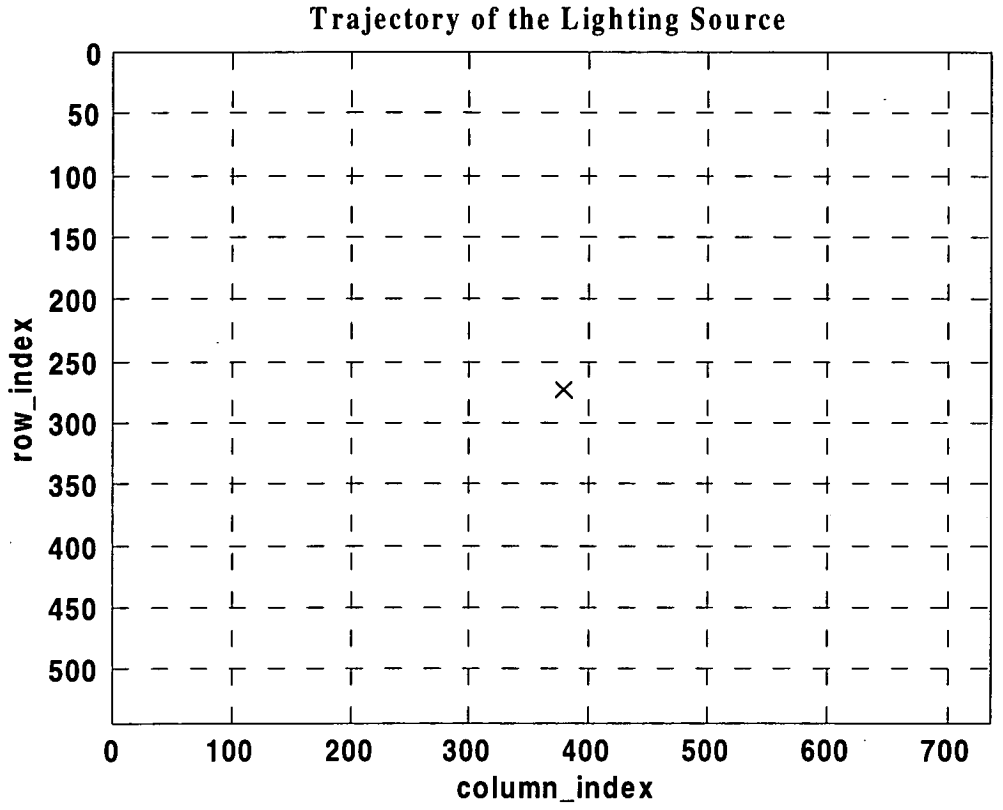


Figure 5.43 (Threshold =100, Cable Length = 1,200 mm)

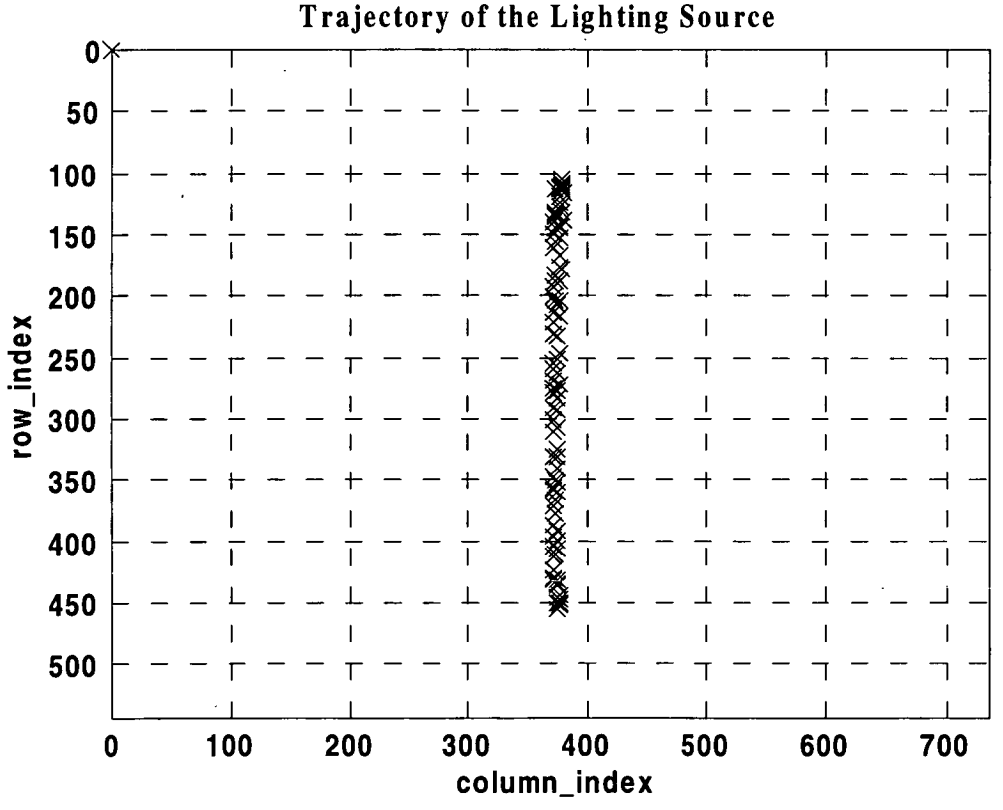


Figure 5.44 (Threshold =100, Cable Length = 1,200 mm)

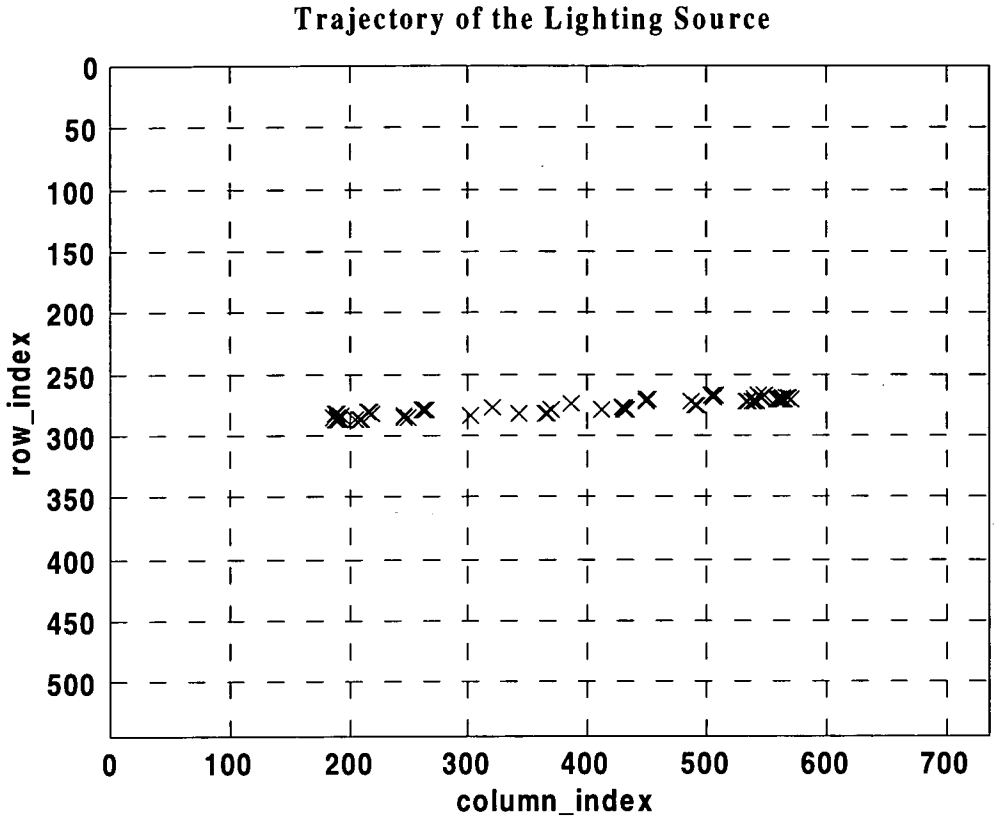


Figure 5.45 (Threshold =100, Cable Length = 1,200 mm)

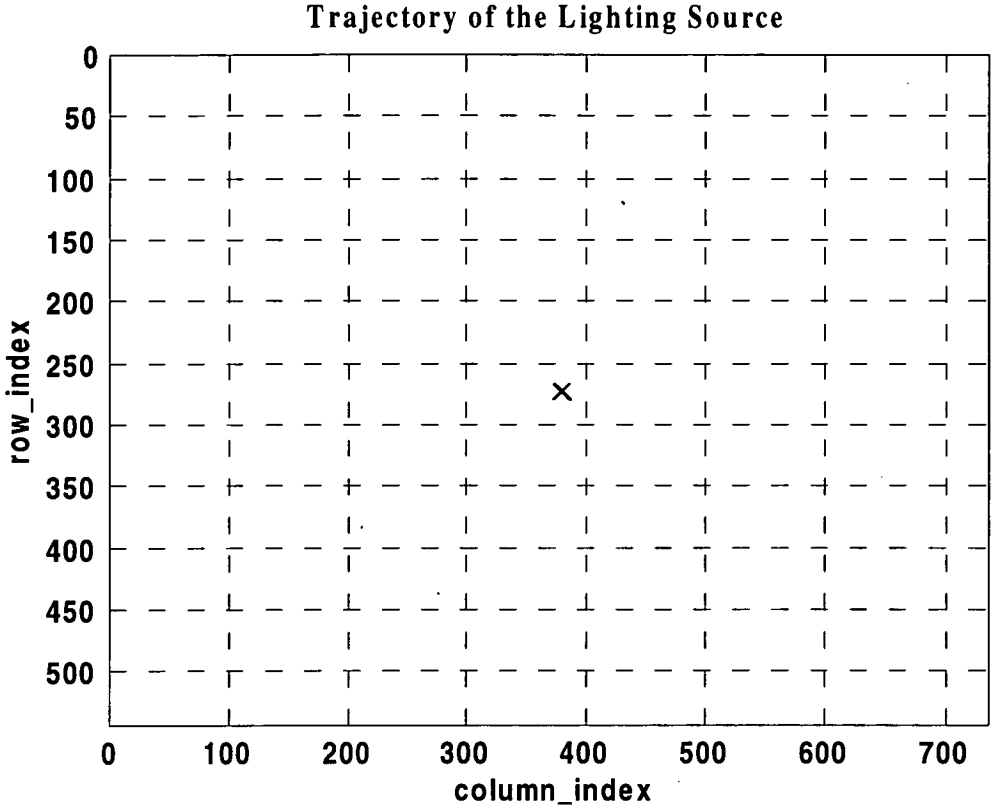


Figure 5.46 (Threshold =100, Cable Length = 700 mm)

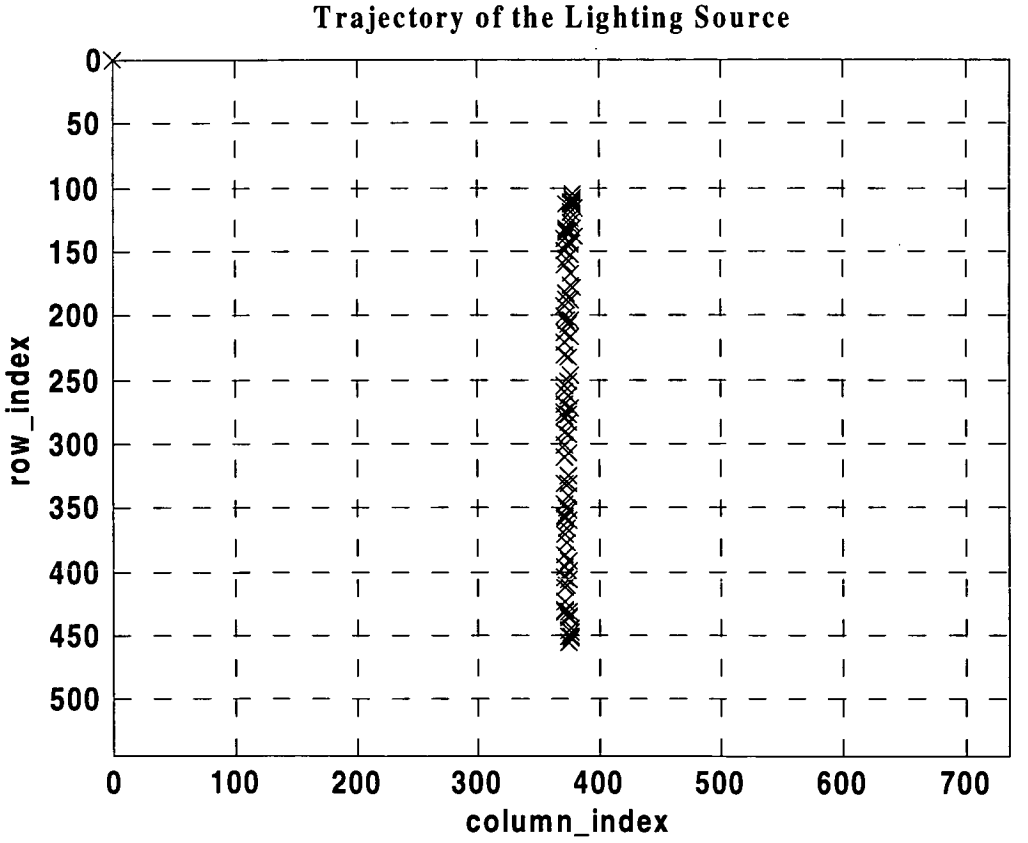


Figure 5.47 (Threshold =100, Cable Length = 700 mm)

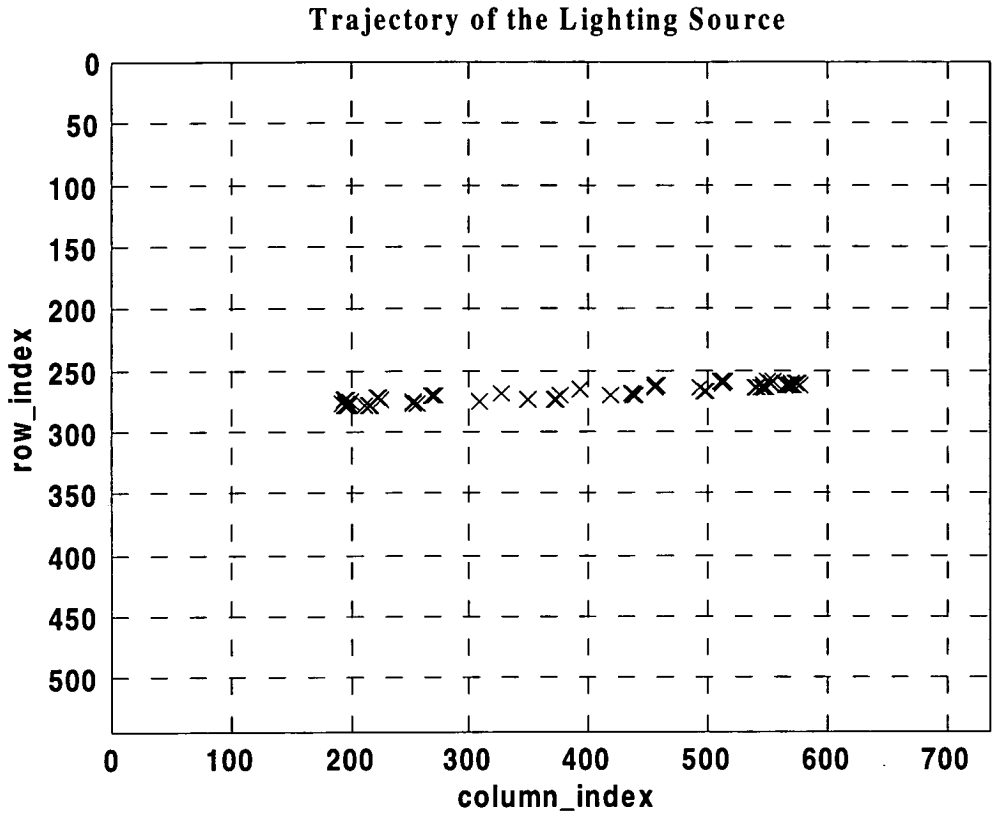


Figure 5.48 (Threshold =100, Cable Length = 700 mm)

5.3.3 An Improved Partial Search Algorithm (IPSA) for use with two lighting sources

In order to enable the PSA to locate two lighting sources, the concept of a so-called *region of rejection* is introduced here. As stated in section 5.3.2 the entire image is divided into regions with a size set up as (*row_div*) × (*column_div*), as shown in Figure 5.35. To modify the PSA described in the last section so that it can be applied to situations with two lighting sources present, there are two assumptions as follows:

The assumptions for the *Improved Partial Search Algorithm (IPSA)*:

(1) The size of an individual region has to be large enough to contain a single lighting source. For example, the setting for the PSA throughout the last section is 32×32 ((*row_div*) × (*column_div*)) = 576, which satisfies this condition, because the maximum number of pixels is only about 79 in Table 5.6 with the lowest threshold (100) and the shortest cable length (700 mm).

(2) The distance between the central coordinate of each lighting source is larger than the size of one individual region. For example, if the central coordinates of the two lighting sources are (row_index_1, $column\_index_1$) and (row_index_2, $column\_index_2$), then,

$$|row\_index_1 - row\_index_2| > row\_div \qquad \text{-------} \quad (5.7)$$

$$|column\_index_1 - column\_index_2| > column\_div \qquad \text{-------} \quad (5.8)$$

Based on the above assumption, Figure 5.49 shows that there are four possible positions for a lighting source present in the divided regions of an image. Once the first lighting source is located the corresponding region has to be labeled and disregarded whist the algorithm searches for the second lighting source. However, if the first lighting source is located in the situations in Figure 5.49 [b][c][d] it is possible that part of the first lighting source may be mistaken as the start of the second lighting source. In the worst case, as shown in Figure 5.49 [d], the PSA will misunderstand

that there are four lighting sources present. Therefore, in order to prevent this from happening it is necessary to skip over the surrounding regions of the first lighting source as shown in Figure 5.49 [d].

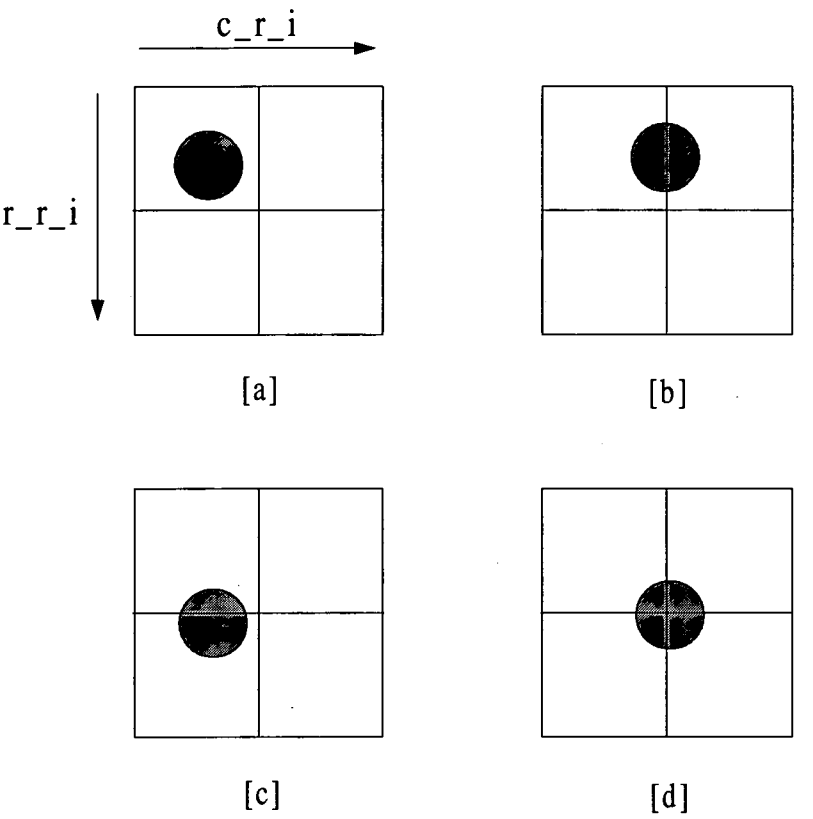


Figure 5.49 Four possible positions for a lighting source situated in the regions shown

The *Improved Partial Search Algorithm (IPSA)* is described as follows:

(1) It executes the Partial Search Algorithm to find the location of the first lighting source, C_1.

(2) Once the location of the first lighting source, C_1, in the region $((r\_r\_i)_1, (c\_r\_i)_1)$, is known those regions around it are marked and skipped over, so that the algorithm can proceed to detect the next target correctly. These regions of rejection , as shown in Figure 5.49 [d], are $((r\_r\_i)_1, (c\_r\_i)_1)$, $((r\_r\_i)_1 + 1, (c\_r\_i)_1)$, $((r\_r\_i)_1, (c\_r\_i)_1 + 1)$, $((r\_r\_i)_1 + 1, (c\_r\_i)_1 + 1)$,).

(3) If the expected quantity of lighting sources is found (in this case, *2*), then the rest of the image is ignored and the coordinates are returned to the main program.

The test results are summarized in Table 5.7 and plotted in Figure 5.50 ~ 5.54. The vertical movement of the spreader still causes some detection errors. The horizontal movement, as before, shows an entirely satisfactory outcome.

The moving direction	Stationary		Vertical ↑ ↓		Horizontal ← →	
The lighting source number	#1	#2	#1	#2	#1	#2
Percentage failure (%)	0		**4.0**		0	
Average number of detected pixels in one frame (excluding the frames with failures)	47.2	48.0	27.4	23.5	39.1	38.2
The average processing speed (frames/second)	8.0		6.6		7.9	

Table 5.7 The test result of two lighting sources with threshold = 100, l = 1,200 mm.

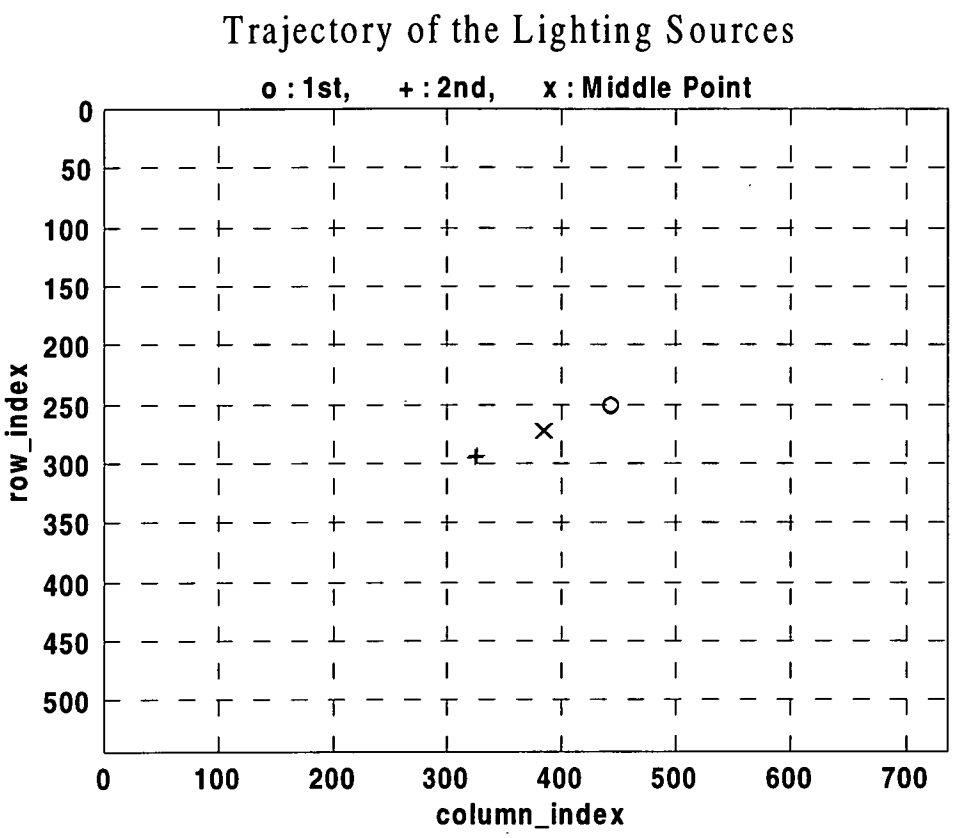


Figure 5.50 (Threshold =100, Cable Length = 1,200 mm)

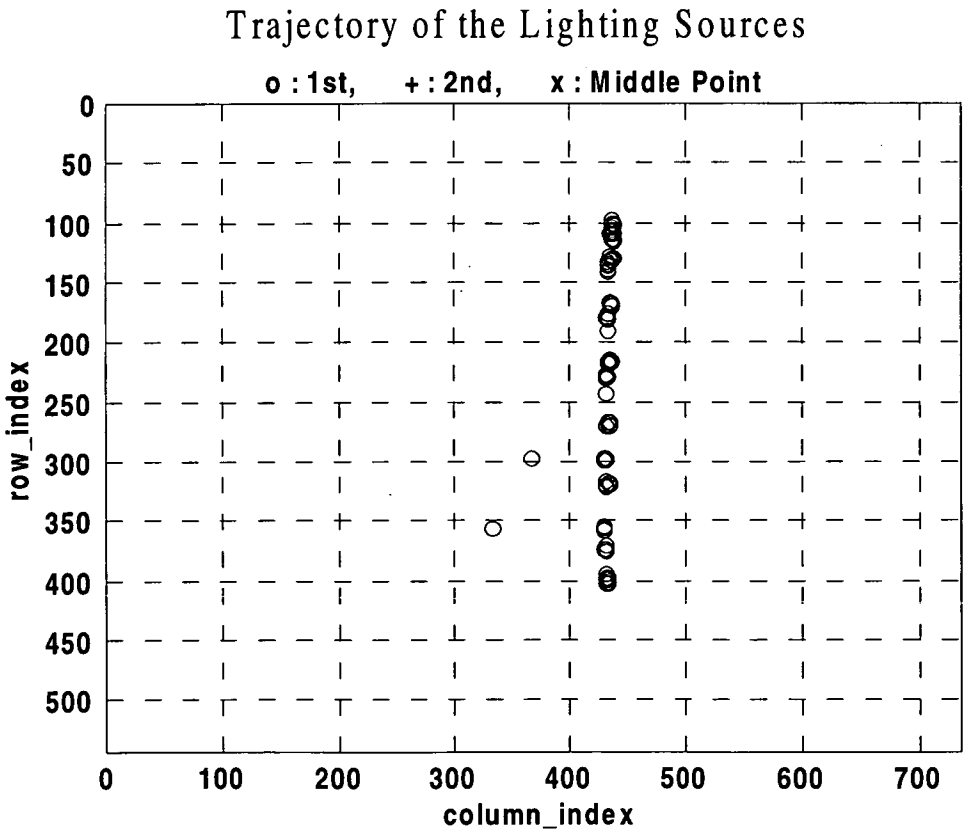


Figure 5.51 (Threshold =100, Cable Length = 1,200 mm)

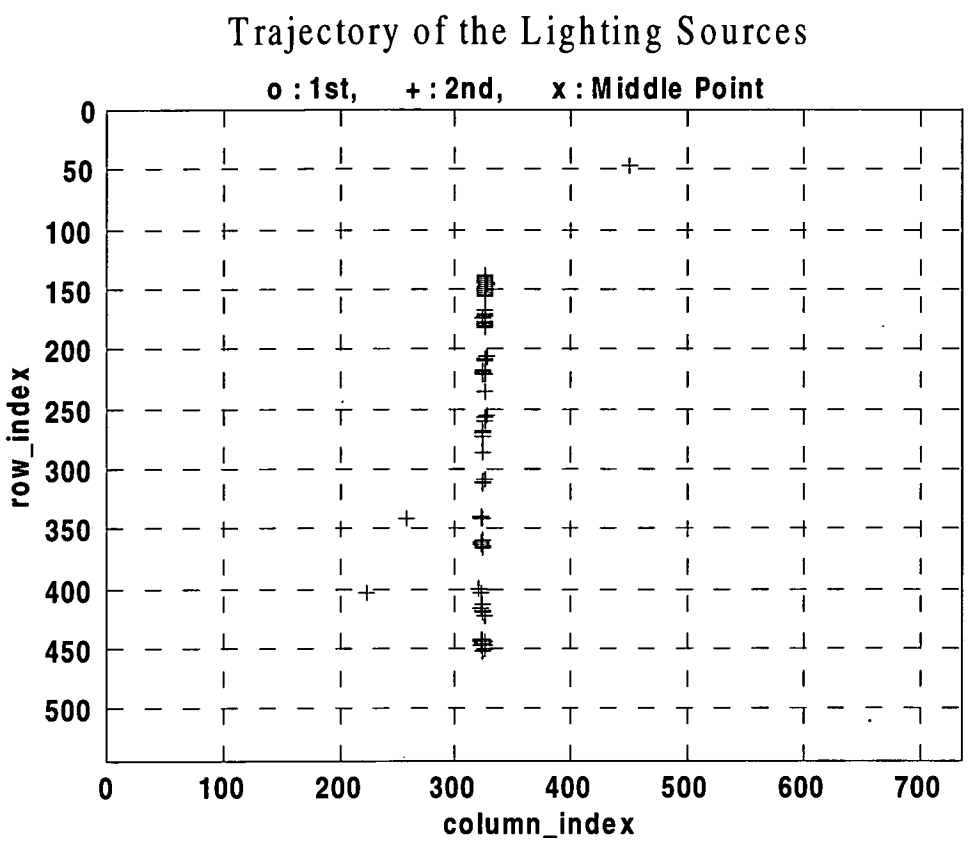


Figure 5.52 (Threshold =100, Cable Length = 1,200 mm)

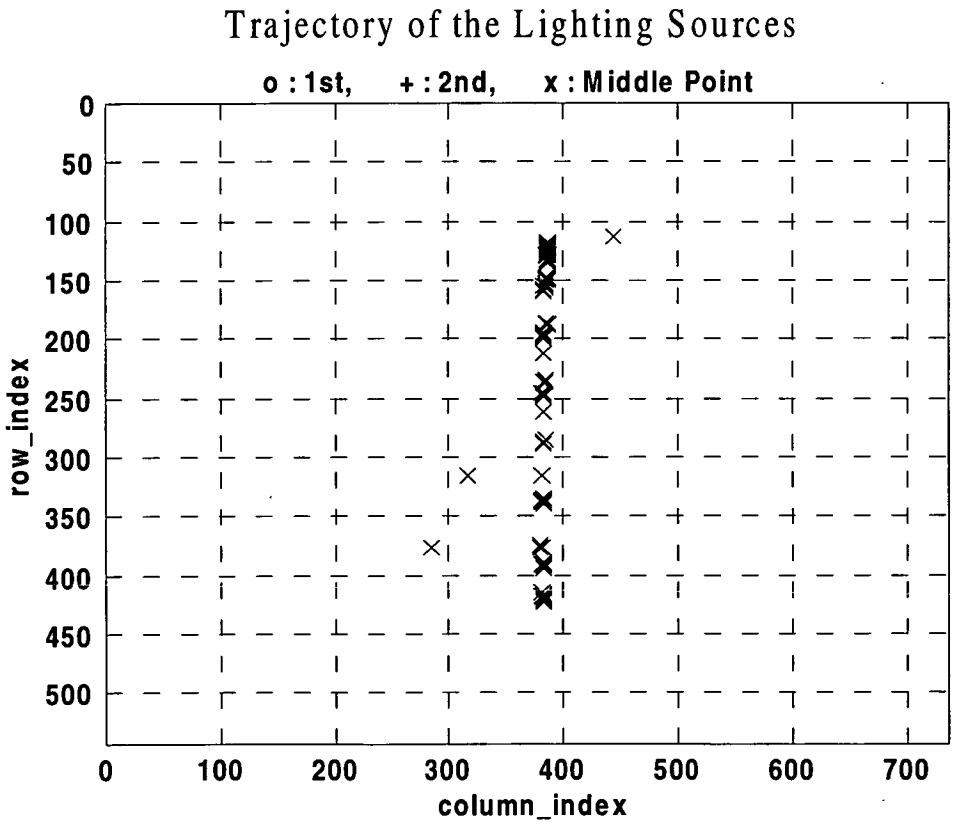


Figure 5.53 (Threshold =100, Cable Length = 1,200 mm)

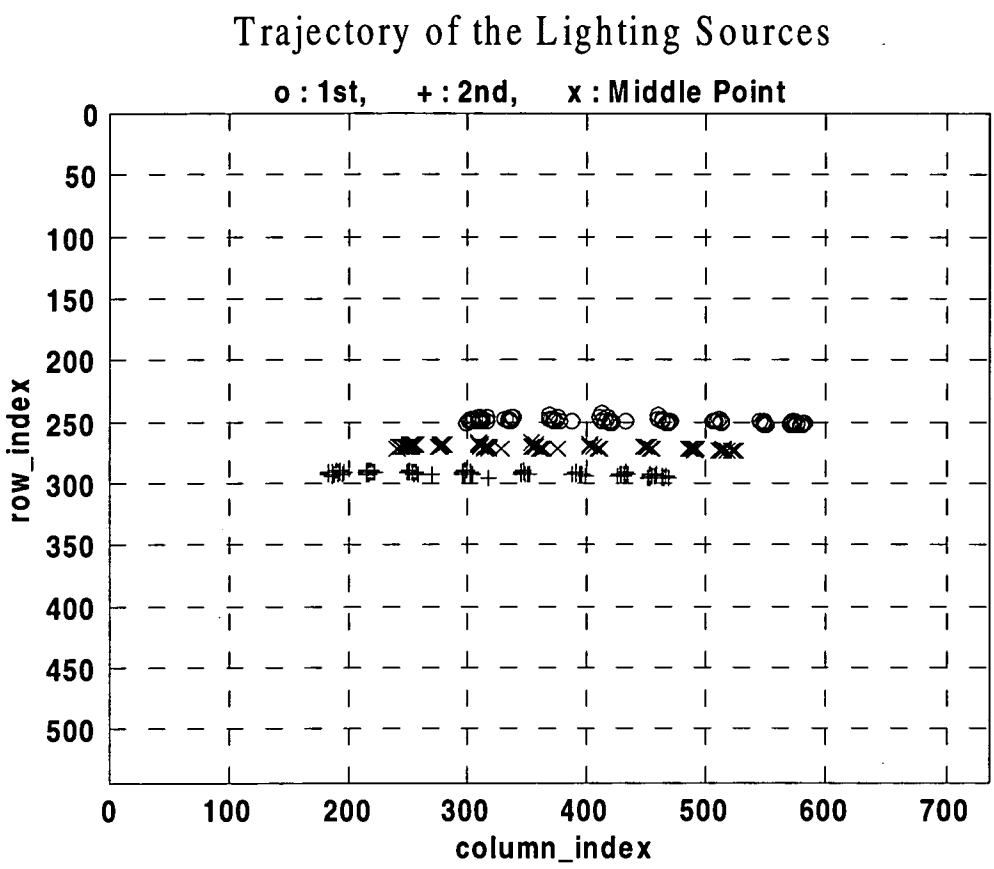


Figure 5.54 (Threshold =100, Cable Length = 1,200 mm)

5.4 Conclusion

Because of the inherent difficulty in measuring the instantaneous position and orientation of the spreader a special visual sensing system has been designed to overcome this problem. In section 5.2, the configuration of this system is described. A *binary* mode is employed, owing to its advantages of simplicity and speed. In section 5.3, three search algorithm are proposed. The Exhaustive Search Algorithm (ESA), in section 5.3.1, can be used for testing the performance, and calibrating the position of the CCD camera. The Partial Search Algorithm (PSA), in section 5.3.2, is proposed to increase the overall processing speed, and can be used for locating two lighting sources. Finally, the Improved Partial Search Algorithm (IPSA) is discussed in section 5.3.3, where the 'regions of rejection' around the first found lighting source are ignored, so that the second lighting source can be accurately located. It should be noted that the IPSA is not limited to locating only two targets. By simply changing the parameters in the program, multiple targets can be found accordingly.

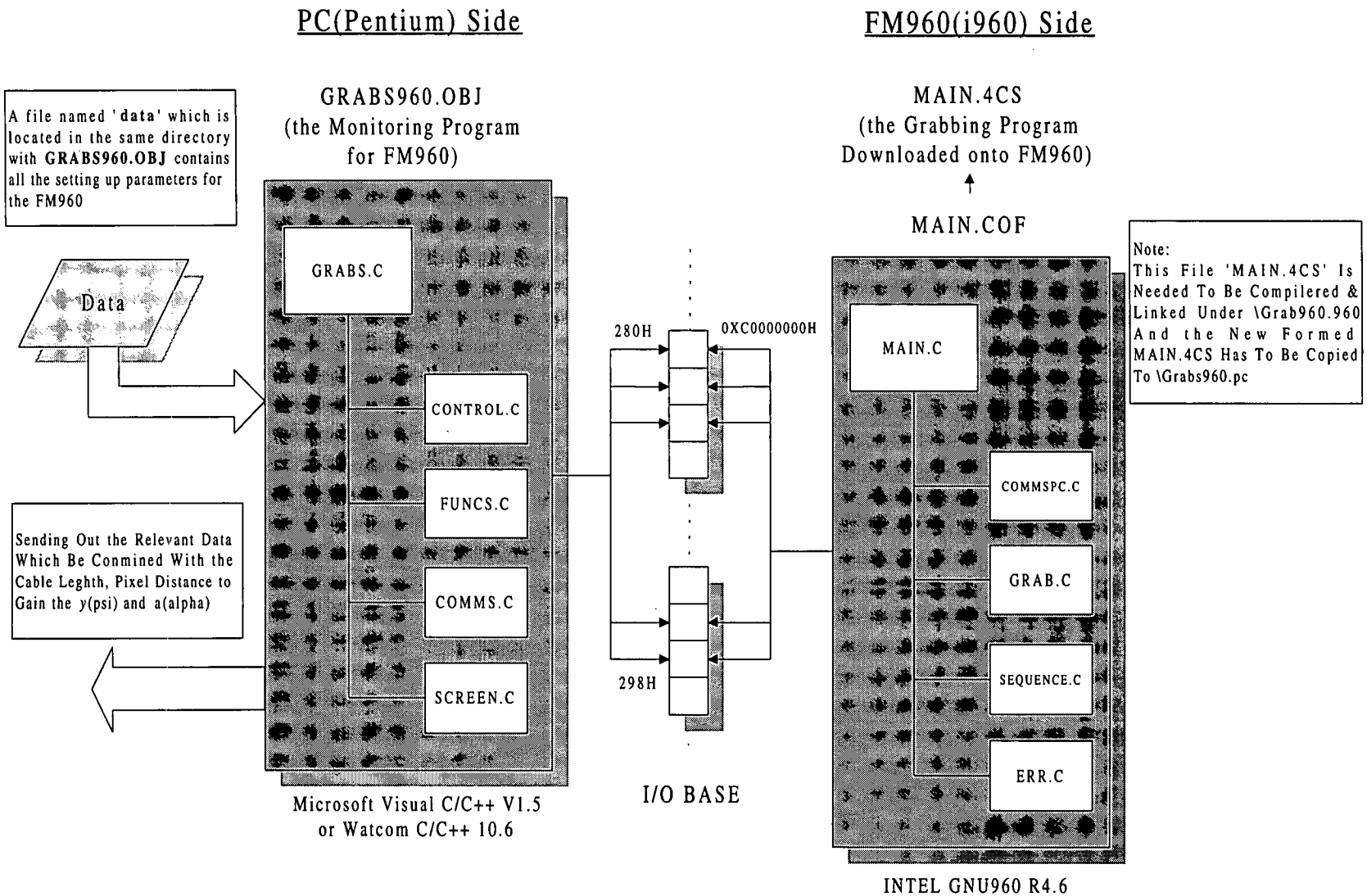


Figure 5.55 The structure of software setup

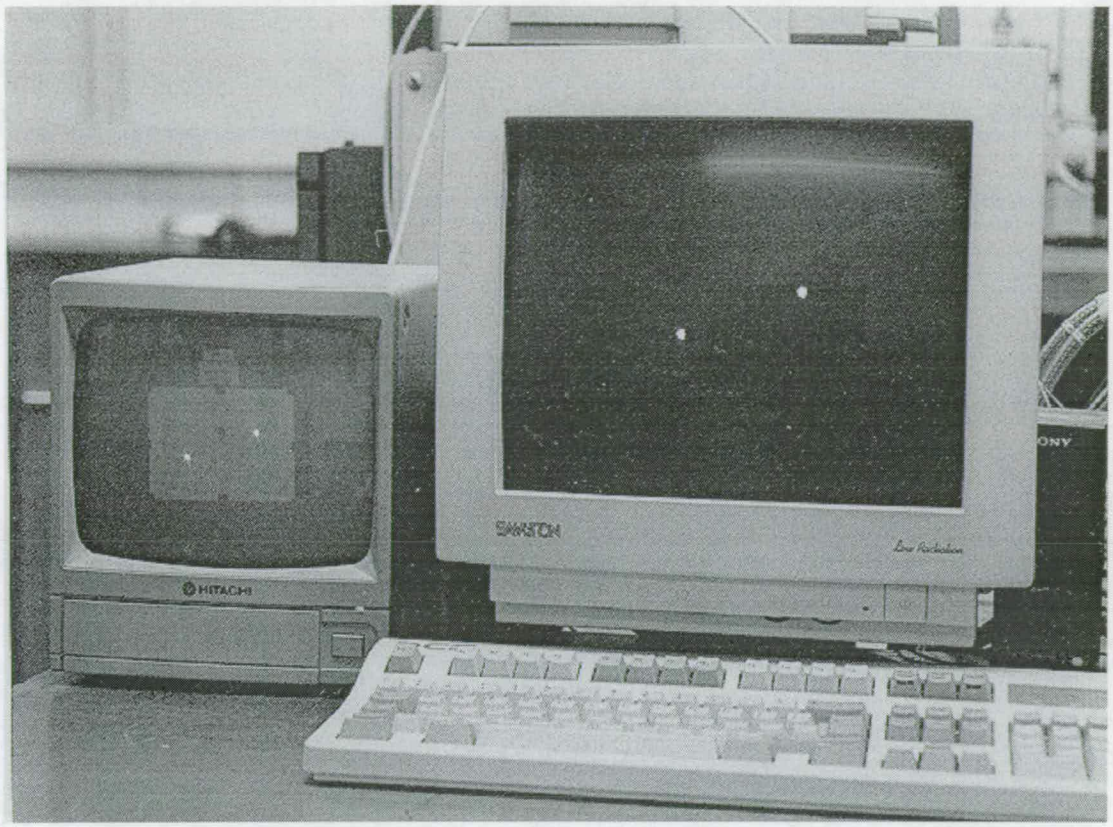


Figure 5.56 Two lighting sources appear as two distinct clusters of pixels

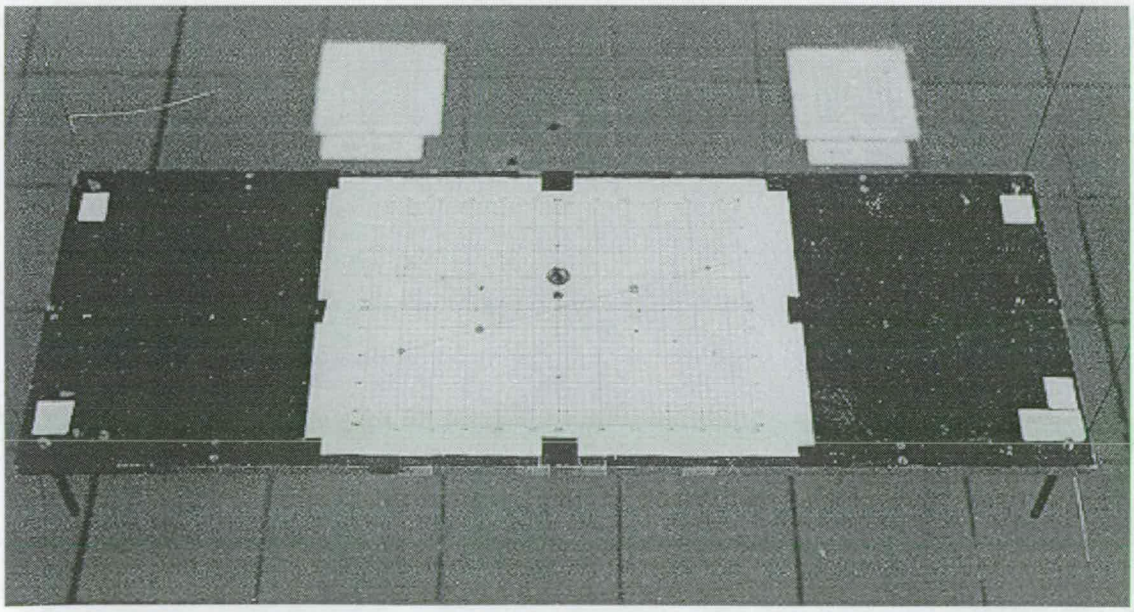


Figure 5.57 A mesh map (for calibration) is affixed on the spreader

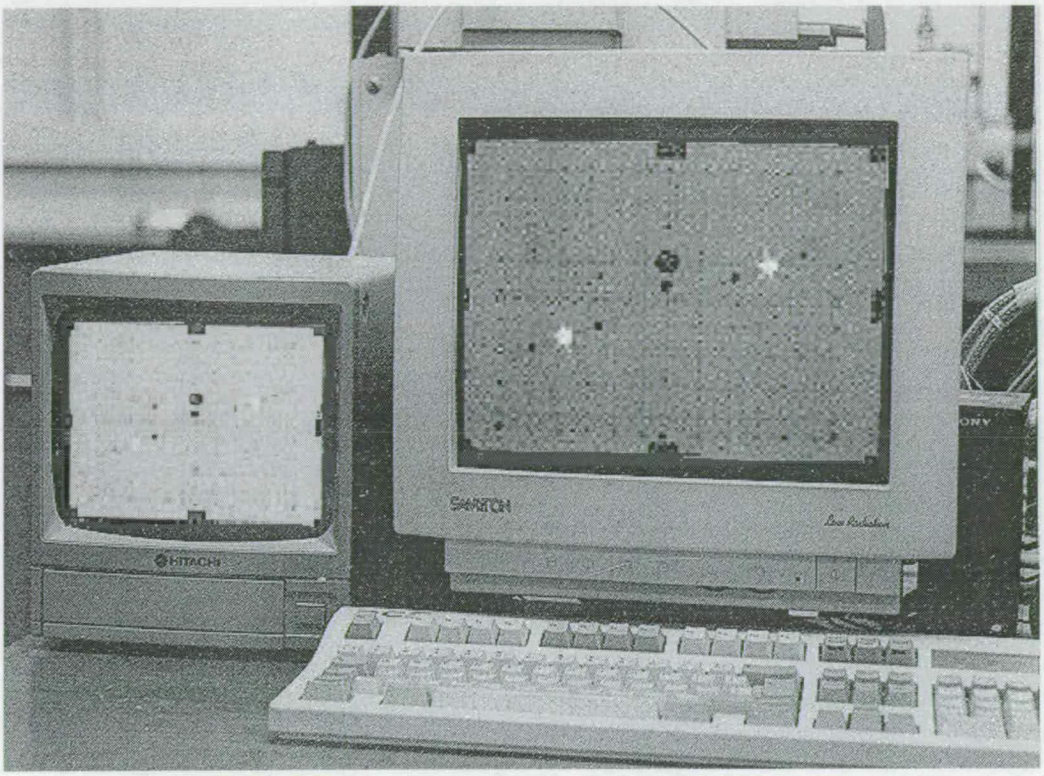


Figure 5.58 A 256 gray-scale image shown on the monitor

Chapter 6

The Crane Application Programming Interface (CAPI) for Real-Time Implementation of the Experimental Rig Control System

6.1 Introduction

In recent years there has been a large and growing demand for real-time simulation and control technology both in academia and industry. In particular, real-time systems, supported by the ability to include the evaluation of feedback control strategies, are finding increasing use and acceptance for tasks such as operator training, fault detection and investigation, and the safe testing of prototype systems. With the current trend towards more complex control strategies increased power is of paramount importance, but the successful management of this to best effect is a very complex general problem.

Commercially available simulation tools have been continuously updated as technology has improved, from FORTRAN based routines, such as TSIM (Rogers [66]), through to more powerful icon-based menu driven packages such as *SIMULINK* (SIMULINK [83]) in *MATLAB* (MATLAB [82]). Although these simulation tools have become more powerful and 'user friendly', they all suffer from portability limitations imposed by the computing platforms (hardware) on which they are implemented. During the last few years there has been an increasing interest in the automation of processes in general from the results of system design directly to hardware implementation. The driving force behind this development is to generate better consistency between designed and implemented control strategies (Torp [80]). Therefore the ultimate goal is to provide technologies capable of bridging the gap between the simulation models and the machine codes used to control experimental equipment without the need to resort to separate re-programming.

As described in Chapter 4 the design of the experimental rig involves different I/O devices, such as those dealing with the analogue outputs to the motor drive modules, the counter inputs from the shaft encoders, or the hardware relating to the data stream sent by the framegrabber etc. To implement a control strategy for such a complex system by means of traditional programming requires an understanding of the specific details of the hardware configuration as well as the source codes for the visual sensing system and the necessary function libraries for each I/O device. This approach is clearly too time-consuming and could be considered to be unnecessarily complicated. Therefore, in order to reduce the amount of time required for system communications development and to maximise resources available for control strategy implementation, a rather more efficient programming environment is investigated in this chapter. In section 6.2 a Crane Application Programming Interface (CAPI), capable of integrating the simulation results and hardware specification, is proposed. In section 6.3 measurement of the system parameters (such as the equivalent moment of inertia for each axis) are presented. Also, an implementation of *torque-to-voltage conversion*, which is capable of converting the required torque at the actuators to equivalent voltage input to the drive modules (in section 4.3.2) is discussed.

6.2 The Crane Application Programming Interface (CAPI)

In the laboratory environment, just as in the industrial one, process control hardware from many different manufacturers is often used. This presents potential difficulty further down the line in integrating all the devices together. Because the simulation of control strategies is traditionally carried out separate to its eventual implementation, the code used for each can appear very different and unrelated. Partial, or complete, re-writing of the hardware control programs is often necessary if a new control strategy is implemented (Rogers [66]). However there is an obvious advantage if these two stages can be merged together into one easy-to-use interface environment (Tang [75]). In such a situation the user should be purely concerned with design and refinement of the control strategy, leaving the interface to produce the compatible machine codes, automatically, to link the various hardware devices.

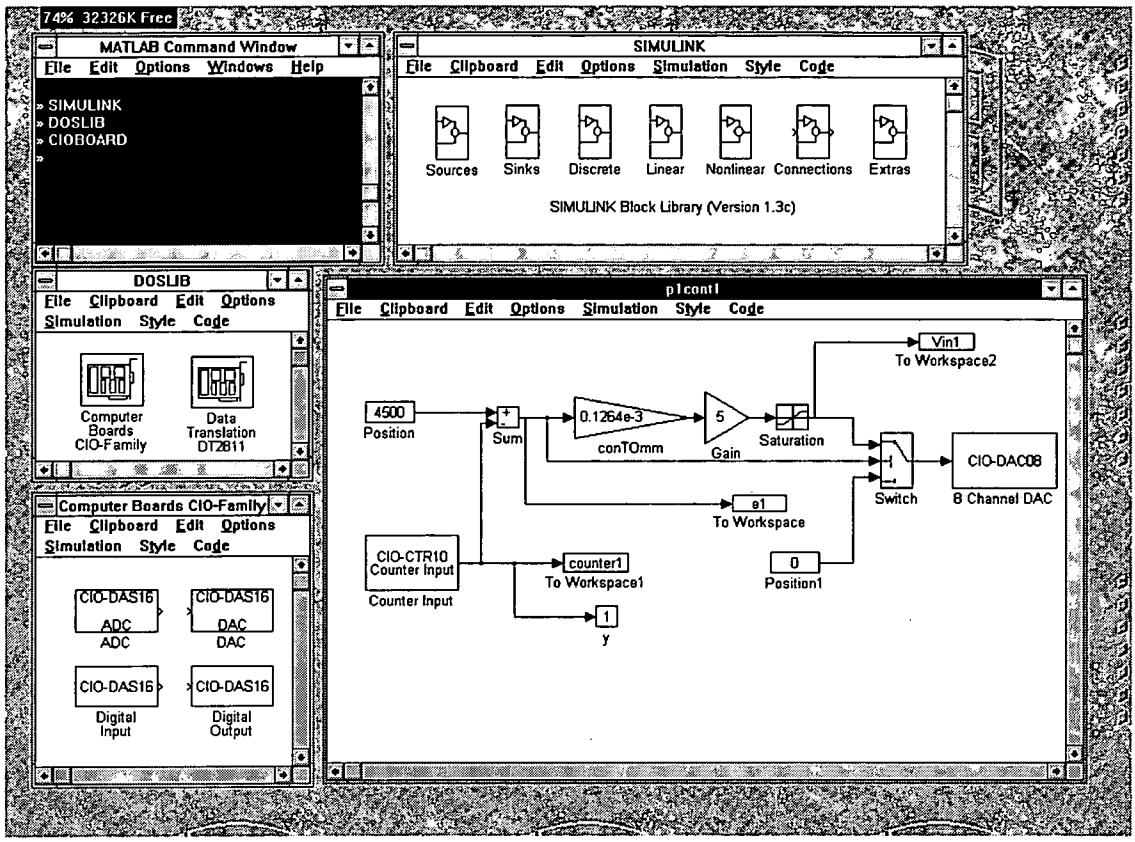


Figure 6.1 The Crane Application Programming Interface (CAPI) for the experimental rig

In order to make the most of the simulation results from *SIMULINK* in Chapter 3 a *MATLAB* accessory toolbox known as the *Real-Time Workshop* [84] is employed in the CAPI. Therefore, the entire *SIMULINK* model can be transferred to an implementation program to control directly the actuators of the system (in this case, the drive motors,) and also to receive the signals from the measuring sensors (the shaft encoders and the visual sensing system). So it is clear that this approach means that the simulation model itself becomes the actual controller of the system. This section is dedicated to describe such an interface (based on a typical windowing system such as the example of Figure 6.1) and the procedure required to develop an appropriate program to control the experimental rig. In section 6.2.1 the development procedure using the CAPI is proposed. In order to extend the capacity of this integrated environment, the techniques for creating icon-based *device driver blocks* (which are directly linked to the physical I/O devices by use of the S-function (System function) structure in *SIMULINK*) are described in section 6.2.2.

6.2.1 The Development Procedure

Basically, the development procedure of a system model using this interactive environment can be shown in Figure 6.2.

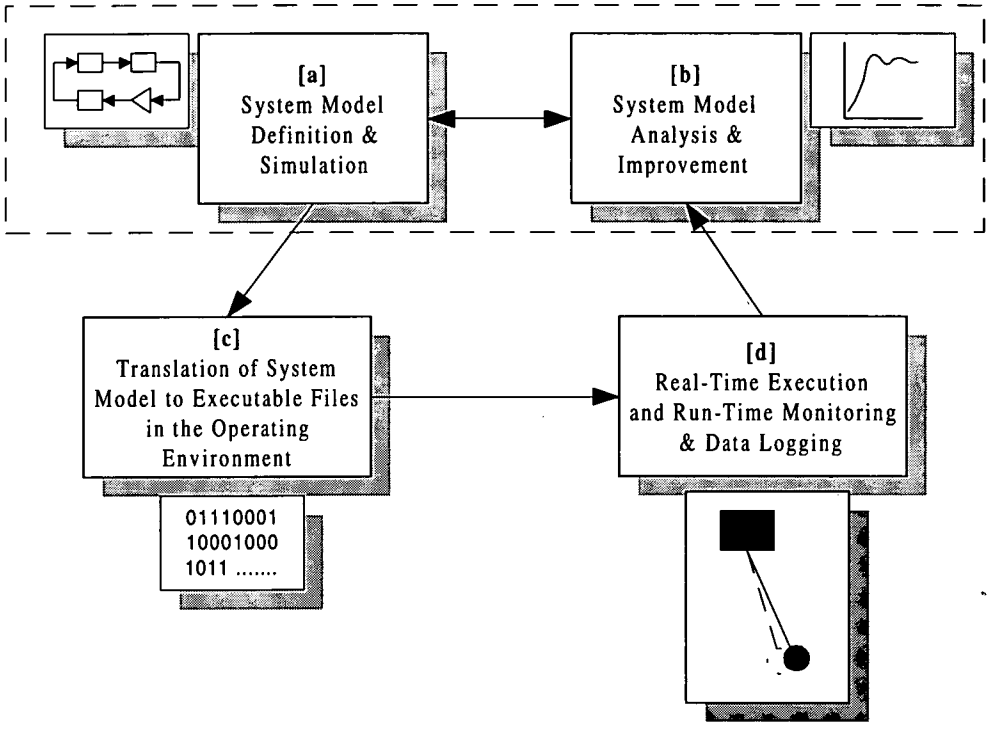


Figure 6.2 The development procedure

[a] System Model Definition & Simulation

The dynamics of a system (or a plant) model can be constructed in the icon-based environment, as shown in Figure 6.1, where the model is created by manipulating different block diagram elements representing different functions, such as differentials, integrals, sums etc. Once a model is defined, different numerical methods, such as Euler or Runge-Kutta (Kreyszig [46]), can be employed to simulate the dynamical behaviour of the system by progressively solving the differential equations etc. as appropriate. In the case of the experimental rig, the governing equations (3.25), (3.29), (3.33), (3.38), (3.43), derived earlier in section 3.2, are used in this stage. The results can be verified by plotting representative graphs or they can be passed to *MATLAB* for further analysis.

Once the model is completed the proposed control strategy, such as Feedback Linearization Control described in section 3.3 can be implemented within the system. The relevant parameters of the block diagrams can be initialised to different sets of values in order to investigate different configurations of the controller.

[b] System Model Analysis & Improvement

By using the powerful computational facilities of *MATLAB*, the results from stage [a] can be imported into in order to obtain better interpretation of performance in terms of visualization. The outcome can then be used to modify the model constructed in *SIMULINK*.

The other function of *MATLAB* exploited here is the provision of a computing engine for *SIMULINK* for which *M-files* (*MATLAB* files) are used for the block diagrams in the system model. These text-based M-files are written in the syntax of *MATLAB* and can invoke facilities such as matrix computation into *SIMULINK*. This has the advantage of easy programming and can provide useful functions which are currently unavailable in *SIMULINK*. This enables the building up of the system model at the preliminary stage. However, M-file block diagrams significantly slow down the overall speed of simulation, because of the fact that they need to be interpreted and compiled every time the simulation process encounters them (Technical [76]). Furthermore blocks written in the M-file format can *not* be used to generate C language-based control programs. Therefore, another computational element, called an *S-function*, has to be employed to rectify this disadvantage. The structure and generation of S-functions will be discussed further in section 6.3. The inter-relationship between these two types of block diagrams, *SIMULINK* and *MATLAB*. is depicted in Figure 6.3.

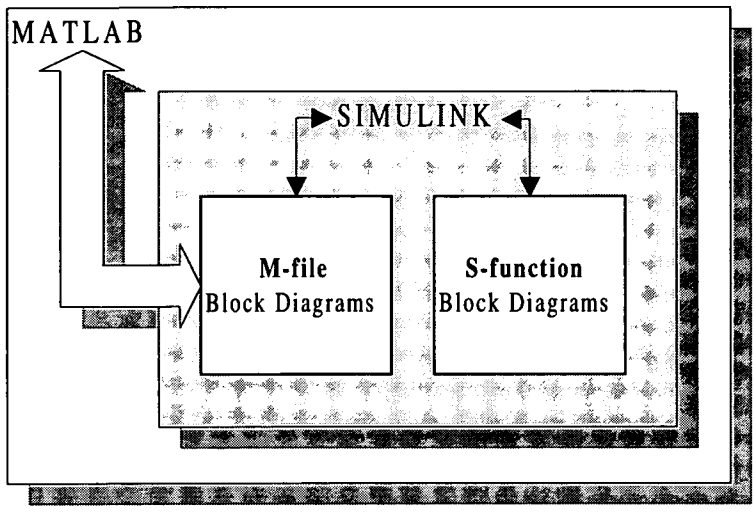


Figure 6.3 The inter-relationship between different types of block diagrams

[c] Translating the System Model to Executable Files in the Operating Environment

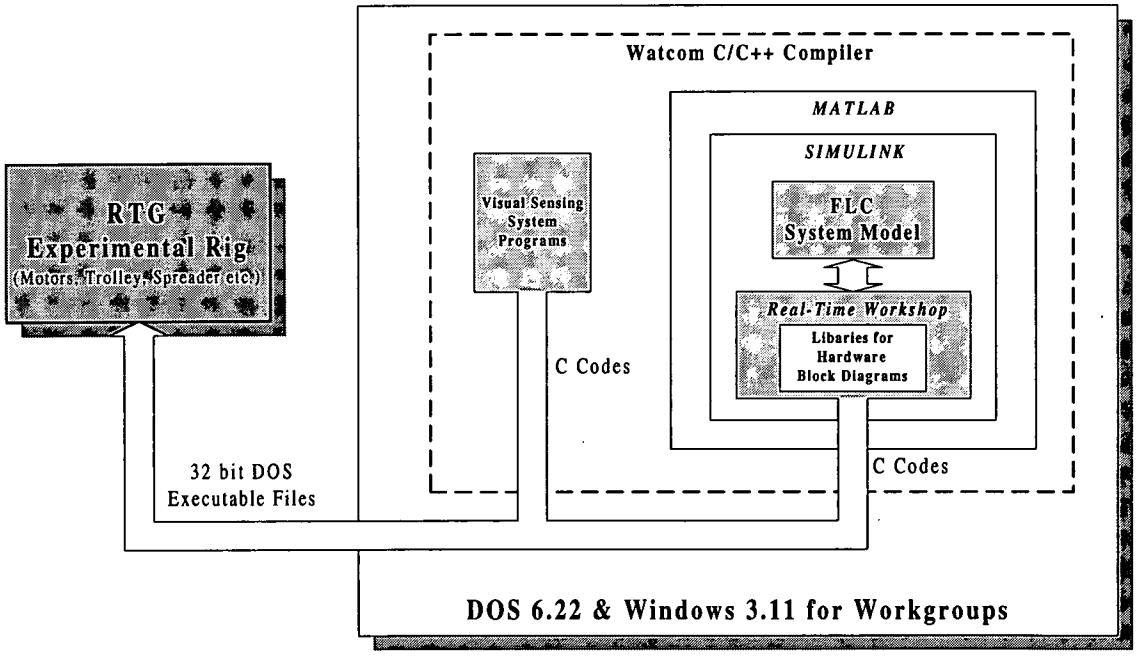


Figure 6.4 The relationship between the system model and *Real-Time Workshop*

By means of processes [a] and [b], the dynamics and control strategy for a system can be converted into a model constructed by *SIMULINK* block diagrams. However, to physically control the experimental rig this model has to be translated into programs which are understood by the host computer and control circuitry. This can be achieved

by using the *Real-Time Workshop* toolbox to generate C-language source codes from the system model and then to invoke the compiler (provided that all the device dependent blocks diagrams are available for all the hardware devices). In the case of this experimental rig a Watcom C/C++ Compiler (Watcom [85]) is used to produce a 32-bit executable file for the DOS operating system of the host computer. The relationship between the system model and the *Real-Time Workshop* is described in Figure 6.4.

[d] Real-Time Execution and Run-Time Monitoring & Data Logging

From the previous process, the executable file of a system model can be obtained. This 32-bit DOS file (normally with the extension name **.exe**) can be executed either in a DOS window within (say) Windows 3.11, or within the native DOS environment. Users can compare the physical behaviour of the experimental rig to the simulation carried out in processes [a] & [b]. Also because of the utility of run-time monitoring and data logging the results can be represented graphically using *MATLAB* and hence modelling corrections made to correct or refine the original design of the system model, or indeed the control strategy itself.

6.2.2 Device Driver Blocks and S-Functions (System functions)

As stated earlier, it is easier to implement a control strategy on some test-bed if the device-dependent parts of the codes, such as analog I/O and inter-task communications are *encapsulated* in software block diagrams. To do this it is necessary to identify all device-dependent codes needed for the platform and then to specify a common structure between these block diagrams and the main interface. Therefore when it is required to convert a system model into an executable file the interface can proceed to compile them without difficulty.

As already stated, in order to make these *device driver blocks* (*Real-Time Workshop* [84]) communicate with external hardware on the experimental rig a computational element called an *S-function* (system function (*SIMULINK* [83])) has to be used. Generally speaking the structure of S-functions includes:

- Initialisation of the S-function.
- Initialisation of the I/O device.
- Calculation of the blocks outputs according to the type of driver being implemented; for example:

  Reading values from an I/O device and assigning these values to the block's output (if is an ADC (analogue-to-digital converter))

  Writing values from the block's input, to an I/O device (if it is a DAC (digital-to-analogue converter).
- Terminating the program (e.g., zeroing the DAC output).

An example program for building up the DAC device driver block used in the experimental rig is shown in Appendix D. Each device driver block has its own dialogue box which is used to configure its particular options. Basically the dialogue boxes have the similar layout to the one shown in Figure 6.5, in which the following parameters for the control device (in this case, a DAC) can be specified:

- The base I/O address to match the hardware configuration on the host computer. In this case, 0x300 is set.
- The output voltage reference to set the range of the output voltage from the DAC. In this case, a range from 0 to 10 volts is set.
- The number of channels and effective number of bits to specify the number of channel outputs simultaneously through this DAC device and their resolution. In this case, one channel is set because only the #1 x-axis motor is controlled by this block. '12' is set for the resolution, according to the specification of the DAC.
- The sampling time to set this to match the required step size for the model.
- Access hardware during the simulation to specify whether or not this device driver block is activated in the executable file.

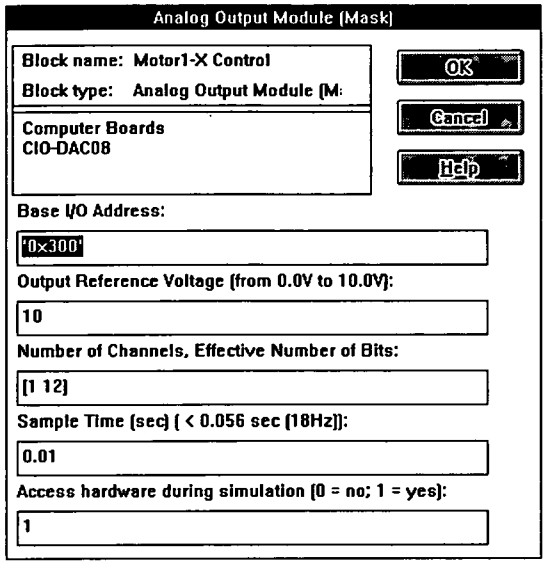


Figure 6.5 The device driver block for the #1 x- axis motor

As shown in Figure 6.6, a library of device driver blocks is created for all the I/O devices of the experimental rig. Also several utility blocks, such as matrix inversion, which were originally written in the M-file format, are re-programmed to comply with the common format of the CAPI.

One of the important features of creating block diagrams using the S-function approach is the *compatibility* and *expandability* of such diagrams to other libraries of block diagrams in *SIMULINK*. So, new block diagrams can be manipulated in exactly the same way as standard ones. This allows later designers to take advantage of earlier effort contributed by their predecessors. Moreover, perhaps the most important feature of this is that all device dependent codes can effectively be *isolated* from the end user, appearing purely as simple block diagrams. This dramatically reduces the time required to master all the specific hardware I/O configurations so that more effort can be concentrated on implementing control strategies.

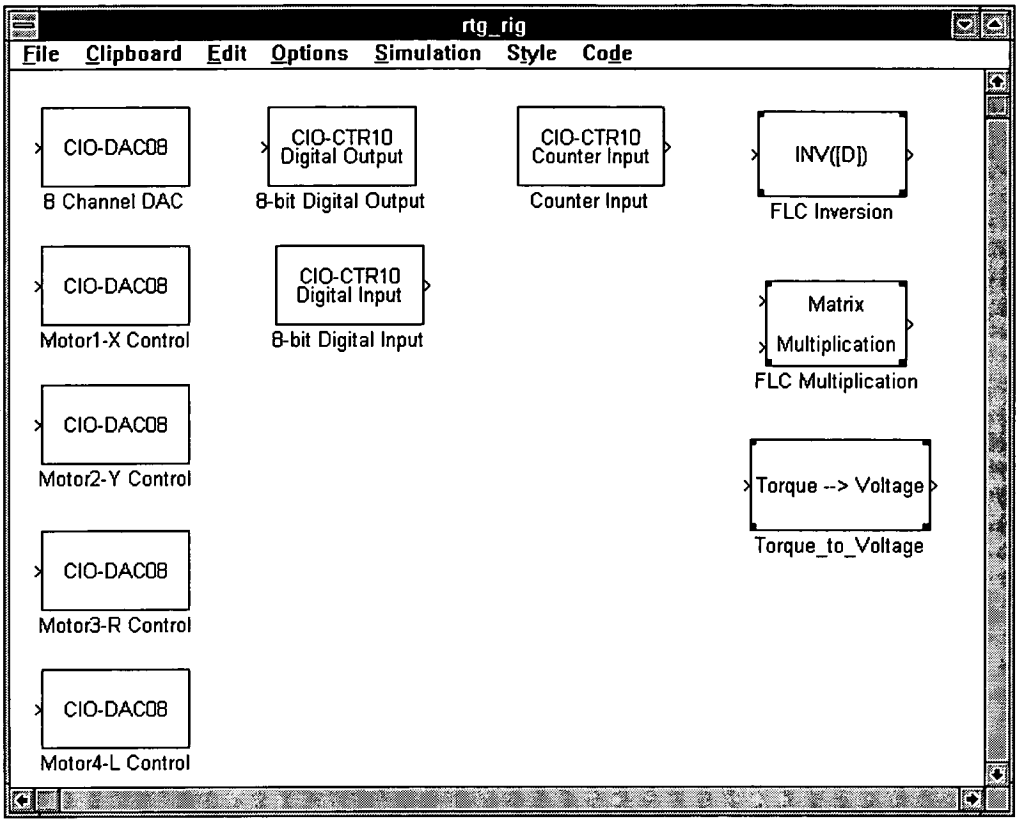


Figure 6.6 The library of specially created device driver blocks for the experimental rig

6.3 Torque-to-voltage Conversion and the Measurement of System Parameters for the Experimental Rig

6.3.1 Torque-to-voltage conversion

After establishing the CAPI, as described in the last section, the next step is to implement a control strategy for the experimental rig and it is considered to be reasonable to make use of the FLC (Feedback Linearization Control) model discussed in Chapter 3.

From equation (3.48),

$$\underset{\sim}{\tau} = \begin{bmatrix} \frac{F_x}{m_s} \\ \frac{F_y}{m_s} \\ \frac{F_l}{m_s} \\ 0 \\ 0 \end{bmatrix} \qquad \text{------- [(3.48)]}$$

it can be seen that the control inputs to the FLC system model are, dimensionally, accelerations. For example, if torque, T_x, is provided by the x -axis actuator (#1 motor) with drum radius, r_1, then,

$$T_x = F_x r_1 \qquad \text{------- (6.1)}$$

$\Rightarrow$

$$\tau_x = \frac{T_x}{m_s r_1} \qquad \text{------- (6.2)}$$

From equation (6.2) it is understood that the torque exerted by a motor is the actual variable control input of the experimental rig at that point. However, as mentioned in section 4.3.1, the motors are actually speed (or voltage) controlled. Thus it is

necessary to find a relationship between the input voltage to a motor drive module and the torque output from that relevant motor.

The representative free-body diagram for a motor-drive axis is as shown in Figure 6.7,

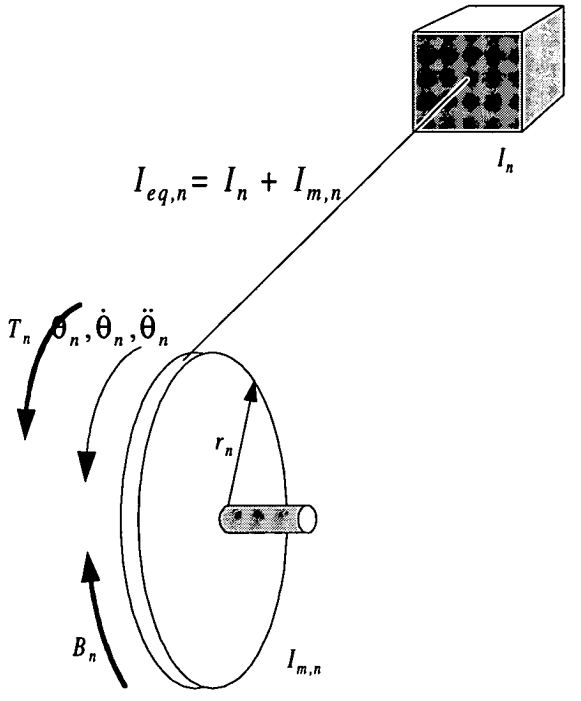


Figure 6.7 Free-body diagram for a motor-drive axis

the equation of motion is,

$$I_{eq,n}\ddot{\theta}_n + B_n = T_n \qquad \text{------- (6.3)}$$

where

$I_{m,n}$: the mass moment of inertia of the actuator including the motor, gear box, drum. n = 1, 2, 3, 4.

I_n : the mass moment of inertia of the rest of the moving components of the drive on that axis. n = 1, 2, 3, 4.

$I_{eq,n}$: the equivalent mass moment of inertia of the axis. $I_{eq,n} = I_n + I_{m,n}$.

n = 1, 2, 3, 4.

B_n: the combined damping and frictional torque term.
n = 1, 2, 3, 4.

r_n: the radius of the drum.
n = 1, 2, 3, 4.

T_n: the torque generated by the actuator.
n = 1, 2, 3, 4.

If the mechanism in Figure 6.7 reaches a *steady-state* at t = t_1, with the angular velocity of the actuator, $\dot{\theta}_n = \dot{\theta}_{n,t1}$, equation (6.3) becomes,

$$B_{n,t1} = T_{n,t1} \qquad \text{-------} \quad (6.4)$$

At and after t_1, $T_{n,t1}$ is only required to overcome the damping and frictional torque.

From Figures 4.4, 4.6, 4.8 and 4.10 in Chapter 4 it can be seen that the relationship between the angular velocity of the actuator and the V_{in} from the host computer is,

$$\dot{\theta}_n = K_{\dot{\theta},n} V_{in(n)} + \dot{\theta}_{offset,n} \qquad \text{-------} \quad (6.5)$$

where $K_{\dot{\theta},n}$ and $\dot{\theta}_{offset,n}$ are both constant.

Now, if the desired torque, $T_{d,n}$, is requested by the host computer, the required torque, T_n, from the relevant actuator should be,

$$T_n = T_{d,n} + B_{n,t1} \qquad \text{-------} \quad (6.6)$$

Because the mechanism in Figure 6.7 is no longer in steady-state conditions (i.e. it is accelerating), then substituting equation (6.6) into equation (6.3), means that,

$$I_{eq,n}\ddot{\theta}_n = T_{d,n} \qquad \text{------- (6.7)}$$

Where the angular acceleration, $\ddot{\theta}_n$, can be assumed to be given by,

$$\ddot{\theta}_n = \frac{\dot{\theta}_{t1+\Delta t} - \dot{\theta}_{t1}}{\Delta t} \qquad \text{------- (6.8)}$$

where Δt is the period of time needed for the actuator to reach another steady-state with angular velocity , $\dot{\theta}_{t1+\Delta t}$.

From equation (6.7),

$$\ddot{\theta}_n = \frac{T_{d,n}}{I_{eq,n}} \qquad \text{------- (6.9)}$$

On substituting equation (6.8) into equation (6.9),

$$\dot{\theta}_{t1+\Delta t} - \dot{\theta}_{t1} = \frac{T_{d,n}}{I_{eq,n}}\Delta t \qquad \text{------- (6.10)}$$

Also from equation (6.5) it can be seen that,

$$\dot{\theta}_{n,t1\Delta t} = K_{\dot{\theta},n} V_{in(n,t1+\Delta t)} + \dot{\theta}_{offset,n} \qquad \text{------- (6.11)}$$

$$\dot{\theta}_{n,t1} = K_{\dot{\theta},n} V_{in(n,t1)} + \dot{\theta}_{offset,n} \qquad \text{------- (6.12)}$$

So, substituting equations (6.11) and (6.12) into equation (6.10) gives the following,

$$V_{in(n,t1+\Delta t)} = V_{in(n,t1)} + \frac{T_{d,n}}{I_{eq,n} K_{\dot{\theta},n}} \Delta t \qquad \text{------- (6.13)}$$

Equation (6.13) shows a relationship between the desired torque, $T_{d,n}$, and the voltage, $V_{in(n,t1+\Delta t)}$, required to obtain it. This is converted into a device-driver block using the procedure discussed in section 6.2.2. In the next sub-section the equivalent mass moment of inertia, $I_{eq,n}$, is investigated.

6.3.2 The principal mass moment of inertia of the experimental rig

In order to implement the FLC model and make use of the torque-to-voltage conversion discussed in the last section, it is necessary to measure the mass moments of inertia of the x-, y-, and l- axes (respectively) of the experimental rig.

(i) Measurement of the Combined Damping and Frictional Torque term, B_n

First the combined damping and frictional torque term, B_n, in equation (6.3) is assumed to be reasonably represented by,

$$B_n = C_n \dot{\theta}_n + T_{f,n} \qquad \text{------- (6.14)}$$

where the damping coefficient, C_n and the frictional torque $T_{f,n}$ are both taken to be constants.

From Figure 6.7 a mass, m, is attached to the free end of the cable wrapped around the pulley (no-slip conditions assumed throughout), the position as shown in Figure 6.8, then the equation of motion is,

$$I_{system} \ddot{\theta}_n + B_n = T_n \qquad \text{------- (6.15)}$$

where

$$I_{system} = I_{eq,n} + mr_n^2 \quad \text{and} \quad T_n = mgr_n$$

So equation (6.15) becomes,

$$(I_{eq,n} + mr_n^2)\ddot{\theta}_n + (C_n\dot{\theta}_n + T_{f,n}) = mgr_n \qquad \text{-------} \quad (6.16)$$

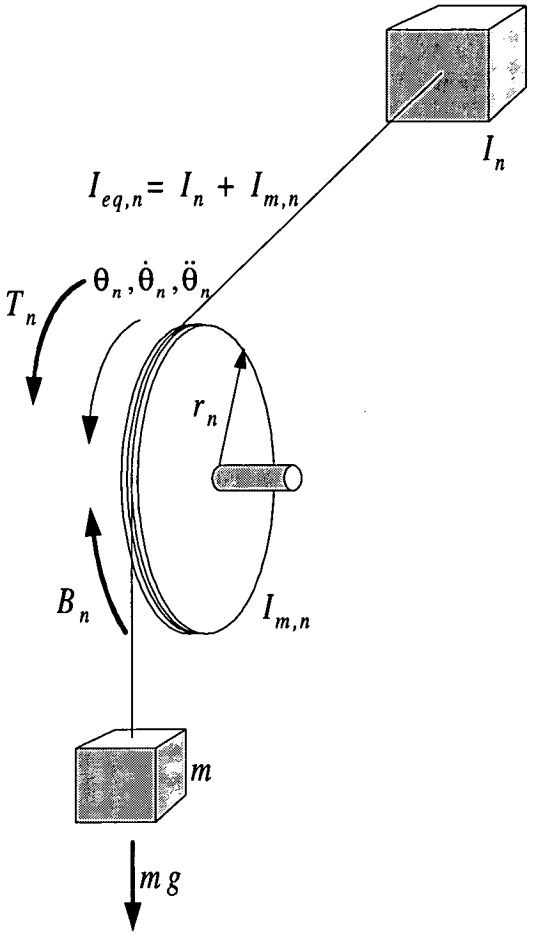


Figure 6.8 The mechanism for measuring the combined torque term B_n

$T_{f,n}$ can be measured by increasing gradually the weight, mg, until the mechanism just starts to move at a very low speed (< 0.1 radian/sec) so that the effect of $I_{eq,n}\ddot{\theta}_n$ and $C_n\dot{\theta}_n$ in equation (6.16) can largely be ignored. Because $T_{f,n}$ is a *dynamic* frictional torque, the mechanism in Figure 6.8 must be pushed slightly each time to overcome the *static* frictional torque threshold. The measured results for $T_{f,n}$ are shown in Table 6.1.

	x- axis (#1)	y- axis (#2)	*l*- axis(#4)
Frictional Torque, T_f	3.679 Nm ($T_{f,1}$)	5.003 Nm ($T_{f,2}$)	2.174 Nm ($T_{f,4}$)

Table 6.1 The measured result for $T_{f,n}$

By further increasing the weight, *mg*, beyond the frictional torque, the mechanism in Figure 6.8 obviously starts moving appreciably. While the velocity builds up the damping torque, C_n, increases accordingly and then reaches a steady-state at the final steady angular velocity. Therefore, from equation (6.16),

$$C_n\dot{\theta}_n = mgr_n - T_{f,n} \qquad \text{-----} \quad (6.17)$$

where $I_{eq,n}\ddot{\theta}_n = 0$ because $\dot{\theta}_n$ is constant when the system reaches steady-state speed.

Figures 6.9, 6.10, 6.11 (lumped together at the end of the chapter) show the results of the experiments applied to the x-, y- and l- axes, respectively. First-order *polynomial curve fitting* (Kreyszig [46], Mathworks [82]) is used to find a polynomial that fits the angular velocity, ω ($\omega \equiv \dot{\theta}$), in a least-square sense. The relatively small dc offsets in the polynomials in Table 6.2 tends to suggest that the measurement of frictional torque (in Table 6.1) is adequate and that the assumption in equation (6.14) is acceptable. By adjusting the offsets to the frictional torque, the final result for B_n is summarized in Table 6.3.

	Damping Torque (Nm)	
x- axis (#1)	$0.4586\dot{\theta}_1$ - 0.03495	------- (6.18)
y- axis (#2)	$2.244\dot{\theta}_2$ - 0.0112	------- (6.19)
l- axis(#4)	$1.126\dot{\theta}_4$ + 0.01565	------- (6.20)

Table 6.2 First-order polynomial curve fitting for damping torque/angular velocity

	Damping Coefficient, C_n Nms	Frictional Torque, $T_{f,n}$ Nm
x- axis (#1)	0.4586	3.644
y- axis (#2)	2.244	4.991
l- axis(#4)	1.126	2.190

Table 6.3 The damping coefficient, C_n, and frictional torque $T_{f,n}$

(ii) Measurement of the Equivalent Mass Moment of Inertia, $I_{eq,n}$

On letting $\dot{\theta}_n = \omega$, equation (6.16) can be re-arranged as follows,

$$(I_{eq,n} + mr_n^2)\dot{\omega} + C_n\omega = mgr_n - T_{f,n} \qquad \text{------- (6.21)}$$

Equation (6.21) is a first-order non-homogeneous ordinary differential equation in ω, which has a solution of the following form,

$$\omega = \frac{c}{b} + Ae^{-\frac{b}{a}t} \qquad \text{------- (6.22)}$$

where

$$a = I_{eq,n} + mr_n^2 \qquad \text{------- (6.23)}$$

$$b = C_n \qquad \text{------- (6.24)}$$

$$c = mgr_n - T_{f,n} \qquad \text{------- (6.25)}$$

If the mechanism in Figure 6.8 starts at ω(0) = 0, equation (6.22) becomes,

$$\omega = \frac{c}{b}\left(1 - e^{-\frac{b}{a}t}\right) \qquad \text{------- (6.26)}$$

A program has been written to record the readings from the shaft encoders of the position and frequency of the motors and this data file can then be imported into *MATLAB* in order to make use of its resident numerical facilities for curve fitting to the form of equation (6.26). There are two different weights, mg, attached to each axis for the experiments and the results for each pair of experiments are shown in Figures 6.11 to 6.16 inclusive. The weights chosen for each pair of experiments are arbitrary (okay as long as they are sufficiently different).

[a] The x- axis

(1) m = 12 kg, r_1 = 0.05 m.

The result of curve fitting on Figure 6.12 is,

$$\omega = 6.11\left(1 - e^{-0.425t}\right) \qquad \text{------- (6.27)}$$

By using equations (6.23) ~ (6.25), the equivalent moment of inertia, $I_{eq,1}$, is,

$I_{eq,1}$ = 0.832 kgm^2

C_1 = 0.367 Nms

(2) m = 13 kg, r_1 = 0.05 m.

The result of curve fitting on Figure 6.13 is,

$$\omega = 7.651\left(1 - e^{-0.385t}\right) \qquad \text{------- (6.28)}$$

By using equations (6.23) ~ (6.25), the equivalent moment of inertia, $I_{eq,1}$, is,

$I_{eq,1} = 0.8951$ kgm^2

$C_1 = 0.3571$ Nms

[b] The y- axis

(1) $m = 22$ kg, $r_2 = 0.05$ m.

The result of curve fitting on Figure 6.14 is,

$$\omega = 2.229\left(1 - e^{-1.255t}\right) \qquad \text{------- (6.29)}$$

By using equations (6.23) ~ (6.25), the equivalent moment of inertia, $I_{eq,2}$, is,

$I_{eq,2} = 2.602$ kgm^2

$C_2 = 2.018$ Nms

(2) $m = 24$ kg, $r_2 = 0.05$ m.

The result of curve fitting in Figure 6.15 is,

$$\omega = 2.865\left(1 - e^{-1.052t}\right) \qquad \text{------- (6.30)}$$

By using equations (6.23) ~ (6.25), the equivalent moment of inertia, $I_{eq,2}$, is,

$I_{eq,2} = 2.8648$ kgm^2

$C_2 = 2.19$ Nms

[c] The *l*- axis

(1) m = 8 kg, r_4= 0.0375 m.

The result of curve fitting in Figure 6.16 is,

$$\omega = 2.586\left(1 - e^{-0.6888t}\right) \qquad \text{-------} \quad (6.31)$$

By using equations (6.23) ~ (6.25), the equivalent moment of inertia, $I_{eq,4}$, is,

$I_{eq,4}$ = 1.246 kgm^2

C_4 = 0.87 Nms

(2) m = 10 kg, r_4= 0.0375 m.

The result of curve fitting in Figure 6.17 is,

$$\omega = 4.068\left(1 - e^{-0.576t}\right) \qquad \text{-------} \quad (6.32)$$

By using equations (6.23) ~ (6.25), the equivalent moment of inertia, $I_{eq,4}$, is,

$I_{eq,4}$ = 1.2525 kgm^2

C_4 = 0.7328 Nms

In order to create substantial acceleration to get enough data to analyze before the mechanism in Figure 6.8 reaches steady-state, the weight, mg, has to be rather large compared to that used in measuring B_n in section 6.3.1.i. However, because of the limited span of the ẋ- and *l*- axes the experiments have to be terminated before they reach a true steady-state speed (in Figures 6.11, 6.12, 6.15, 6.16). This means that the curving fitting process needs to 'extrapolate' the rest of the curve from the available

data. However, the experiments for the y-axis (in Figures 6.14, 6.15) include part of the steady-state curve. Therefore the two test results for $I_{eq,2}$ in the (y-axis) show a better match with Table 6.3 in terms of the damping coefficient, C_n. It can also be seen that C_n is closer to the value in Table 6.3 with the lower added weight for the x- and l-axes. From this it is assumed to be reasonable to choose the mass equivalent moment of the inertia value, $I_{eq,n}$, from the result with the lower added weight attached on each axis, as shown in Table 6.4.

	x- axis (#1)	y- axis (#2)	l- axis(#4)
Equivalent moment of inertia, $I_{eq,n}$	0.832 kgm^2 ($I_{eq,1}$)	2.602 Nm ($I_{eq,2}$)	1.246 Nm ($I_{eq,4}$)

Table 6.4 The equivalent moment of inertia, $I_{eq,n}$,

6.4 Conclusion

For economic reasons, practical implementation of control algorithms often requires the use of existing control equipment that is not at all compatible with the system on which the software was initially developed and tested. Hence it is advantageous for the engineer implementing the software if the device-dependent parts of the code, such as analog I/O, inter-task communication and real-time scheduling, are *encapsulated* in a single software library. The Crane Application Programming Interface (CAPI) (Figure 6.1) described in section 6.2 provides a user-friendly environment in which end-users can concentrate on exploring the more fundamental issues of choice of control strategies, rather than spending significant amounts of time in programming I/O routines between physical devices. The development procedure shown in Figure 6.2 forms a closed-loop routine and by following this a system model can quickly be established, tested, and improved. This interactive environment can be used as a research tool for investigating different control strategies on the experimental rig on a real-time basis. The purpose is to ease the process of porting a control algorithm to a real-time platform and testing it on an experimental rig. As mentioned earlier the development process works closely with *SIMULINK* and *MATLAB*.

In section 6.2.2 a procedure for creating device-driver blocks is introduced. These blocks provide a method for extending the capacities of the interactive interface. Users who follow on can then create customized blocks associated with new devices and proceed to incorporate them as part of the CAPI.

In section 6.3 the torque-to-voltage conversion procedure for the actuators is discussed. This is required for the motor drive modules (which are basically driven by voltage), on the understanding that the FLC model uses torques as control inputs. Important parameters, such as the equivalent moment of inertia for the x-, y-, and *l*-axes, are also investigated by using a curve fitting method.

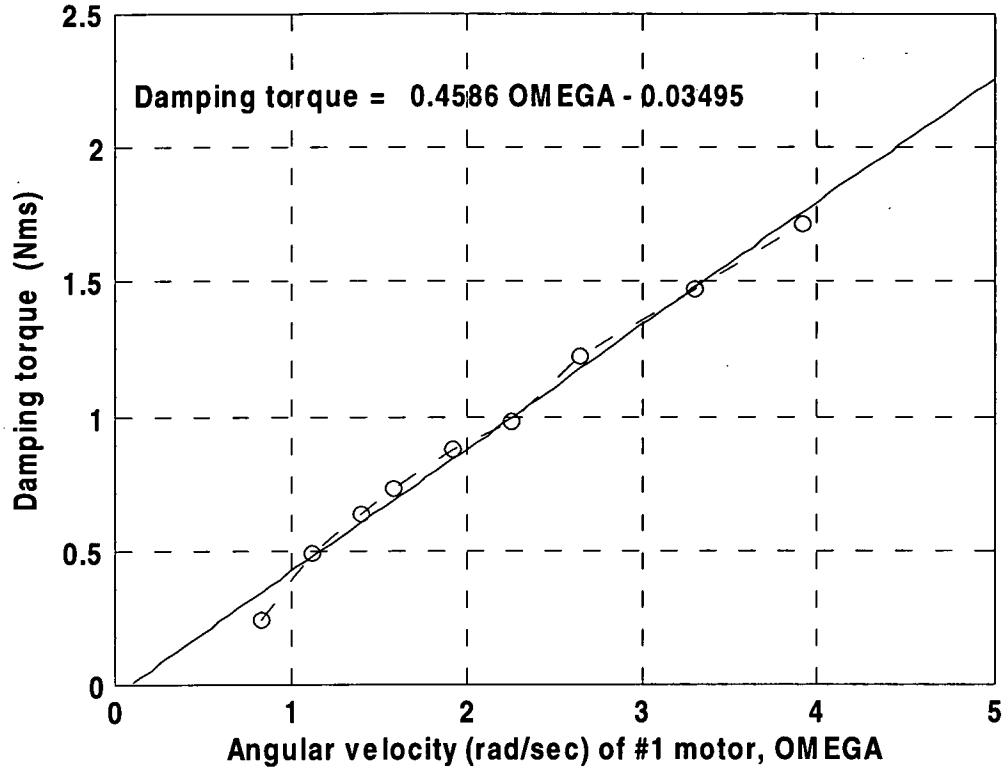


Figure 6.9 Damping torque for the x- axis

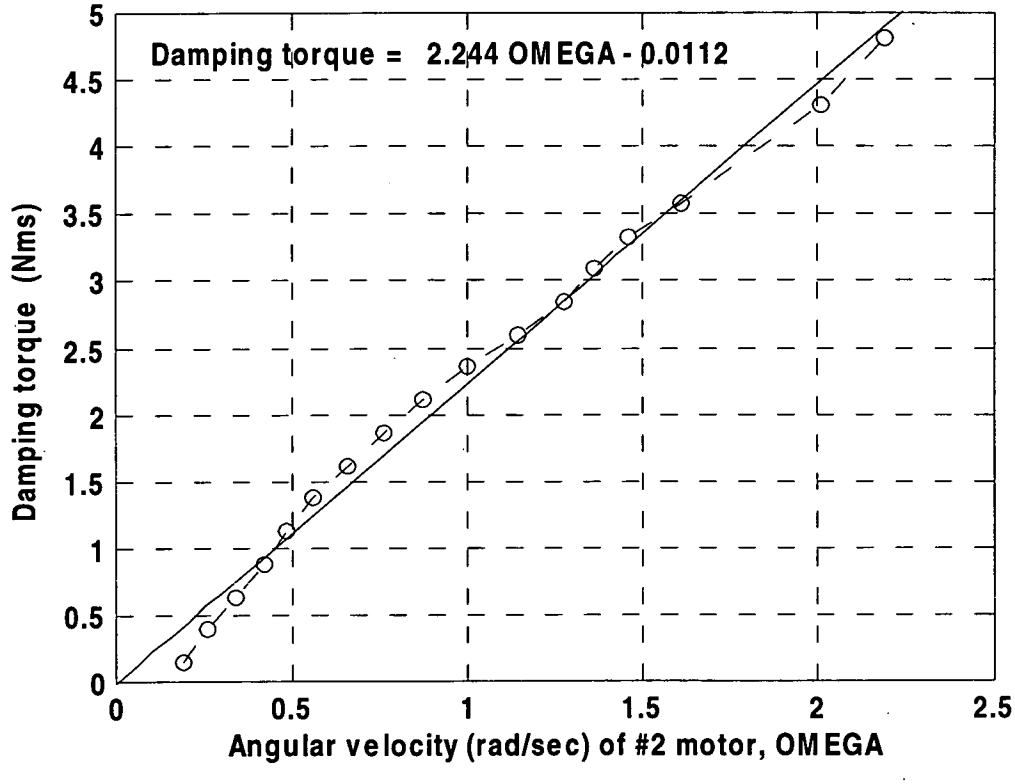


Figure 6.10 Damping torque for the y- axis

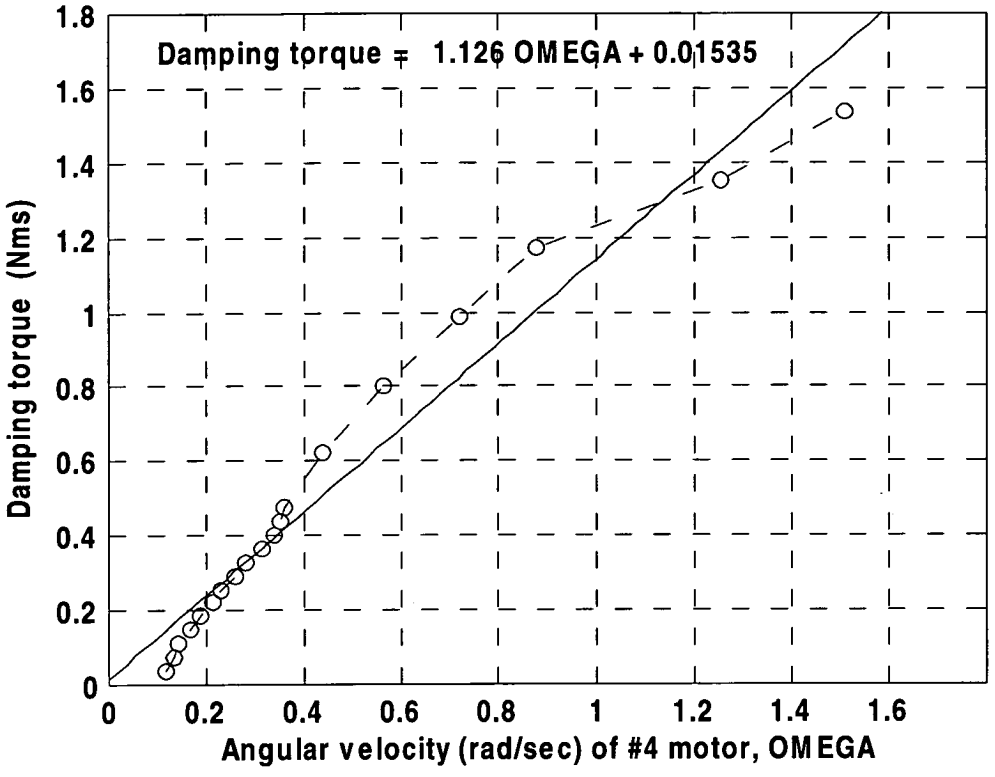


Figure 6.11 Damping torque for the *l*- axis

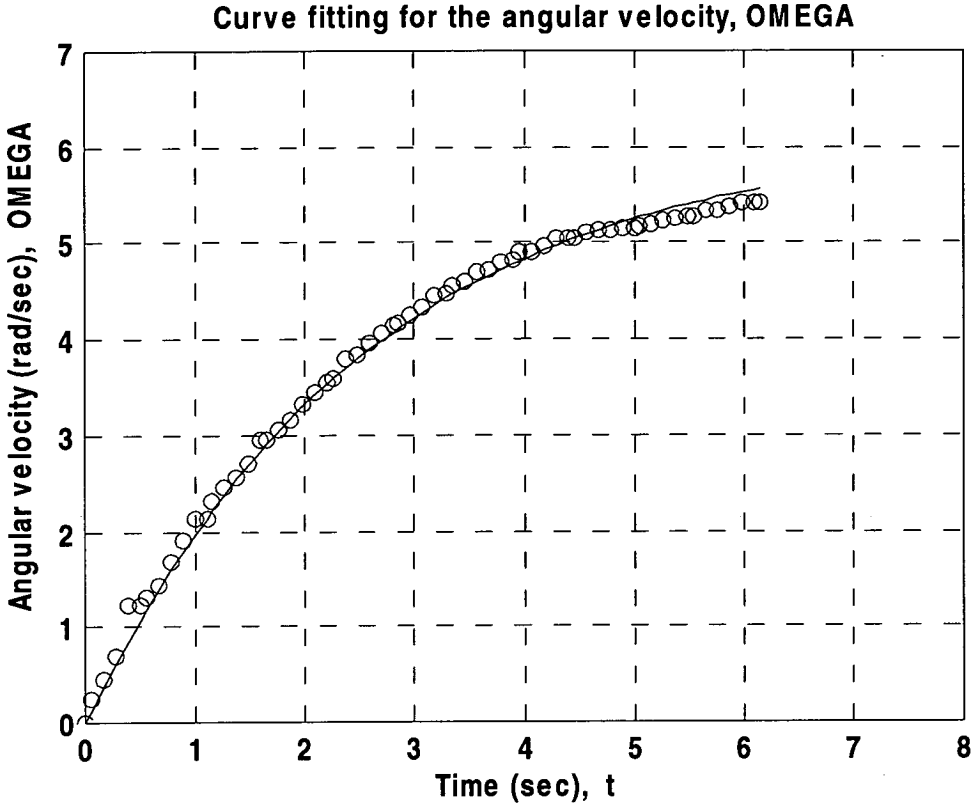


Figure 6.12 The curve fitting result for $I_{eq,1}$ with m = 12 kg.

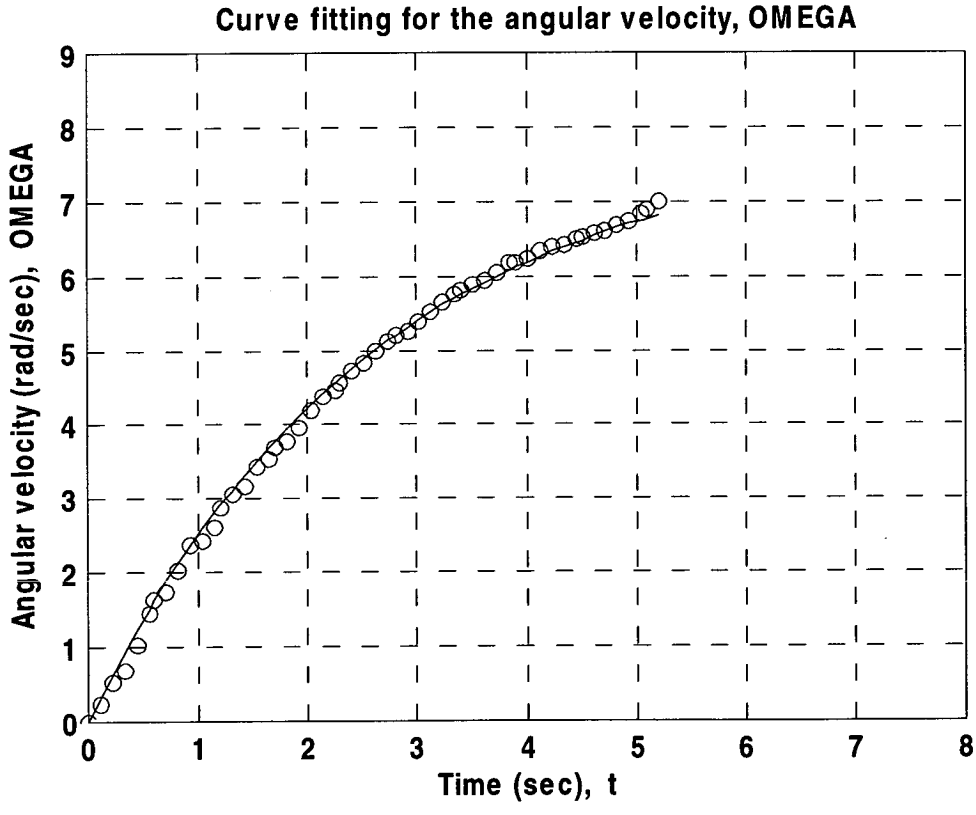


Figure 6.13 The curve fitting result for $I_{eq,1}$ with m = 13 kg.

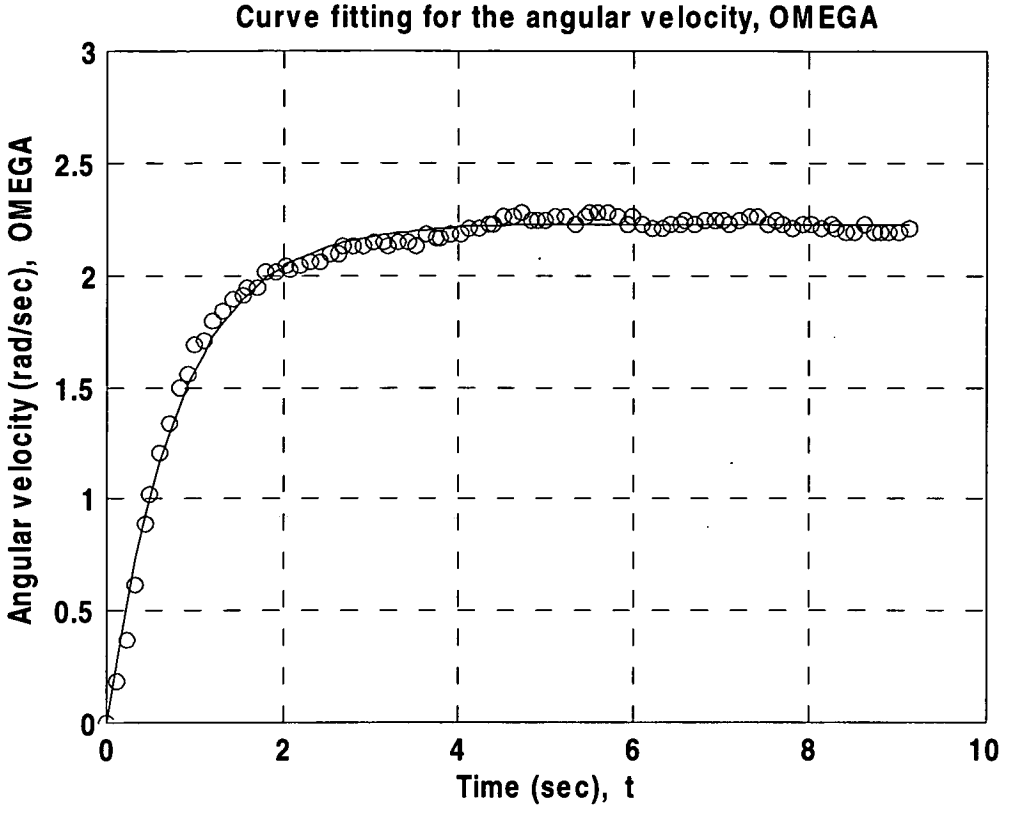


Figure 6.14 The curve fitting result for $I_{eq,2}$ with m = 22 kg.

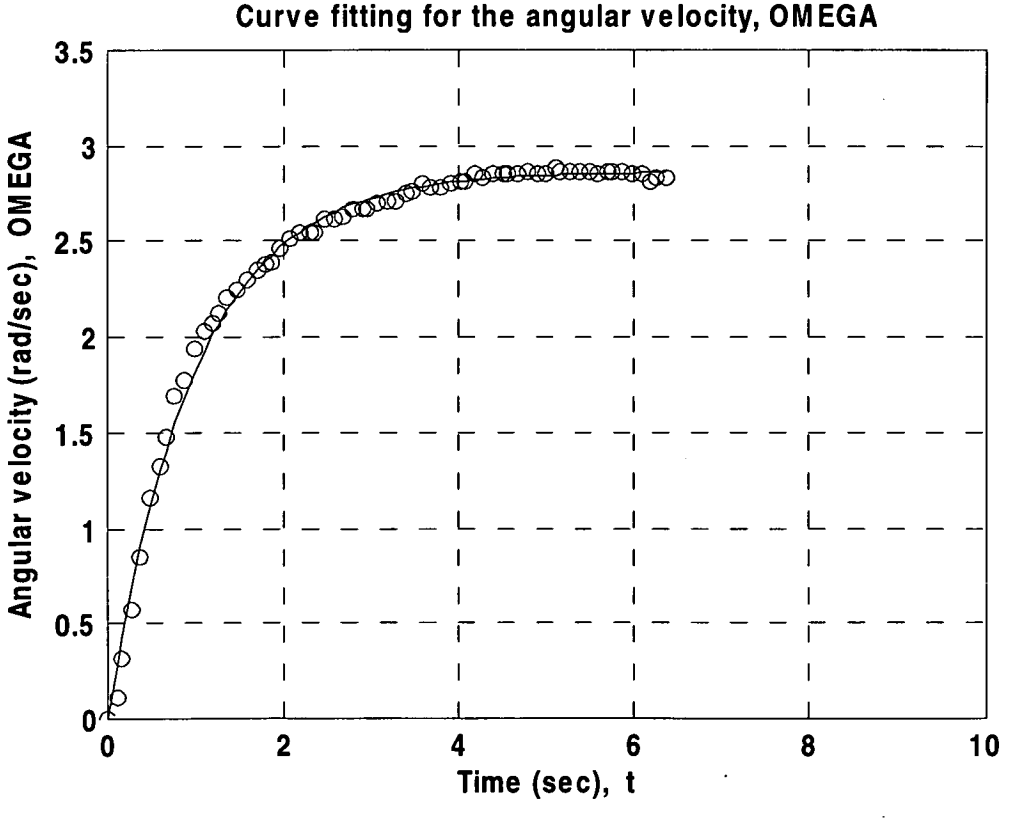


Figure 6.15 The curve fitting result for $I_{eq,2}$ with m = 24 kg.

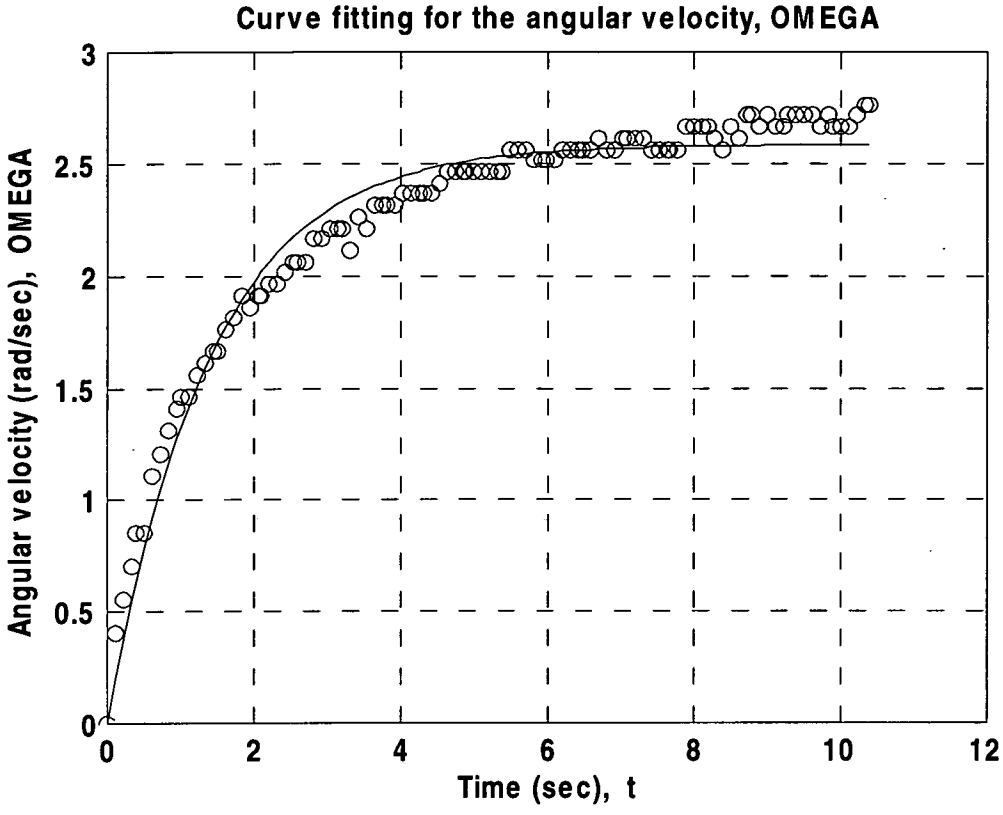


Figure 6.16 The curve fitting result for $I_{eq,4}$ with m = 8 kg.

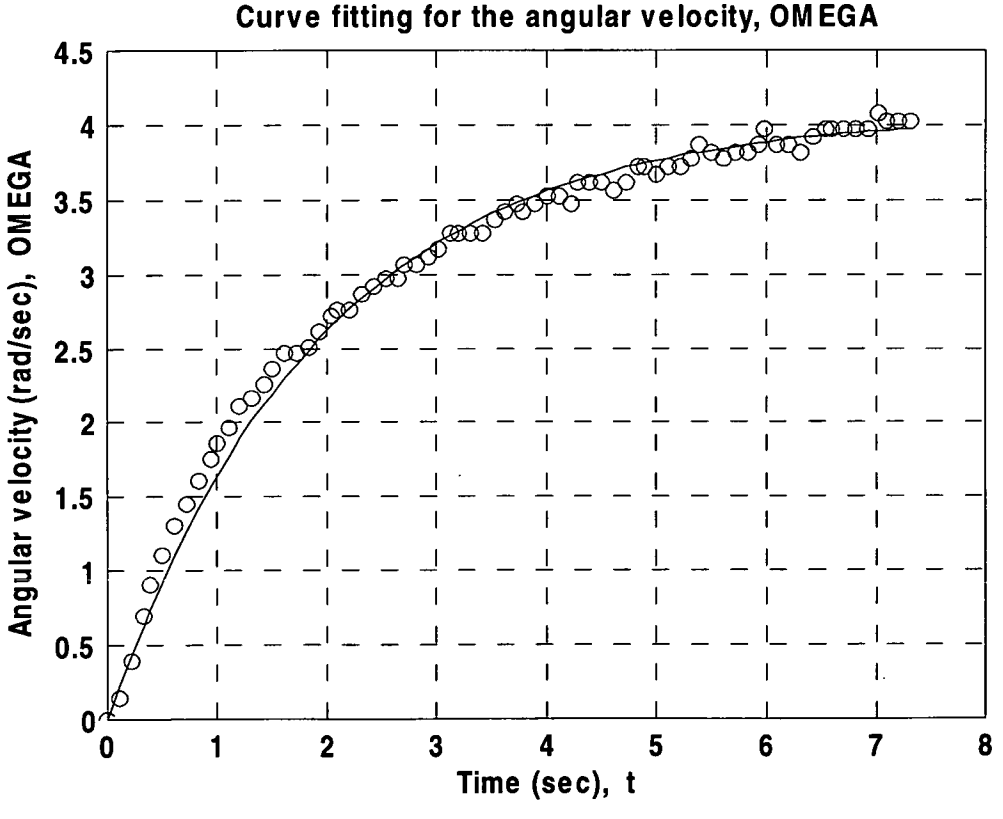


Figure 6.17 The curve fitting result for $I_{eq,4}$ with m = 10 kg.

Chapter 7

The Real-Time Implementation of the Single Cable Mode FLC Controller on the Experimental Rig

7.1 Introduction

In Chapter 3 a simulation using the control strategy feedback linearization control (FLC) has been chosen to handle the highly significant nonlinearities of the 3-D single-cable-mode (SCM) of the RTG crane, which is based on the 1/8 experimental crane rig illustrated in Chapter 4. Five governing equations of motion in x (equation (3.25)), y (equation (3.29)), l (equation (3.33)), ψ (equation (3.38)), α (equation (3.43)), as shown in Figure 7.1, have also been derived correspondingly. As mentioned in section 2.2, one of the most difficult, yet essential, aspects of the experimental system is the practical arrangement required for measuring the swing angle, ψ, and the polar angle, α, of the cable. In order to obtain these two parameters a *visual sensing system* (VSS) (including a CCD camera and a framegrabber) is used to measure directly the location of the lighting sources on the spreader. This was introduced in Chapter 5. By means of the search algorithms discussed in section 5.3, the coordinates of the lighting sources located on an *image plane*, I_C, (where the focal length of the lens on the CCD camera, f, is equal to PC as shown in Figure 2.10) can then be retrieved. This sensing strategy is encoded computationally in a customized S-Function block within the *SIMULINK* diagram constructed for the FLC simulation model. Thus it can be used to control directly the actuators of the system by means of I/O *Device Driver Blocks* in the Crane Application Programming Interface (CAPI), illustrated in Chapter 6.

This chapter is principally concerned with integrating the theories provided in previous chapters of this thesis. The experimental results are then compared to the result from the simulation in Chapter 3. In section 7.2 the modified FLC model, which takes the

data from the visual sensing system and controls the actuators in real-time, is described. The experimental results are presented and discussed in section 7.3.

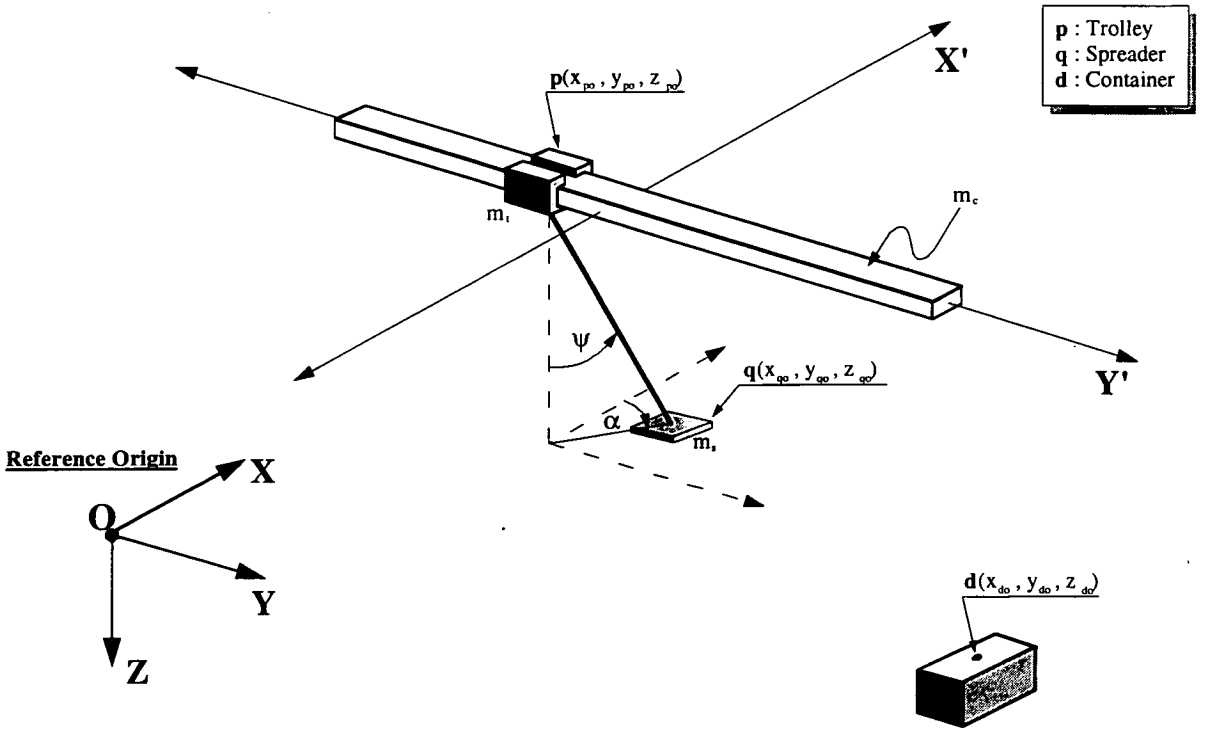


Figure 7.1 (Figure 3.1) The 3-D RTG crane model

7.2 The Real-Time FLC-VSS (Feedback Linearization Control - Visual Sensing System) Model

7.2.1 Acquisition of measurements of the generalised coordinates x, y, *l*, ψ, α

As shown earlier in Chapter 3, there are five coordinates (x, y, *l*, ψ, α) required to describe the single-cable FLC model. The first three coordinates can be obtained directly from shaft encoder and pulse counters. Because the spreader is treated as a lumped mass, only one lighting source located at the *central* point of the spreader is needed. As shown in Figure 7.2, the coordinate of the lighting source, $I_{Q''}$, relative to the coordinate-system ***F***, is:

$$ {}^{F}_{i}I_{Q''} = ({}^{F}x_Q,\ {}^{F}y_Q) \qquad \text{------- (7.1)} $$

(where the pre-subscript, i, denotes that the contents, ${}^{F}x_Q$ and ${}^{F}y_Q$, which are the row_index and column_index output from the visual sensing system respectively, are integers.)

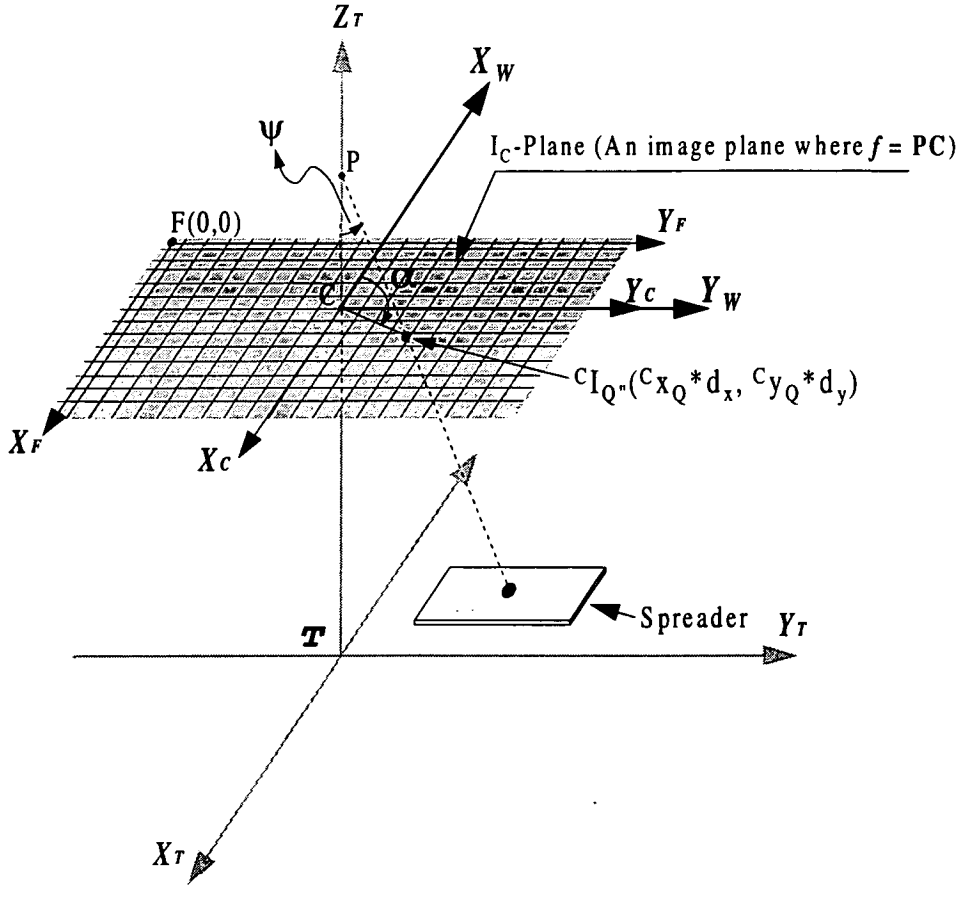


Figure 7.2 The relationship between the single lighting source on the spreader and the coordinate systems

Following the process described in section 2.2.3, the coordinates in equation (7.1) have to be transformed relative to the coordinate-system ***C*** first, and then, by multiplying them by the scaling factors, d_x and d_y, the *physical dimensions* of the coordinates can be obtained; that is:

$$ {}^{F}_{i}I_{Q''} = ({}^{F}x_Q,\ {}^{F}y_Q) \longrightarrow {}^{C}_{i}I_{Q''} = ({}^{C}x_Q,\ {}^{C}y_Q) \longrightarrow {}^{C}I_{Q''} = ({}^{C}x_Q * d_x,\ {}^{C}y_Q * d_y) $$

where

$$^{C}x_{Q} = {}^{F}x_{Q} - {}^{F}x_{CORG} \qquad \text{------- (7.2)}$$

$$^{C}y_{Q} = {}^{F}y_{Q} - {}^{F}y_{CORG} \qquad \text{------- (7.3)}$$

($^{F}x_{CORG}$, $^{F}y_{CORG}$) is the origin C of the coordinate-system $\boldsymbol{C}$ relative to the coordinate-system $\boldsymbol{F}$, as shown in equation (2.32). The scaling factors, d_x and d_y, are measured by means of the method of section 2.2.2.iv. The results are detailed in Table 7.1 and 7.2. given at the end of this section.

As shown in Figure 7.2, the swing angle, ψ, can then be found by using the geometrical relationships of $\Delta PCI_{Q''}$, thus,

$$\psi = \tan^{-1}\left(\frac{\sqrt{(^{C}x_{Q} * d_x)^2 + (^{C}y_{Q} * d_y)^2}}{PC}\right) = \tan^{-1}\left(\frac{\sqrt{(^{C}x_{Q} * d_x)^2 + (^{C}y_{Q} * d_y)^2}}{f}\right)$$

------- (7.4)

Also the polar angle, α, can be measured from the $\boldsymbol{X_W}$ axis in the world coordinate-system, recalled from Figure 2.18 and Figure 7.1. Therefore,

$$\alpha = \tan^{-1}\left(\frac{^{C}y_{Q} * d_y}{-(^{C}x_{Q} * d_x)}\right) \qquad \text{------- (7.5)}$$

7.2.2 The Layout of the FLC-VSS Model

Figure 7.3, at the end of this chapter, shows the complete FLC-VSS model. In fact, it resembles Figure 3.2 in Chapter 3 in some aspects. The desired (or final) coordinates of the spreader, $\underset{\sim}{q_d}$, can be set up in the block, **qd**. The device driver block, *Counter Input*, is responsible for reading the number of counts from the encoders on each

actuator and, after multiplying by a scaling factor (metres/count), measurements of the generalised coordinates x, y and l can be obtained. The *VSS* block outputs the coordinates ((row_index, column_index)) of the lighting source relative to the coordinate-system $\boldsymbol{F}$. The equations, (7.1) ~ (7.5) in the last section, are then applied to them to retrieve ψ and α. These five coordinates are multiplexed together as the vector, $\mathbf{q}$ ($\underset{\sim}{q}$), and passed to the FLC controller to calculate the control torques, T.

With the conversion of torque-to-voltage (in section 6.3), the computed voltages are obtained and input to the DAC block (8 channel DAC) to activate the motors. It should be noted that, as in almost all other systems, actuators can *saturate* because the dynamic range is usually limited. This also applies to the FLC-VSS model, so an artificial saturation effect (which can be pre-specified and set up as required) is added. Moreover, because there is no braking device as in a real RTG crane, the maximum linear velocity in each actuator is constrained to 70 percent of its full capacity to help to prevent the trolley from over-running the physically available spans. It should be noted that the current configuration of the experimental rig can only allow the trolley to move in one direction on the same axis. For example, if the target x- coordinate is larger than the initial one the controller will certainly cause the trolley move towards to the x(+) direction from the beginning. However once this direction is decided the trolley is no longer allowed to move backwards (the x- direction). This is because the counting direction of the counters used in the rig can not be altered during the operation. Once it is configured to count up (or down) there is no way change it unless the configuration procedure is re-started. This implies that, even if the trolley is allowed to move in either direction, the direction of counting will not change to respond and result is unpredictable. Also, as shown in Figures 4.4 ~4.11, there is a dead-zone for each actuator. For example, if the input voltage is smaller than 0.6 volts for the x- actuator there will not be any movement. Moreover the counters in such situations will mistake the very small ripples from the encoders as a pulse signal and start counting. Therefore a small voltage which pushes the trolley away from zero velocity is imposed. Once the control sequence commences the trolley will not move until it reaches the desired coordinates.

L_D (mm)	f (mm)	D (mm)	M_1 M_2 (row_index, column_index)	$(n-1)$	d_x ($\times 10^{-3}$ mm)
400	8.5	100	(144, 372) (397, 372)	252	8.433
500	8.5	100	(170, 372) (372, 372)	201	8.458
600	8.5	100	(186, 372) (356, 372)	169	8.383
700	8.5	100	(197, 372) (343, 372)	145	8.373
800	8.5	100	(206, 372) (335, 372)	128	8.301
900	8.5	100	(214, 372) (326, 373)	111	8.509
1,000	8.5	100	(219, 372) (322, 372)	102	8.333
1,100	8.5	100	(224, 371) (317, 372)	92	8.399
1,200	8.5	100	(228, 374) (310, 372)	82	8.638

Table 7.1 The scaling factor, d_x (↕)

The average value of the scaling factor, d_x, (↕) in equation (2.30) is 8.425×10^{-3} mm

L_D (mm)	f (mm)	D (mm)	M_1 M_2 (row_index, column_index)	$(n-1)$	d_y ($\times 10^{-3}$ mm)
400	8.5	100	(271, 248) (270, 497)	248	8.569
500	8.5	100	(270, 276) (270, 476)	199	8.543
600	8.5	100	(271, 292) (271, 460)	167	8.483
700	8.5	100	(270, 305) (271, 447)	141	8.612
800	8.5	100	(271, 313) (272, 439)	127	8.366
900	8.5	100	(272, 316) (271, 428)	111	8.509
1,000	8.5	100	(270, 324) (272, 427)	102	8.333
1,100	8.5	100	(272, 325) (272, 420)	94	8.221
1,200	8.5	100	(228, 328) (310, 410)	81	8.745

Table 7.2 The scaling factor, d_y ($\leftrightarrow$)

The average value of the scaling factor, d_y, ($\leftrightarrow$) in equation (2.30) is 8.486×10^{-3} mm

7.3 Using the FLC-VSS on the experimental rig

[a] The trolley moving only in the x-direction

In order to compare the actual results with the simulations of section 3.5.[a] the system is actuated with the same initial conditions, as follows:

$\{h\}=[1.8122\ 0.0\ 0.0\ 0\ 0]^T$, $\{g\}=[0.821\ 0.0\ 0.0\ 0\ 0]^T$

$\mathbf{q_i}=\mathbf{[0.0\ 0.0\ 1.0\ 0.0\ 0.0]^T}$, $\mathbf{q_d}=\mathbf{[1.5\ 0.0\ 1.0\ 0.0\ 0.0]^T}$

($[...]^T$ denotes the transpose of the relevant vector, and $\mathbf{q_i}$ is the initial position of the spreader.)

The detailed experimental results are shown in Figures 7.4 ~ 7.7 at the end of this chapter, in which the tests show:

t_s (stop time, with $\dot{x}=0$) = 10.2 seconds

ψ ($t=t_s$) $\approx 2.8°$

The major difference between this experiment and the simulation in section 3.5.[a] is that the swing angle, ψ, is immediately damped out when the trolley reaches the desired position (x_d = 1500 mm) because the configuration of the experimental rig only allows the trolley to either accelerate or decelerate in one direction. Therefore there is a residual swing angle, of around 2.8°, which can be damped more readily if the passive damping is increased, perhaps in line with that found in the full-size RTG machines.

By comparing Figure 7.6 with Figure 3.6 it can be seen that the swing frequency is no longer equal to the natural swing frequency of 0.4985 Hz (based on a $\frac{1}{2\pi}\sqrt{\frac{g}{l}}$ approximation with the fixed cable length (l) of 1 metre). The reason for this is because the polar angle, α, as shown in Figure 7.7, is not equal to 0, as in the simulation of section 3.5.[a]. Slight perturbations, caused by local aerodynamics, or the

misalignment of the sliding guides of the trolley, result in some circular motion of the spreader.

In Figure 7.5 it is also interesting to see clearly evident 'accelerate-decelerate' behaviour of the trolley taking place and this can be regarded as being the controller's attempt to compensate for the swing angle oscillations.

[b] The trolley moving only in the y- direction

In this section the trolley is constrained to move only in the y-direction. The system is actuated with initial conditions as follows:

$\{h\}=[0.0\ 1.359\ 0.0\ 0\ 0]^T$, $\{g\}=[0.0\ 0.4617\ 0.0\ 0\ 0]^T$

$\mathbf{q_i=[0.0\ -0.5\ 1.0\ 0.0\ 0.0]^T}$, $\mathbf{q_d=[0.0\ 0.5\ 1.0\ 0.0\ 0.0]^T}$

The experimental result is :

t_s (stop time, with $\dot{y}=0$) = 12.1 sec

$\psi(t=t_s) \approx 1.3°$

Figure 7.10 shows a dc offset of 1.3 degrees. It has a frequency of approximately 1 Hz, which is similar to the result of Figure 3.12 in section 3.5.[c]. The phenomenon of conical displacements, via α, is also present. In Figure 7.9 it can be seen that the 'accelerate-decelerate' behaviour of the trolley movement (between t = 7.0 sec ~ t_s) is also present here. The details are shown in Figure 7.8 ~ Figure 7.11.

[c] The trolley moving in the x- and y- direction simultaneously

At this stage, actuations in the x- and y-direction are both invoked simultaneously. The initial conditions are as follows:

$\{h\}=[1.8122\ 1.359\ 0.0\ 0\ 0]^T$, $\{g\}=[1.631\ 0.4617\ 0.0\ 0\ 0]^T$

$\mathbf{q_i=[0.0\ -0.5\ 1.0\ 0.0\ 0.0]^T}$, $\mathbf{q_d=[1.5\ 0.5\ 1.0\ 0.0\ 0.0]^T}$

This results in :

t_{xs} (stop time, with $\dot{x}=0$) = 10.2 seconds

t_{ys} (stop time, with $\dot{y}=0$) = 12.3 sec

ψ ($t=t_{ys}$) $\approx 3.9°$

The details are shown in Figure 7.12 ~ Figure 7.17.

The residual swing angle increases to 3.9°, compared with 3.6° in the simulation shown in Figure 3.21 of section 3.5.[d]. As stated earlier in section [a] the trolley can not be allowed to move 'backward' during the manoeuvre as in the simulation (Figure 3.18 and Figure 3.20). Moreover the trolley cannot have zero velocity unless it reaches the target coordinate. The reason for this is because, as shown in Figures 4.4 ~ 4.11, the actuators must have a minimum voltage to move and in the final part of movement the correspondent voltages for the required velocity will be below this limit, with the result that the trolley stops prematurely.

[d] All three actuators operating simultaneously

Finally all three actuators are operated simultaneously. The initial conditions are,

$\{h\}=[1.8122\ 1.359\ 2.3918\ 0\ 0]^T$, $\{g\}=[1.631\ 0.4617\ 1.4302\ 0\ 0]^T$

$\mathbf{q_i=[0.0\ -0.5\ 0.7\ 0.0\ 0.0]^T}$, $\mathbf{q_d=[1.5\ 0.5\ 1.2\ 0.0\ 0.0]^T}$

This results in :

t_{xs} (stop time, with $\dot{x} = 0$) = 10.2 seconds

t_{ys} (stop time, with $\dot{y} = 0$) = 12.3 seconds

t_{ls} (stop time, with $\dot{l} = 0$) > 30 seconds

ψ **(t=30)** $\approx 3.0°$

The details are shown in Figure 7.18 ~ Figure 7.28.

Although actuation in the x- and the y- axis show similar results to the previous sections, the actuation in the *l*- direction fails to finish in 30 seconds. From Figure 7.28 it can be seen that the amount of required torque in the *l*- direction is very small, because in considering the original FLC controller in Chapter 3 the friction term was not taken into account. However, as shown in Table 6.1, there exists a level of static friction, 2.714 Nm, in the *l*- direction, which is larger than the required dynamic torque. Therefore the spreader only moves at the minimum velocity set up for overcoming the dead-zone in Figure 4.10 and Figure 4.11, as mentioned in the last section. Therefore because the required amount of torque is far larger than the static frictions in the x- and y- directions (as shown in Figure 7.26 and Figure 7.27) the friction aspect will not have too much effect.

7.4 Conclusions

In this chapter the analyses of the previous chapters are applied in order to establish the final real-time implementation of the FLC-VSS model. Because the dynamics of FLC controller developed in Chapter 3 treated the spreader as a single-cable point mass, where only the central position of the spreader is of interest, one lighting source, located at the centre of the spreader, is sufficient to obtain the relevant information. In section 7.2.1 a method is derived to retrieve the swing angle, ψ, and the polar angle, α, which can then be combined with other data such as x, y and l to input into the model.

The experimental results are presented in section 7.3. The planar motion in section 7.3.[a], and [b], in which the spreader travels with a fixed cable length, shows a reasonable performance (ψ $(t=t_s) \approx 2.8°$ in 7.3.[a], $\psi(t=t_s) \approx 1.3°$ in 7.3.[b]) compared with the simulation results (ψ $(t=t_s) \approx 7°$ in 3.5.[a], $\psi(t=t_s) \approx 2.7°$ in 3.5.[b]) in section 3.5. Also in section 7.3.[c] the FLC controller shows some ability to reduce the swing angle of the spreader. However, because the unmodelled friction terms have not been taken account in the original FLC simulation model, the FLC-VSS model does not cope well with the condition of variable cable lengths, as shown in 7.3.[d]. However the work presented in this chapter does prove that the CAPI (Crane Application Programming Interface) can be fully integrated with the experimental rig to actually control the movement of the trolley by sensing a lighting source on the swinging spreader. The whole system presents an entirely unique platform which can be used with different control strategies in the future.

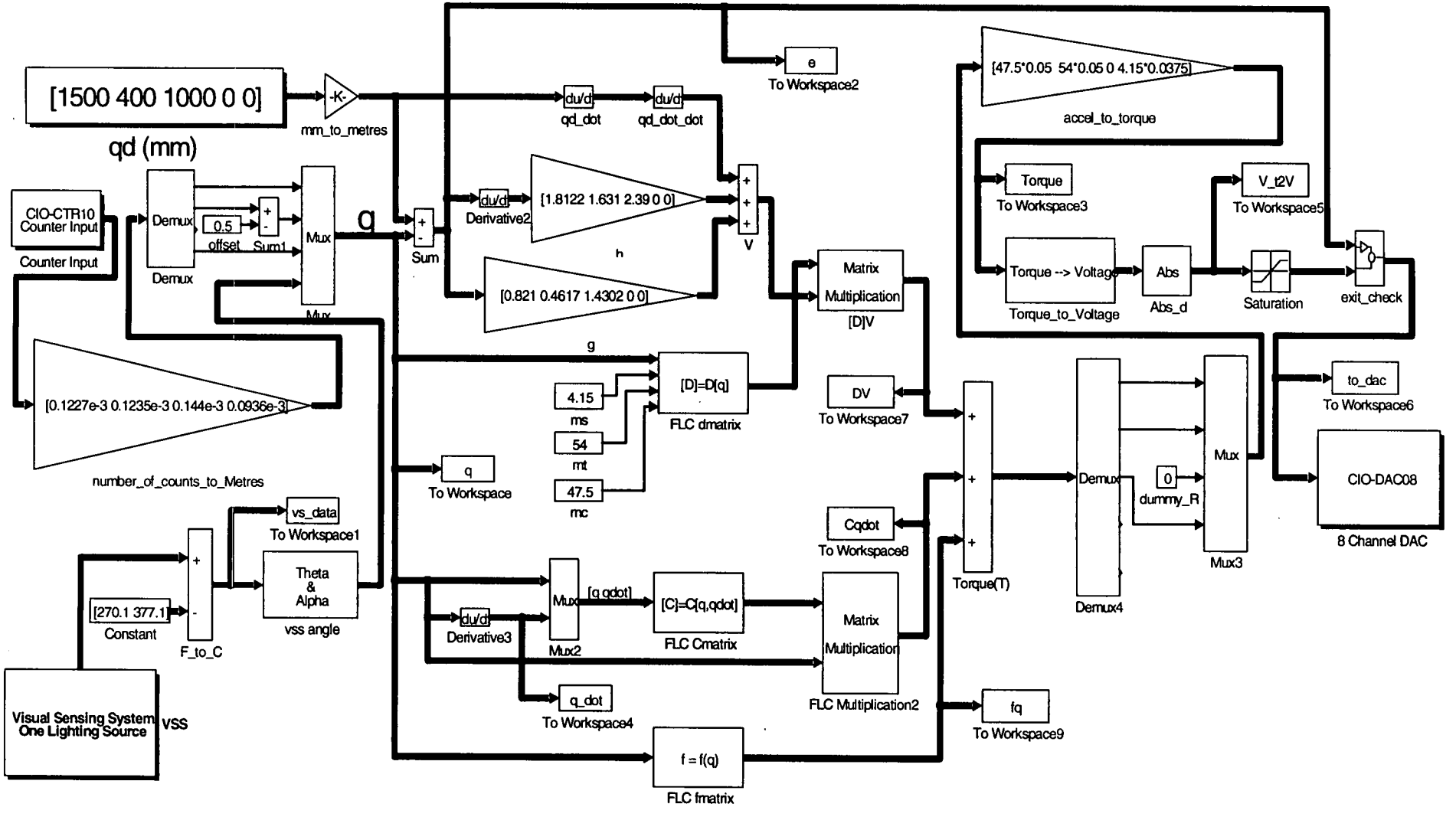


Figure 7.3 FLC-VSS model

Simulation results for section [a] in section 7.3

Setup:

$\{h\}=[1.8122\ 0.0\ 0.0\ 0\ 0]^T$, $\{g\}=[0.821\ 0.0\ 0.0\ 0\ 0]^T$

$\mathbf{q_i=[0.0\ 0.0\ 1.0\ 0.0\ 0.0]^T}$, $\mathbf{q_d=[1.5\ 0.0\ 1.0\ 0.0\ 0.0]^T}$

Test results:

t_s (stop time, with $\dot{x} = 0$) = 10.2 seconds

ψ **($t=t_s$) ≈ 2.8°**

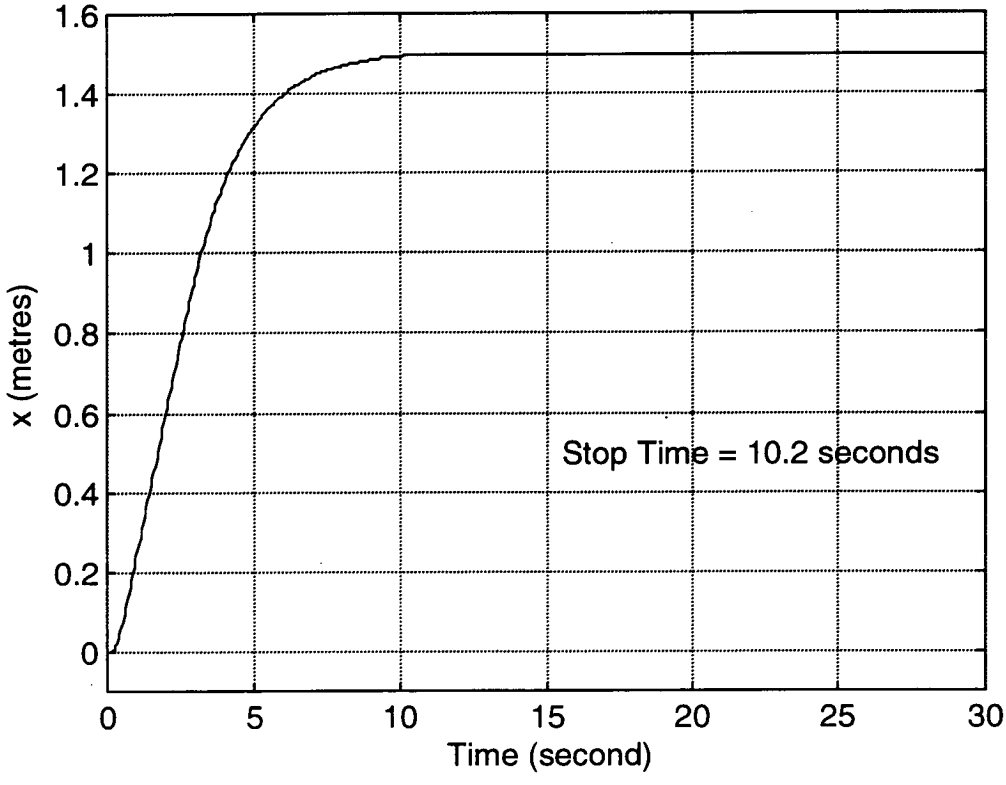


Figure 7.4 Position of the trolley in the x- direction

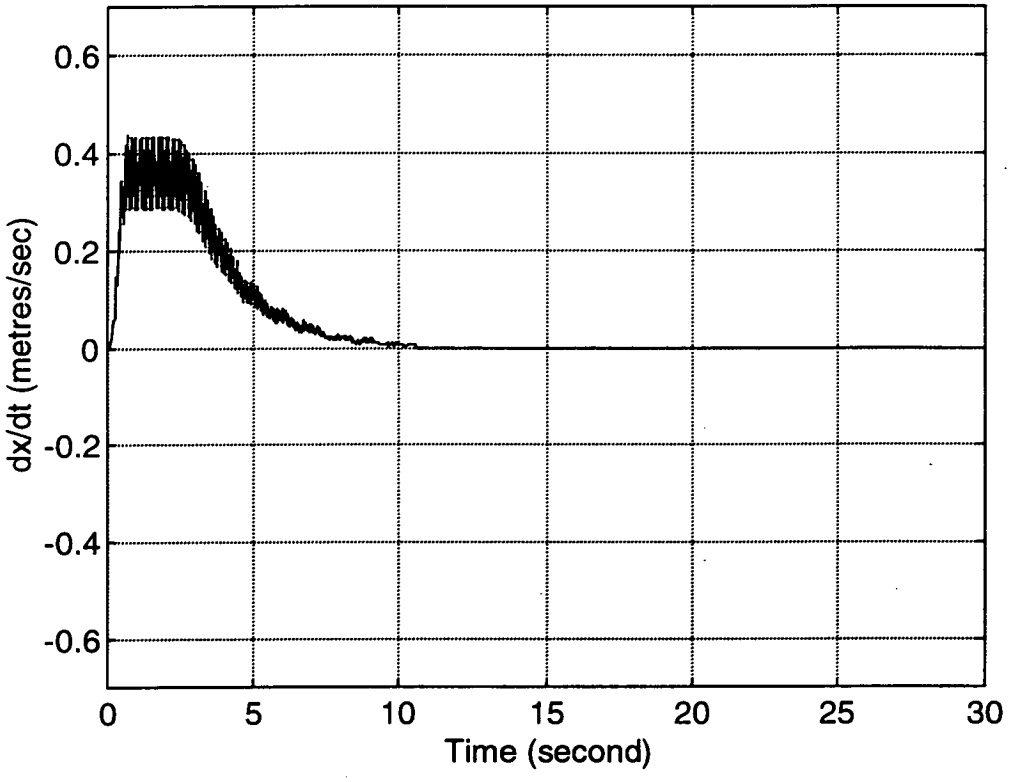


Figure 7.5 Velocity of the trolley in the x- direction

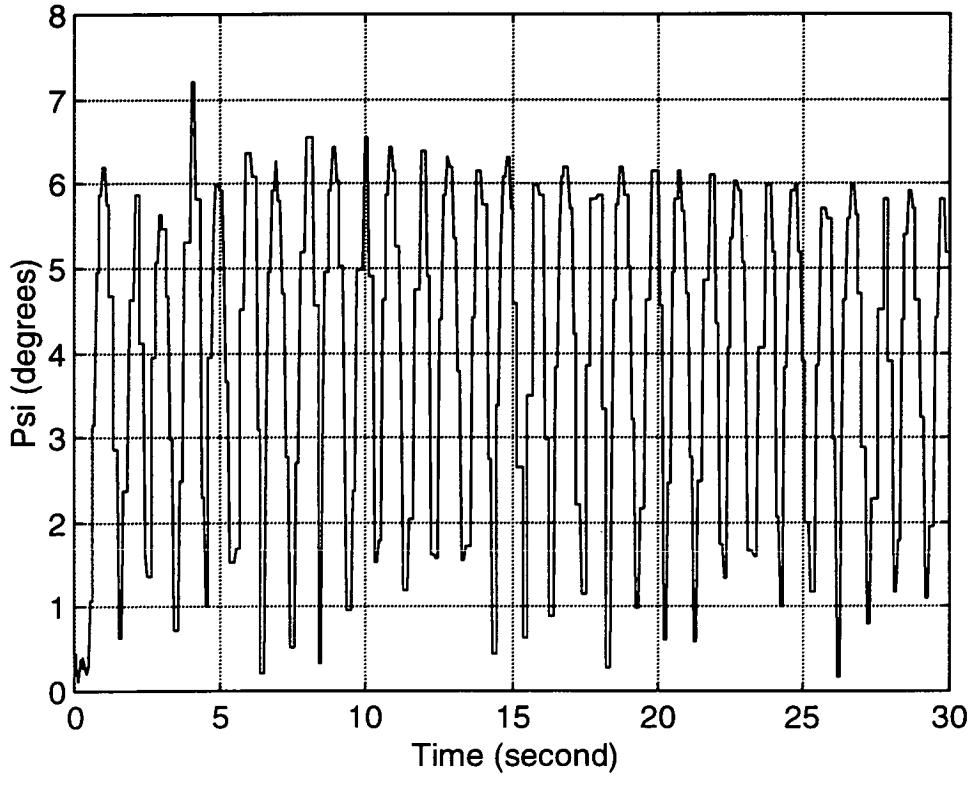


Figure 7.6 The swing angle ψ

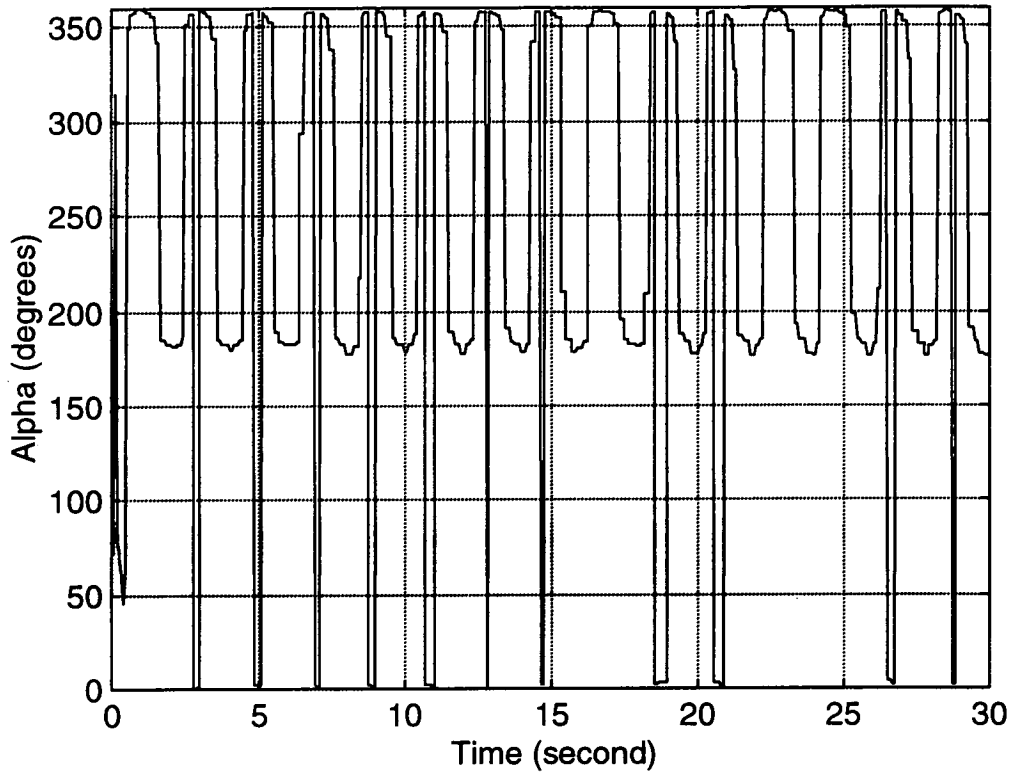


Figure 7.7 The polar angle α

Simulation results for section [b] in section 7.3

Setup:

$\{h\}=[0.0\ 1.359\ 0.0\ 0\ 0]^T$, $\{g\}=[0.0\ 0.4617\ 0.0\ 0\ 0]^T$

$\mathbf{q_i=[0.0\ -0.5\ 1.0\ 0.0\ 0.0]^T}$, $\mathbf{q_d=[0.0\ 0.5\ 1.0\ 0.0\ 0.0]^T}$

Test results:

t_s (stop time, with $\dot{y}=0$) = 12.1 sec

$\mathbf{\psi(t=t_s) \approx 1.3°}$

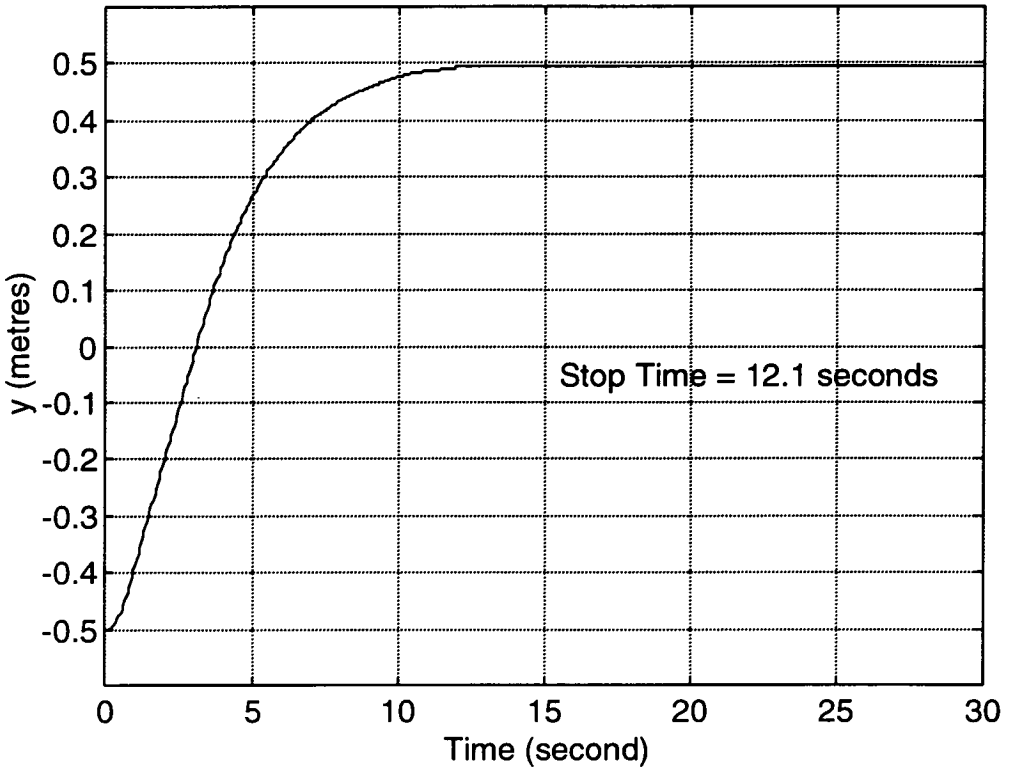


Figure 7.8 Position of the trolley in the y- direction

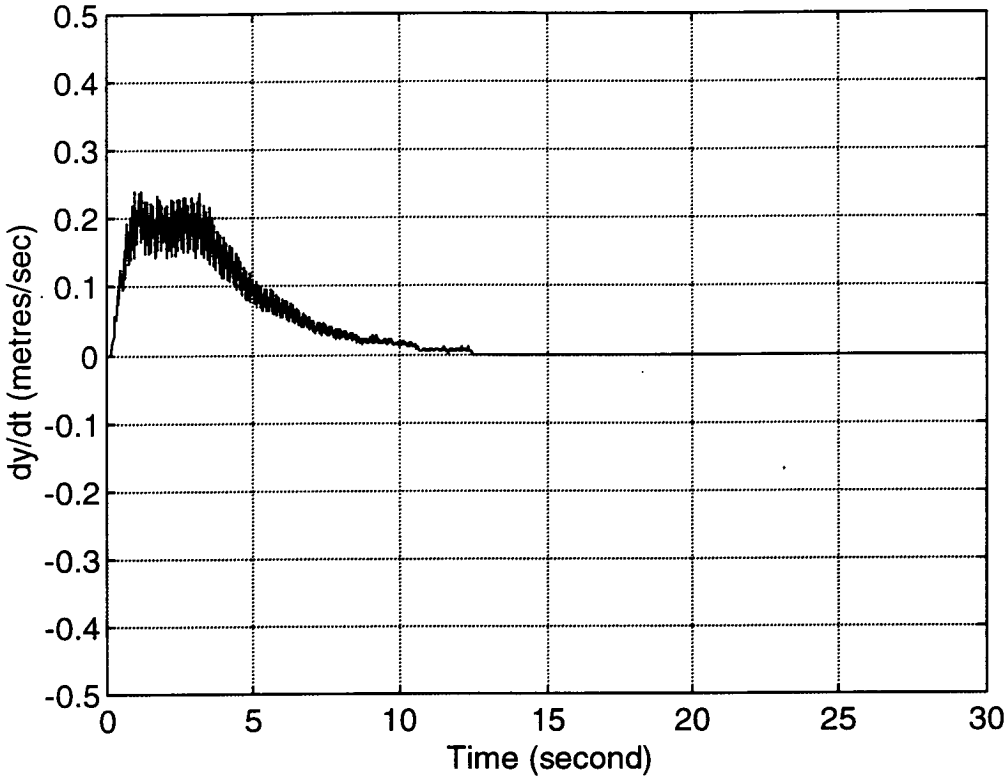


Figure 7.9 Velocity of the trolley in the y- direction

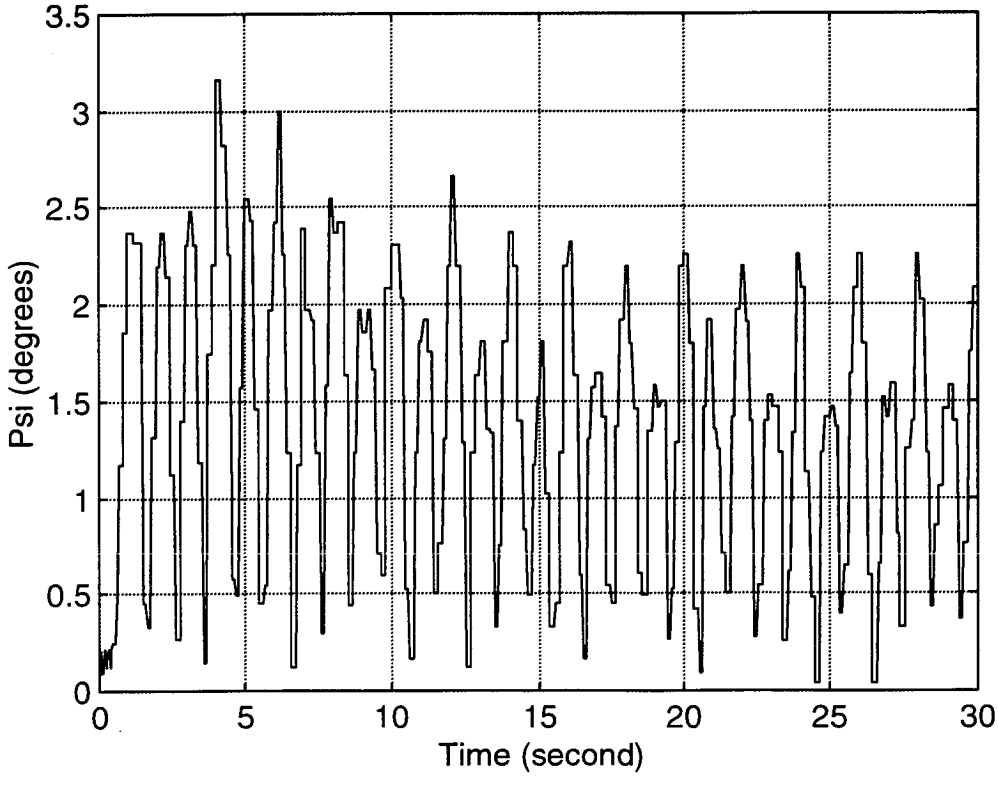


Figure 7.10 The swing angle ψ

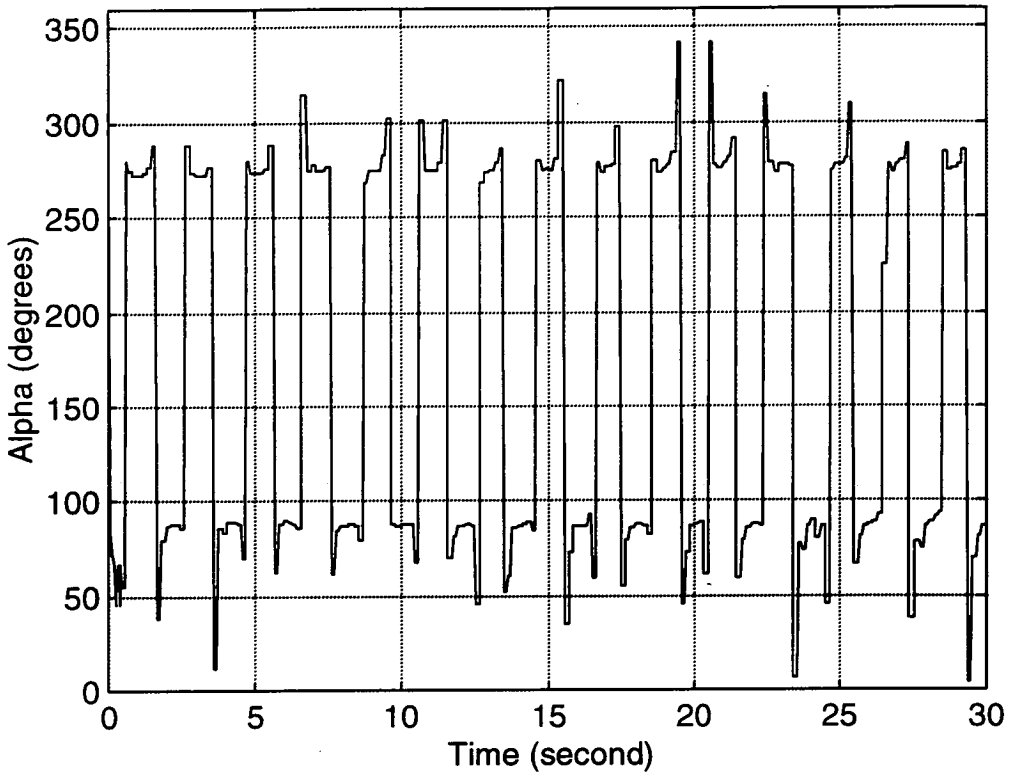


Figure 7.11 The polar angle α

Simulation results for section [c] in section 7.3

Setup:

$\{h\}=[1.8122\ 1.359\ 0.0\ 0\ 0]^T$, $\{g\}=[1.631\ 0.4617\ 0.0\ 0\ 0]^T$

$\mathbf{q_i=[0.0\ -0.5\ 1.0\ 0.0\ 0.0]^T}$, $\mathbf{q_d=[1.5\ 0.5\ 1.0\ 0.0\ 0.0]^T}$

Test results:

t_{xs} (stop time, with $\dot{x}=0$) = 10.2 seconds

t_{ys} (stop time, with $\dot{y}=0$) = 12.3 sec

ψ **(t=t_{ys})** $\approx 4.0°$

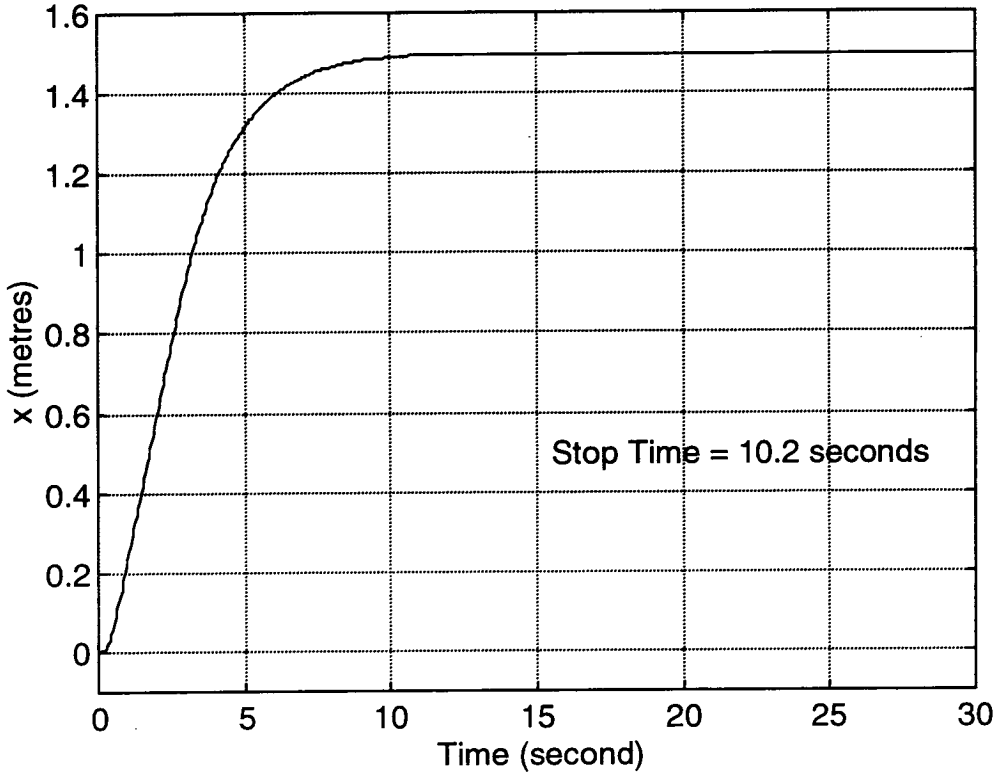


Figure 7.12 Position of the trolley in the x- direction

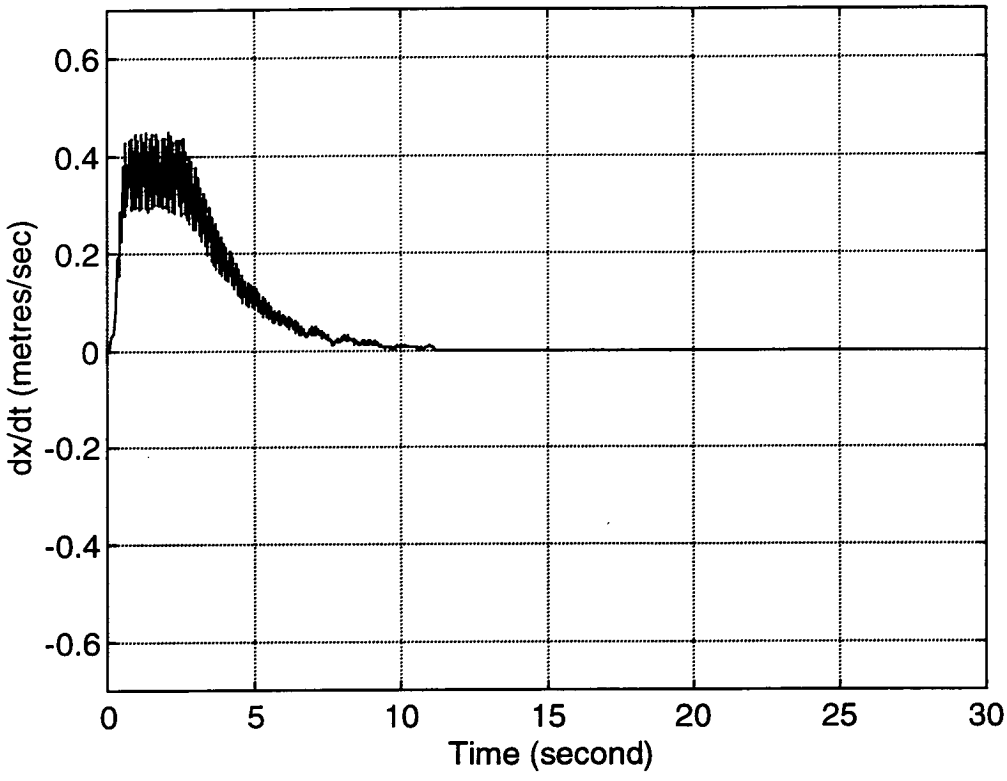


Figure 7.13 Velocity of the trolley in the x- direction

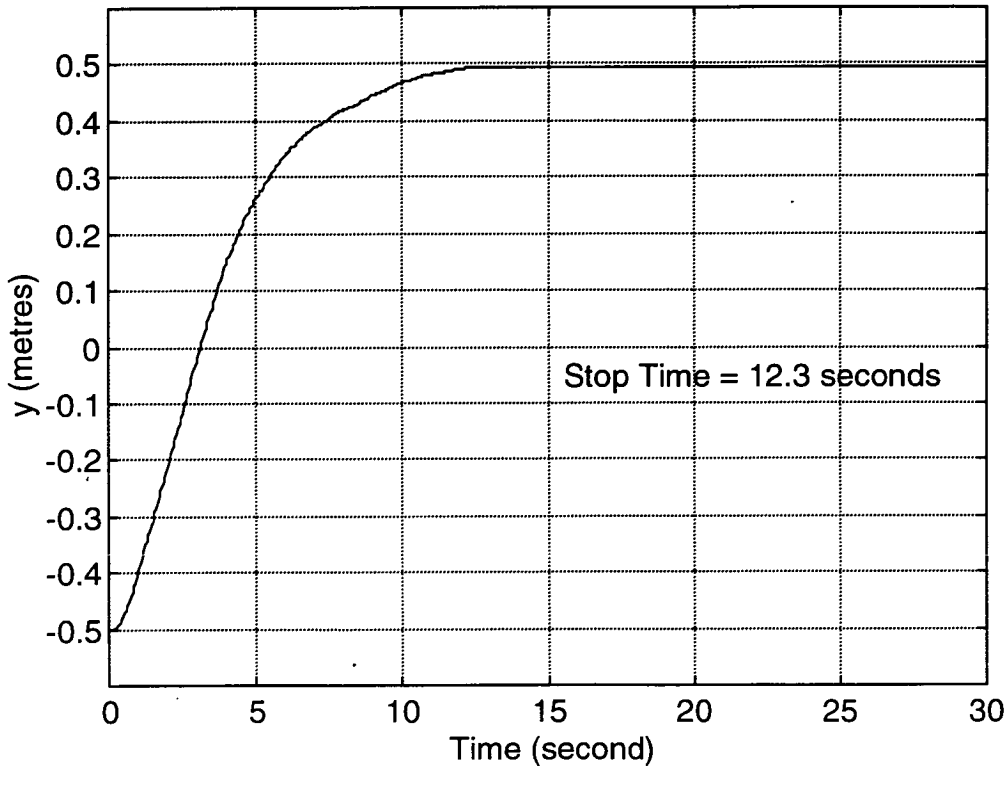


Figure 7.14 Position of the trolley in the y- direction

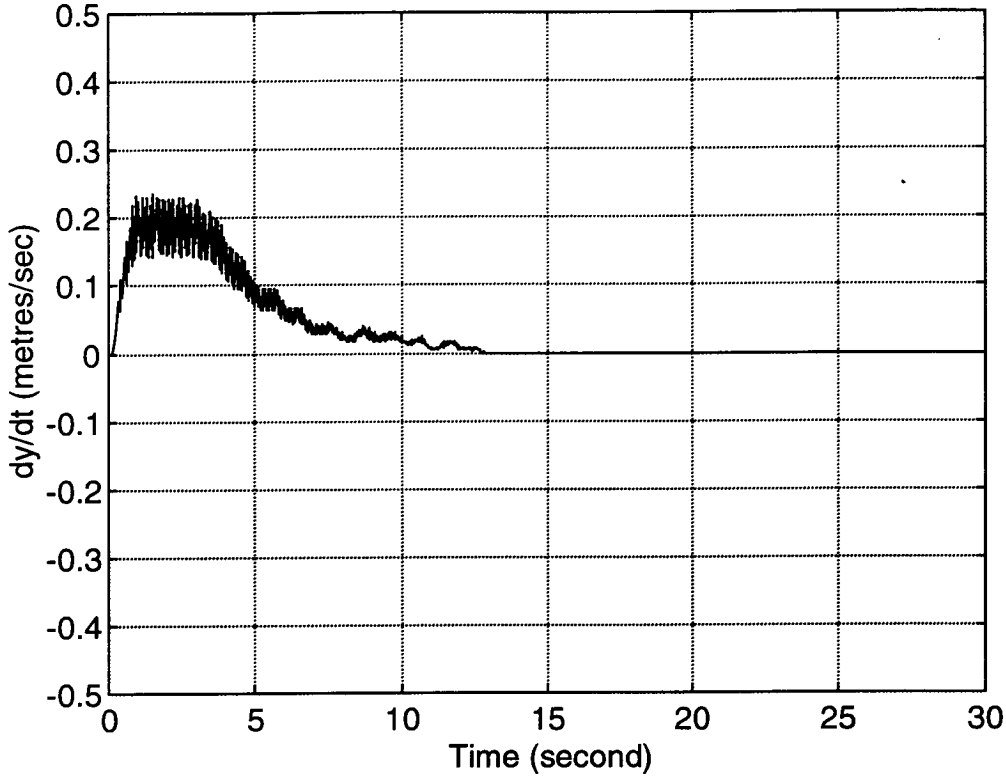


Figure 7.15 Velocity of the trolley in the y- direction

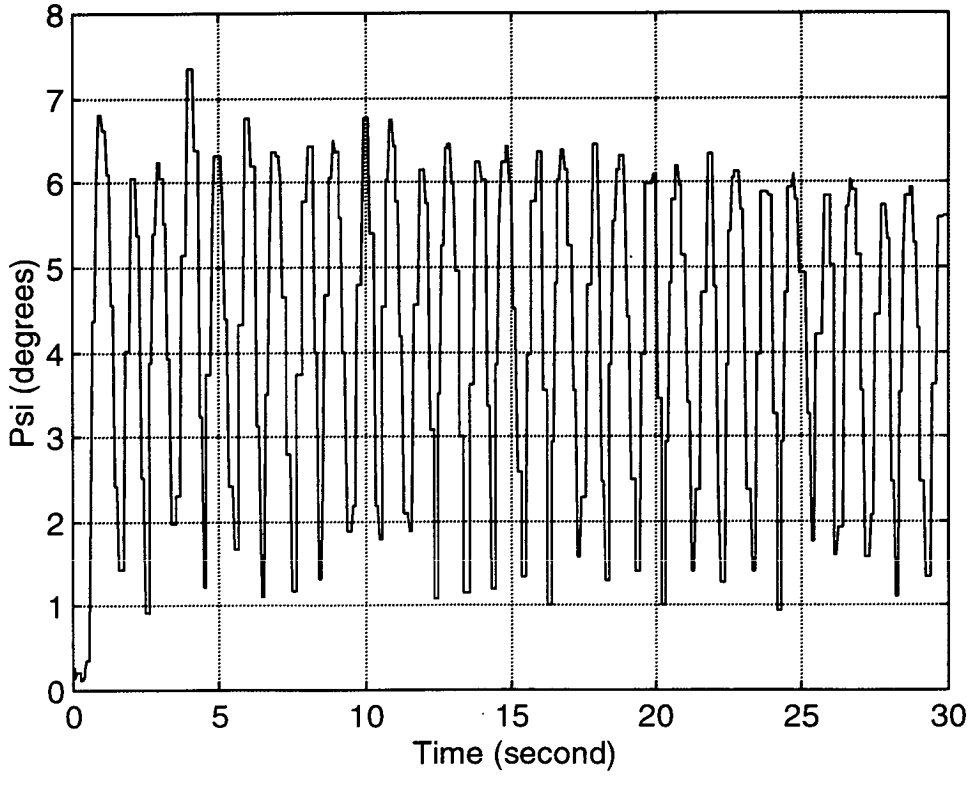


Figure 7.16 The swinging angle ψ

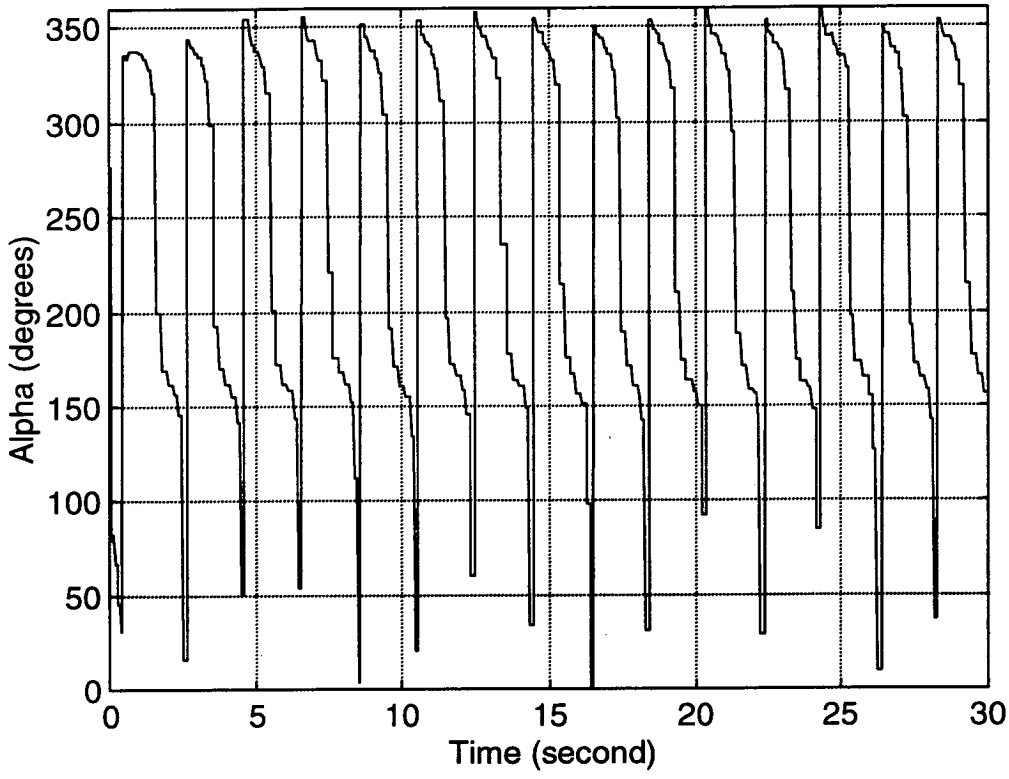


Figure 7.17 The polar angle α

Simulation results for section [d] in section 7.3

Setup:

$\{h\}=[1.8122\ 1.359\ 2.3918\ 0\ 0]^T$, $\{g\}=[1.631\ 0.4617\ 1.4302\ 0\ 0]^T$

$\mathbf{q_i=[0.0\ -0.5\ 0.7\ 0.0\ 0.0]^T}$, $\mathbf{q_d=[1.5\ 0.5\ 1.2\ 0.0\ 0.0]^T}$

Test results:

t_{xs} (stop time, with $\dot{x}=0$) = 10.2 seconds

t_{ys} (stop time, with $\dot{y}=0$) = 12.3 seconds

t_{ls} (stop time, with $\dot{l}=0$) > 30 seconds

ψ **(t=30)** $\approx 3.0°$

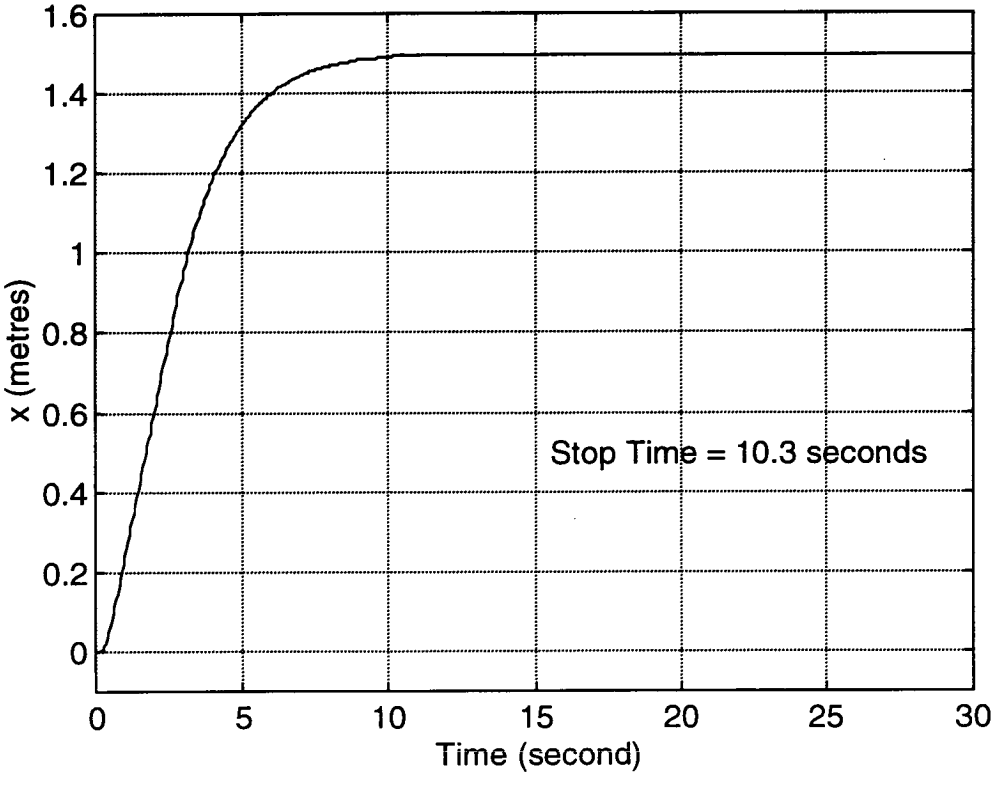


Figure 7.18 Position of the trolley in the x- direction

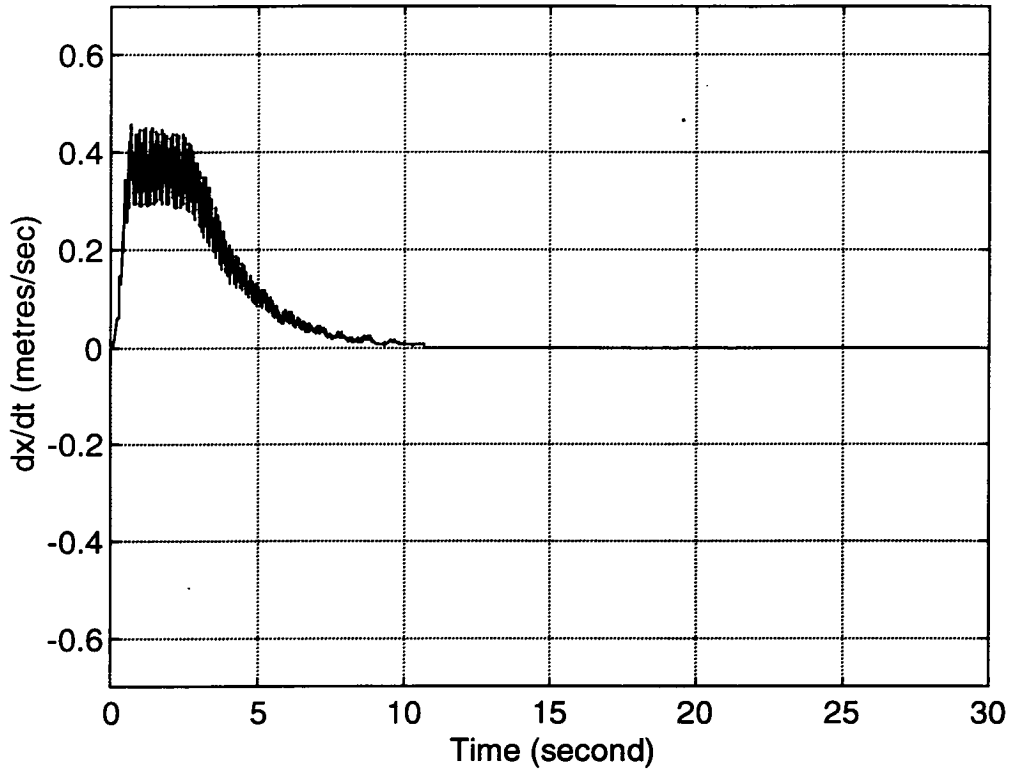


Figure 7.19 Velocity of the trolley in the x- direction

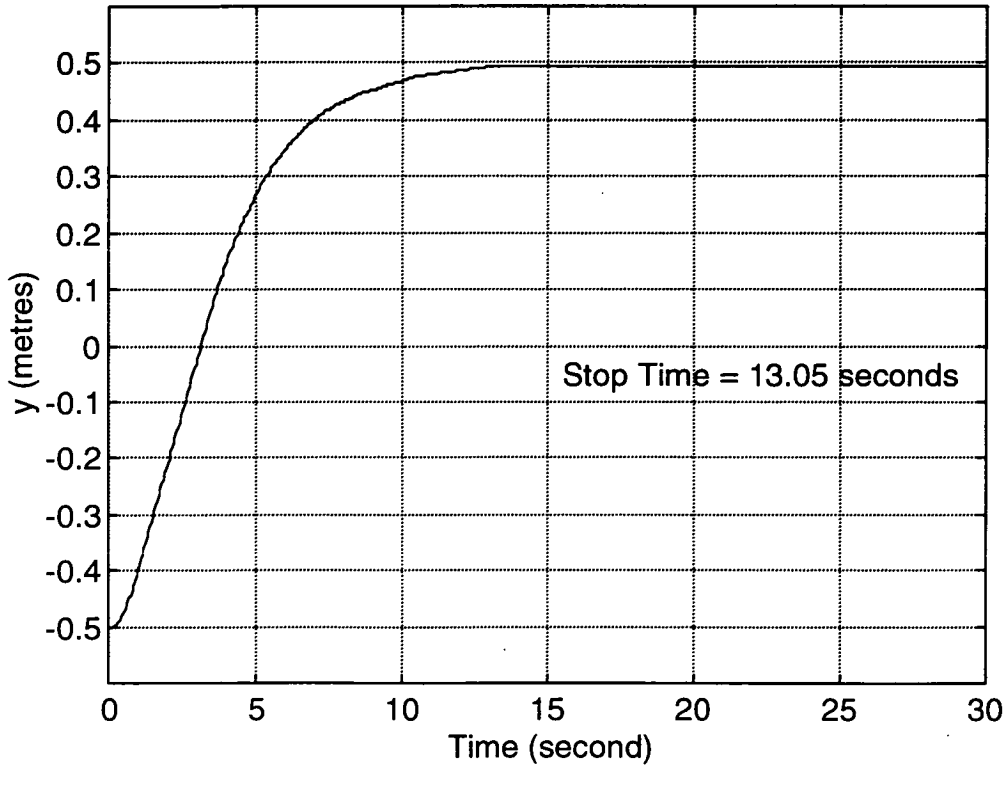


Figure 7.20 Position of the trolley in the y- direction

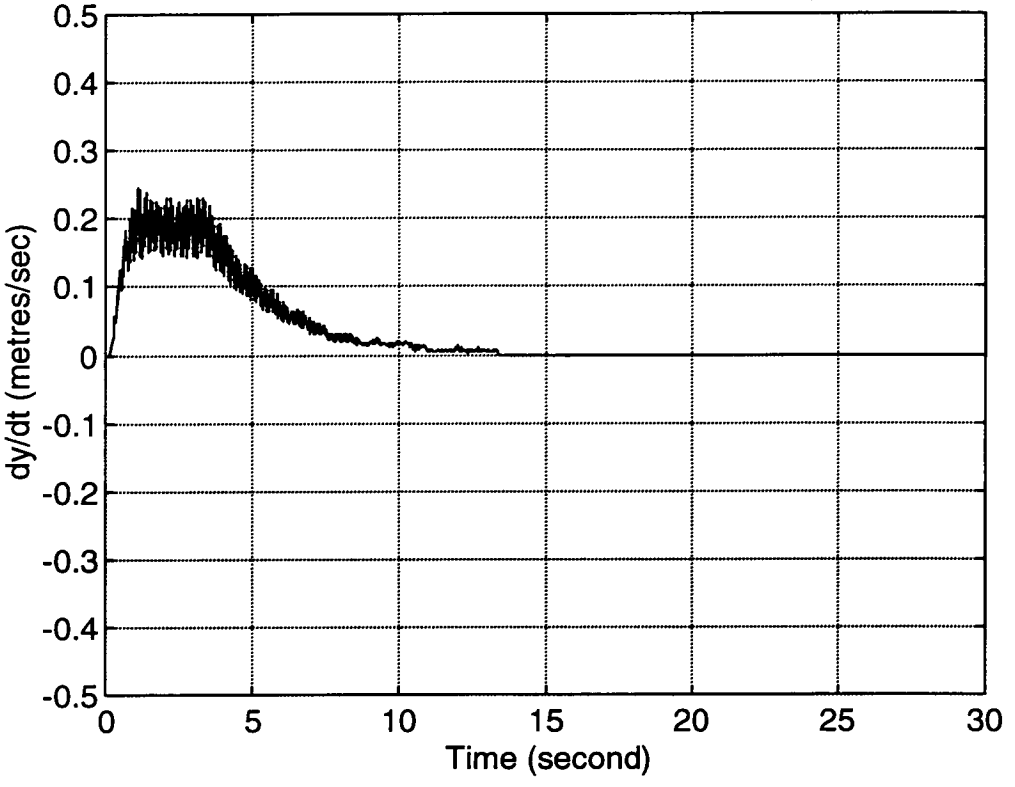


Figure 7.21 Velocity of the trolley in the y- direction

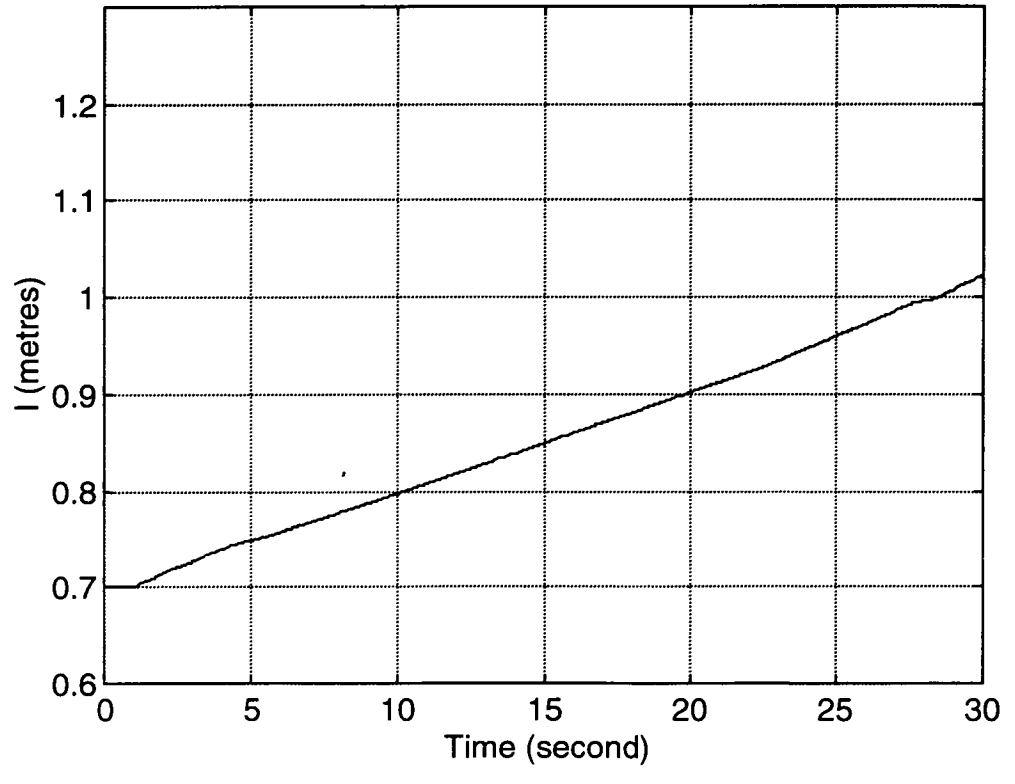


Figure 7.22 The cable length of the spreader l

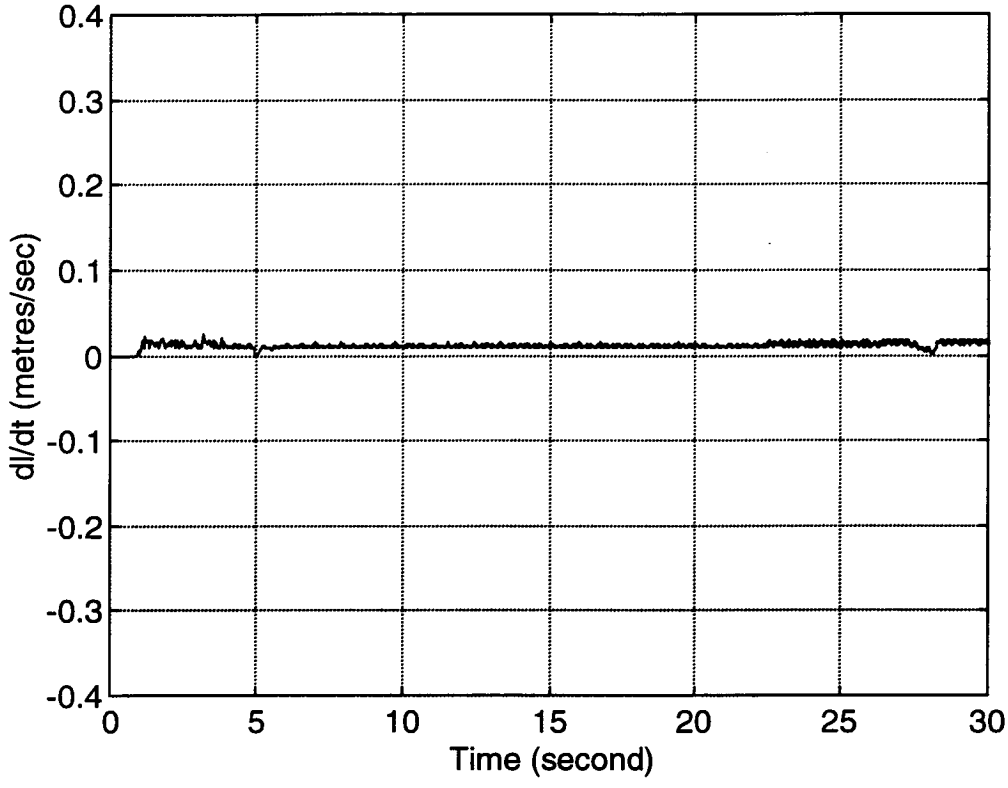


Figure 7.23 The linear velocity of the spreader in the l-direction

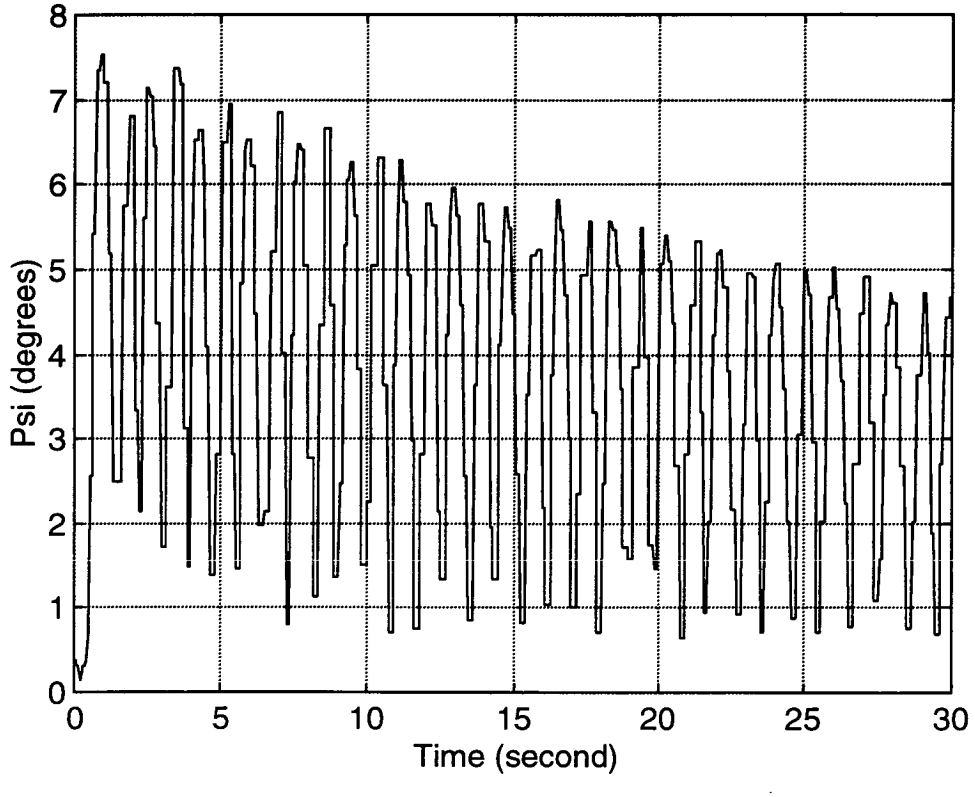


Figure 7.24 The swinging angle ψ

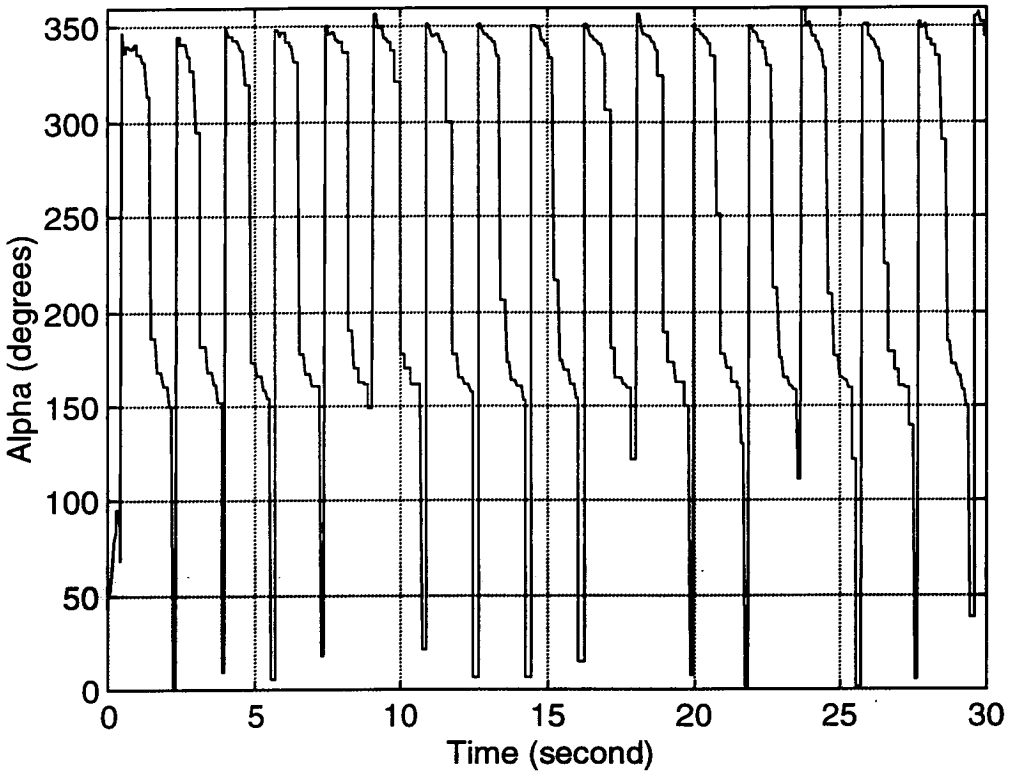


Figure 7.25 The polar angle α

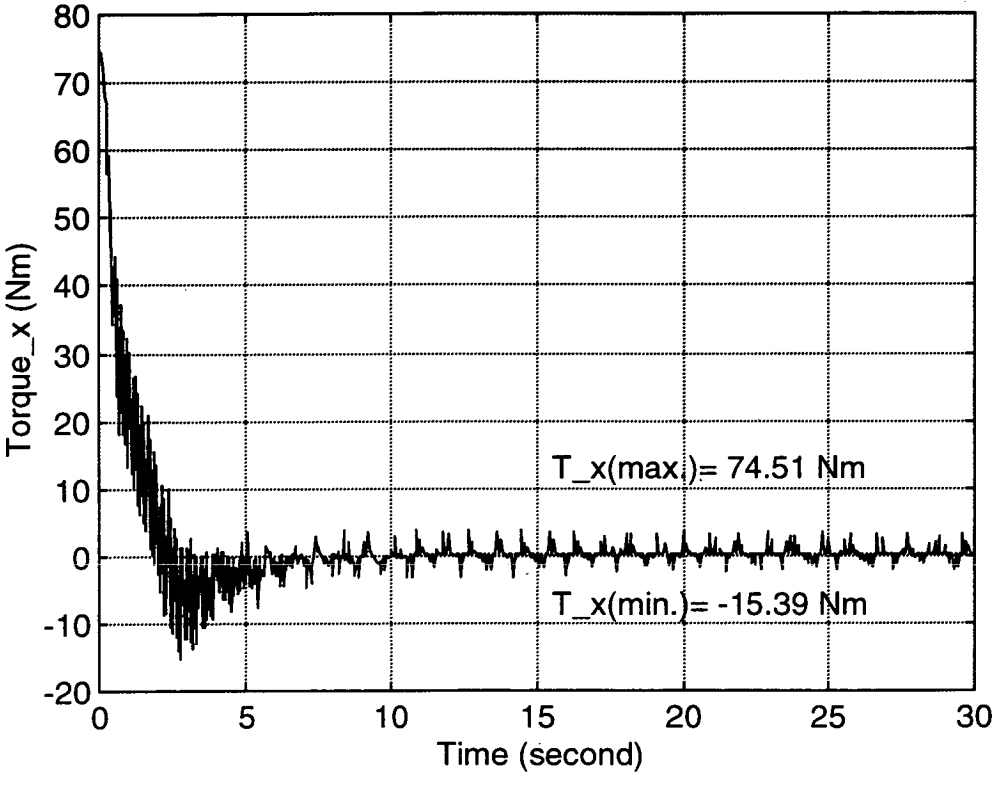


Figure 7.26 The required actuating torque of the trolley in the x- direction

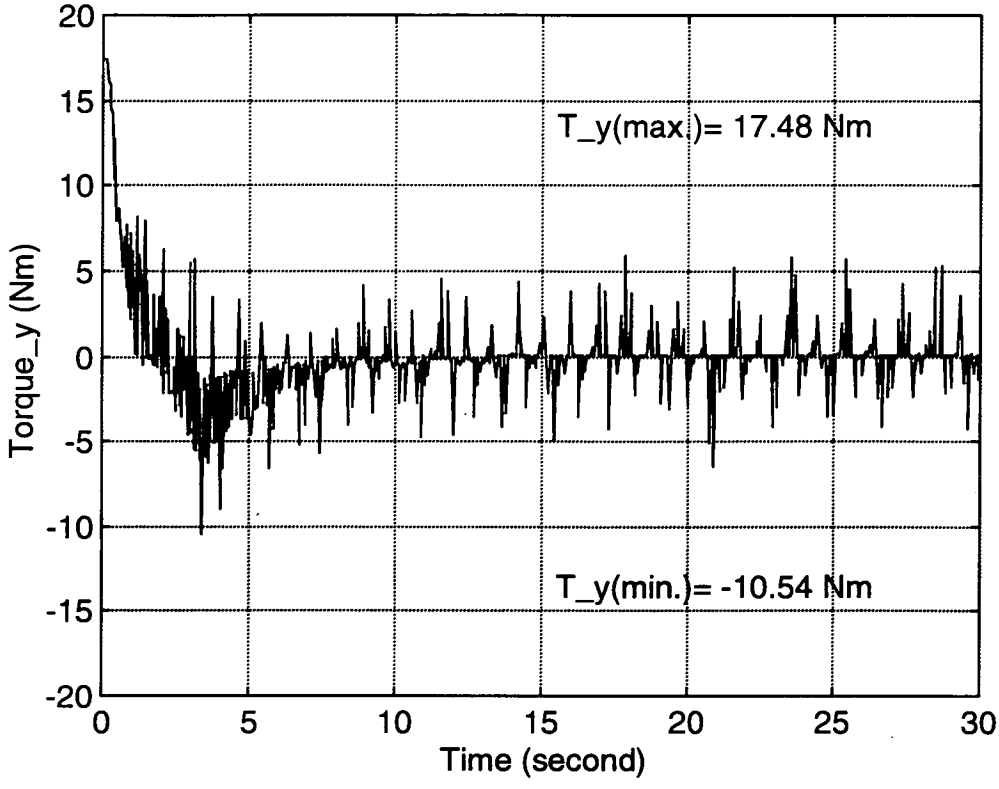


Figure 7.27 The required actuating torque of the trolley in the y- direction

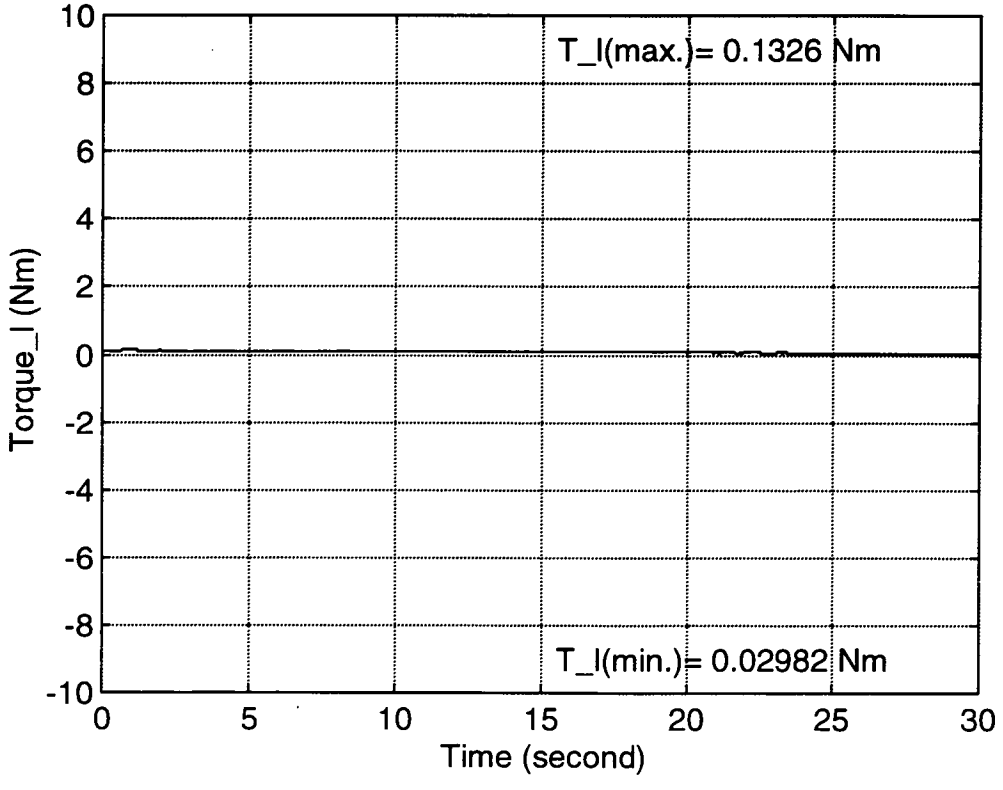


Figure 7.28 The required actuating torque of the trolley in the *l*- direction

Chapter 8

Conclusions and Recommendations for Future Work

8.1 Conclusions

The research described within this thesis deals with the dynamics and control of a RTG crane and a novel and flexible programming interface has been proposed for control of an experimental crane system. The contribution of this research can be separated into theoretical and experimental aspects. In theoretical terms the work has shown the following:

- A concept for a *Global Sensing System* (GSS) has been proposed, this being a closed loop automatic sensing system capable of guiding the spreader to the location of the target container by using feedback signals from the actuated degrees of freedom. The system can work with all of the motored degrees of freedom actuated simultaneously.
- Because of the difficulty in retrieving essential data for the swing angle and orientation of the moving spreader, a novel *Visual Sensing System* (VSS) has been proposed. This employs a camera-framegrabber combination to locate the coordinates of lighting sources (on the spreader) on an image plane. By using the mathematical model proposed for the GSS and a coordinate-system transformation derived in Chapter 2, the coordinates generated by the VSS can be transformed to physical coordinates relative to a fixed world coordinate system.
- A *Partial Search Algorithm* (PSA) has also been developed for the VSS. This is responsible for seeking the central coordinates of the clustered pixels of the lighting sources, without having to look into the entire digitised image from the CCD camera. In addition an *Improved Partial Search Algorithm* (IPSA) is proposed and this can be used for locating multiple lighting sources.

- The dynamics of the Super-RTG Crane by means of a single-cable-mode have been investigated in order to propose a suitable control strategy, *Feedback Linearization Control* (FLC). This can handle the significant non-linearity found in the cross-coupled governing equations. A simulation using *SIMULINK/MATLAB* has also been presented.
- A *Crane Application Programming Interface* (CAPI), utilising the metaphor of 'block diagrams', has been designed to integrate software (control strategies) and hardware (VSS, shaft encoders, actuators etc:) seamlessly without involving complex I/O routines between physical devices. It provides a relatively user-friendly environment in which the end-user can concentrate on implementing the more fundamental issues of control strategies rather than spending significant amounts of time in low-level hardware-dependent programming.
- The concept of a *Local Sensing System* (LSS) has been proposed. This makes use of laser technology to locate the *edges* of the target container. It can be seen as either a supplementary system to the GSS after the spreader has been automatically directed approximately above the target container, or as an independent retro-fitted piece of equipment for existing RTG cranes. Therefore, the position of the spreader above the container can be adjusted within the required tolerance for the final docking process.

The experimental program has resulted in further findings:

- The design and construction of a scaled-down, 1/8, size experimental rig has been completed to explore the feasibility of a new level of crane automation. It offers an excellent testbed for investigating different control strategies in practice. Moreover, all the components used, (such as the host computer, shaft encoders, the CCD camera and the framegrabber) are general purpose devices. This not only reduces the overall complexity of the system but also implies that such technology can readily be transferred to the practical application easily and cost-effectively.
- The implementation and assessment of different search algorithms such as the *Exhaustive Search Algorithm* (ESA), the PSA and the IPSA on the VSS has been explored in considerable detail. These routines have been programmed into an S-

Function block diagram to be used in real-time applications. Also a library of device-dependent S-Function blocks diagrams, such as Digital Input, Digital Output and Digital-Analogue-Converter (DAC), has been completed for efficient and high speed communications with the hardware.

- Real-time implementation on the integrated experimental rig using the CAPI and the VSS has demonstrated the feasibility of transferring the simulation work to direct control of the RTG crane.

8.2 Recommendations for future work

1. Taking the frictional term of the system (in Figure 3.1) into account.

   By adding the measured frictional term, $T_{f,n}$ in Table 6.1 to the 3-D RTG crane model, the governing equations (3.25), (3.29), (3.33), (3.38), (3.43), can be modified so the Feedback Linearization controller can reflect the effect of frictional torques in each activated degree of freedom. This would improve the performance where changing cable lengths of the spreader are involved and also give more accurate results in practice.

2. Implementing different control strategies, such as fuzzy logic control on the experimental rig.

   As the experimental rig and CAPI interface is designed with flexibility it would be sensible to test different control strategies in order to make the most of this integrated platform. Fuzzy logic control could be an alternative choice due to its ability to accommodate severe nonlinearities. The work done by M.P. Cartmell and L. Morrish [16] could be used as a starting point in which the authors proposed a fuzzy logic controller with similar system dynamics developed to those of this thesis. The results from different control methods can also be used to compared to the performance of the FLC control strategy presented in this thesis.

3. Introduction of the dynamics of the Multi-Cable-Mode (MCM) (Cartmell et.al. [15]) suspension system.

   By taking the dynamics of the Multi-Cable-Mode suspension system into account the GSS in Chapter 2 can provide the capacity for multi-cable spreader control, which not only involves actuations in the x-, y- and l- directions but also the orientation (R) of the spreader. The mathematical model and coordinate-system transformations of the GSS (and the S-function blocks library for this), which have all already been completed, can then be fully exploited. The outcome of the experiments could then be used to compare with the simulation results published by other researchers (Cartmell [14], Morrish [57]).

4. Completion of the experimental rig for the Local Sensing System (LSS).

   With the complete LSS the performance of both the line laser module and the spot laser module can be examined. This system should then be integrated with the GSS to verify the algorithms proposed in section 2.3.

5. Exploration of the possibility of a twin-camera visual sensing system.

   Because each camera can produce independent information from a lighting source and by knowing the distance between the two cameras (and the swing angles), the coordinates of the lighting source can then be defined. Therefore if three lighting sources are placed on the spreader, the centre of the spreader can be located as well as the orientation and the tilt angle of the spreader. The true potential of the IPSA, developed in Chapter 5, could be used in this way.

6. Examination of the feasibility of creating memory buffers for the counters of the experimental rig.

   This would mean that the controller would not be confused by readings from the counters caused by direction changing (as happened in section 7.3). The trolley can perform 'backward' movements to reduce the swinging motion of the spreader underneath. The improved counting facility could then be encapsulated again into a block diagram using the S-function format described in Chapter 6.

In summary, the work in this thesis not only provides an excellent test-bed in the form of a design for an experimental crane rig with high level of maneuverability, but also contributes a control strategy which proves to be capable of controlling this complex machine with satisfactory results. The proposal of an integrated programming interface (CAPI) gives a sound basis for investigating different control methods with much less difficulty by encapsulating the low-level device dependent I/O routines software compatible block diagrams. Furthermore, with the completion of the Global Sensing System (GSS), and the Local Sensing System (LSS) in the near future, it is believed that the automation of RTG crane systems should now be able to proceed. On this basis the technology proposed in this thesis should be transferred to the appropriate industrial sector as soon as possible.

Appendix A : References

[1] Al-Garni A.Z., Moustafa K.A.F., Javeed Nizami S.S.A.K.., 'Optimal control of overhead cranes', Control engineering practice, Vol. 3, no.9, September 1995, pp1277-1284.

[2] AM9513 Technical Manual, Advanced Micro Devices, inc.

[3] Archibald C., Petriu E., 'Robots skills development using a laser range finder', Conference Record IMTC/93, 1993, Irivin, USA, pp448-451.

[4] Auernig, J.W., Troger H., 'Time optimal control of overhead cranes with hoisting of the load', Automatica, Vol. 23, no. 4, 1987, pp437-447.

[5] Benhidjeb A., Gissinger G.L., 'Fuzzy control of an overhead crane performance comparison with classic control', Control engineering Practice, Vol. 3, no. 12, Dec. 1995, pp1687-1696.

[6] Biran A., Breiner M.,'MATLAB for engineers', 1995, Addison-Wesley.

[7] Bishop B., 'On the performance of state estimations for visual servo systems' Proceedings 1994 IEEE International Conference on Robotics and Automation, Vol.1 pp168-171.

[8] Bohlman M.T., Dougherty E.J., 'GRAIL- The Container Terminal of the Future', The society of naval architects and marine engineers, April 12-15, 1989, S2-1-1~S2-1-8.

[9] Bryfors U., 'Electronic positioning of container cranes', Freight international, 1992, pp.41-44.

[10] Butler H., Honderd G., Amerongen J.V., 'Model reference adaptive control of a gantry crane scale model', IEEE control system magazine, Vol.11, 1991, pp841-861.

[11] Burg T, Dawson D, Rahn C., Rohdes W., 'Nonlinear control of an overhead crane via the saturating control approach of Teel', Proceeding of IEEE international conference of robotics and automation, Vol.4 , April 1996, Minneapolis, USA, pp3155-3160.

[12] Bury B., 'Proximity sensing for robots', IEE Colloquium on 'Robots Sensor', Digest no.16, 1991, London, UK, pp3/1-3/18.

[13] The reference note for C-Vision/FM960, FM960.004, 4C's Vision & Control inc.

[14] Cartmell M.P. Morrish L., Taylor A. 'Control of a mobile gantry crane incorporating multiple cable suspension of the lifting gear', Mechatronics- the basis for new Industrial Development1, 1994, pp3-8.

[15] Cartmell M.P. Morrish L., Taylor A. 'Dynamics of spreader motion in a crane gantry', Proceedings of the institution of mechanical engineers, part C, Journal of mechanical engineering sciences (in press, to be published March 1998).

[16] Cartmell M.P. , Morrish, L., Alberts T.E. ; 'Controlling the nonlinear dynamics of gantry cranes', Machine vibration, Vol. 5, 1996, pp197-210.

[17] Casper B., 'Electric reactions', Cargo system, August 1989, pp69-71.

[18] Champion B., 'Swayed by arguments?', Cargo system, August 1989, pp63-67.

[19] Charlet B., Levine J., Marino R. 'Sufficient conditions for dynamic state feedback linearization', SIAM Journal of control and optimization, Vol.29, 1991, pp38-57.

[20] Chung C, Hauser J., 'Nonlinear control of a swing pendulum', Automatica, Vol. 31, 1995, no. 6, pp851-852.

[21] Craig J. J. ; Adaptive control of mechanical manipulators, 1988, Addison-Wesley.

[22] Craig J. J. ; Introduction to robotics : mechanisms and control , 1989, Addison-Wesley.

[23] d'Andrèa-Novel B., Boustany F., 'Adaptive control of a class of mechanical systems using linearization and Lyapunov methods. A comparative study on the overhead crane example', Proceedings of the IEEE conference on decision and control, Jan. 1992, pp120-125.

[24] The Catalogue of Data acquisition & control, Vol. 7, ComputerBoards inc.

[25] Dodman K., 'Crane Automation', Cargos today, March 1991, pp41-44.

[26] Dougherty E.J., Lee D.E., Shively P.D. 'Automated all-weather cargo transfer system', The society of naval architects and marine engineers, April 12-15, 1989, S2-3-1.

[27] *The Catalogue of Product Handbook,* 1996, Data Translation inc.

[28] *The Catalogue of Products,* 1996, Ealing Electro-Optics Ltd.

[29] Feddema J.T., 'Model-based visual feedback control for a hand-eye coordinated robotic system', Computer, Vol.25, August, 1992, pp21-31.

[30] The technical specification of RTG crane, FEL International Ltd.

[31] The technical specification for Ferranti wide span gantry crane - Type FD40

[32] Fu, K.S., Gonzalez R., Lee C.S.G., 'Robotics: control, sensing, vision, and intelligence', 1987, McGraw-Hill

[33] Fukuda T., 'Fuzzy, neural network, and genetic algorithm based control system' IECON '94, 20th international conference on industrial electronics, control, and instrumentation, Vol.2, 1994, pp1220-1225.

[34] Gonzalez R. 'Digital signal processing', 1992, Addison-Wesley.

[35] Hämäläinen J.J. Marttinen A., Virkkunen J.,'Optimal path planning for a trolley crane: fast and smooth transfer of load' IEE Proc.-Control Theory Appl. , V.142, no.1, Jan., 1995, pp51-57.

[36] Harashima F., 'Sensor based robot control systems', Proceedings of the 1990 IEEE Colloquium in South America, 1990, pp203-208.

[37] Hashimoto Y., 'Energy dissipation control and its application' Optimal Control Appl. Mech. , V.3, 1982, pp115-120.

[38] Hickling R., Marin S.P., 'The use of ultrasonic for gauging and proximity sensing in air', Journal of acoustics of society of America, vol. 79, 1986, pp1303-1310.

[39] Huang K.C., Cartmell M.P. , 'Report on fundamental design considerations for a local sensing and control system for RTG spreader, 1996, Dept. of mechanical Engineering in Edinburgh University.

[40] Hytönen K., 'Guiding force', Container Management, May 1997, pp. C29-C33.

[41] Javed M.A., Sanders S.A.C. 'An artificial neural networks for applications in automated industrial systems', IECON '91, Proceedings of IEEE 1991 international onference on industrial electronics, control, and instrumentation, Vol.2, 1991, pp1496-1499.

[42] Joshi S., Rahn C., 'Position control of a flexible cable gantry crane: theory and experimentation', Proceeding of American control conference, Vol. 4, June 1995, Seattle, pp2820-2824.

[43] Key J., Stone R., Rehder M., 'Estimating high latitude readiative fluxes from satellite data; problem and sucess', IGARSS '94 International Geoscience and Remote Sensing Symposium, Surface and Atmospheric Remote Sensing, Technology, Data Analysis, and Interpretation, 1994, Vol. 2, pp1018-1020.

[44] Khatib O., 'A unified approach for motion and force control of robot manipulators: the operational space formulation', IEEE journal of robotics and automation, Vol.3, no.1., 1987, pp43-53.

[45] Kijima Y., Ohtsubo R., 'An optimization of fuzzy controller and its application to over ', Procedding of the 1995 IEEE INCON 21st International Conference on industrial electronics, control and instrumentation, 1995, Vol. 2, pp1508-1513.

[46] Kreyszig E., 'Advanced enginering mathematics' - 7th edition, 1993, John Wiley & Sons.

[47] Leonard N.E., Levine W.S., 'Using MATLAB to analyze and design control systems - 2nd edition', 1995, Addison-Wesley.

[48] Macleod R., 'Developing an automated container terminal', Freight international, 1992, pp37-38.

[49] Maravall D., 'Adaptive control of a video camera for the automatic detection and tracking of mobiles', Conference Proceedings 1993 International Conference on Systems Man and Cybernetics, System Engineering In the service of humans, Vol.2, 1993, pp53-58.

[50] Martindale S., Dawson D., Zhu J., Rahn C., 'Approximate nonlinear control for a two degree of freedom overhead crane: theory and experimentation', Proceeding of American control conference, Vol. 1, June 1995, Seattle, pp301-305.

[51] Marttinen A., Virkkunen J., Seitsonen J., 'Pilot crane control - Part I', SÄHKÖ Elect. Electron., Vol. 62, no. 9, 1989, pp34-38.

[52] Marttinen A., Virkkunen J., Seitsonen J., 'Pilot crane control - Part II', SÄHKÖ Elect. Electron., Vol. 62, no. 11, 1989, pp22-26.

[53] Marttinen A., Virkkunen J., Seitsonen J., 'Control study with a pilot crane', IEEE transaction on education, Vol. 33, no. 3, 1990, pp298-305.

[54] Mason G.A.,'Time optimal control of an overhead crane model' Optimal control applied mech. , V.3, 1982, pp115-120.

[55] Merryman H., 'Diagnostic technique for power system using infrared thermal imaging', IEEE transaction on industrial electronics, Vol.42, Issue 6, pp615-628.

[56] Morris A., 'The essence of measurement', 1996, Prentice Hall inc.

[57] Morrish L., 'Modelling of spreader hoist systems in mobile gantry cranes', Ph.D. thesis, 1996, University of Edinburgh

[58] Moustafa K.A.F, Ebeid A.M., 'Nonlinear modeling and control of overhead crane load sway', Journal of dynamic systems, measurement, and control, Vol. 110, 1988, pp266-271.

[59] Okamoto K., '3D object recognition using laser range finders', 91 ICAR Fifth International Conference on Advanced Robotics, Robots in Unconstructed Environments., Vol.2, pp1245-1250.

[60] Pallás-Areny R.,Webster J.G., 'Sensors and digital conditioning', 1991, John Wiley & Sons.

[61] Paul R. P., 'Modeling, Trajectory calculation, and servoing of a computer controlled arm', Technical report AIM-177, Stanford University Artificial Intelligence Laboratory, 1972.

[62] RS data sheet, 171-253, for the pixel line sensor.

[63] RS data sheet, F18534, for the laser diode.

[64] RS data sheet, B14699, for dc speed control module and accessories.

[65] Rudolf CD, 'Swaying at gantries', Cargo systems, June 1992, pp27-31.

[66] Rogers E., 'Generic tools for real-time simulation and control of complex systems', IEE Colloquium on 'High Performance Computing for Advanced Control' London, UK, 1994, pp3/1.

[67] Sakawa Y.,'Optimal control of container crane' Automatica , V.18, 1982, pp257-266.

[68] Sakawa Y., Nakanumi A., 'Modeling and control of a rotary crane', Journal of dynamics systems, measurement and control, Vol. 107, Sept. 1985, pp200-206.

[69] Serway R. A. (1982) , *Physics: For Scientists and Engineers* , pp764-90, CBS College Publishing.

[70] Shirai y., Inoue H., 'Guiding a robot by visual feedback in assembling tasks', Pattern Recognition, 5, 1973, pp99-108.

[71] Spong M.W. , Vidyasagar, M. ; 'Robot Dynamics and control', 1989, John Wiley and Sons.

[72] Starr G., 'Swing-free transport of suspended objects with a path-controlled robot manipulator', Journal of dynamics systems, measurement and control, Vol. 107, Mar. 1985, pp97-100

[73] The Student Edition of MATLAB, Ver.4 , 1995, Mathworks Ltd.

[74] Suzuki Y., Yamada S., Fujikawa H., 'Anti-swing control of the container crane by fuzzy control', Proceedings of the 19th annual international IEEE conference on industrial electronics, control, and instrumentation, Hawaii, USA, Vol.1, Nov. 1993, pp230-235.

[75] Tang R., Jalel N.A., Mirzai A.r. 'Identification and modeling of fermentation process using MATLAB: a case study', Modeling and control of biotechnical processes, IFAC '92, Colorado, USA, 1992, pp331-334.,

[76] The Technical Note 1812, 'Real-Time Workshop and *SIMULINK* Accelerator - Questions and Answers', Version 1.0, 1997, Mathworks inc.

[77] Tekalp A., 'Digital video processing', 1995, Prentice-Hall inc.

[78] Tong R.M., A control engineering review of fuzzy systems', Automatica, Vol.13, 1977, pp679-686.

[79] Tong W., Lu G., 'Data structure for segmenting binary image', Proceeding TENCON '93 1993 IEEE Region 10 Conference on 'Computer Communication, Control and Power Engineering', Vol. 2, Oct. 1993, pp1146-1149.

[80] Torp S., Nogaard P.M., 'Implementation issues in CACSD', Proceeding. IEEE/IFAC Joint Symposium on Computer-Aided Control System Design, Tucson, USA, March 1994, pp527-532.

[81] The Universal Library programmers manual, rev. 3.2, June 1995, ComputerBoards inc.

[82] The user's guide for MATLAB 4.2 , 1993, Mathworks Ltd.

[83] The user's guide for SIMULINK 1.3c , 1995, Mathworks Ltd.

[84] The user's guide for Real-Time Workshop 1.1 , 1996, Mathworks Ltd.

[85] The user's guide for Watcom C/C++ Compiler, Version 10.6, 1996, PowerSoft Inc.

[86] The user's manual for CIO-CTR10, revision 2, Feb. 1994, ComputerBoards inc.

[87] The user's manual for CIO-DAC08, revision 2, Feb. 1994, ComputerBoards inc.

[88] Virkkunen J., Marttinen A., Seitsonen J., 'Computer control of overhead and gantry cranes', IFAC Symposia Series - 11th Triennial World Congress., Vol. 4, no. 9, 1991, pp401-406.

[89] Verschoff I.J., 'Automated Stacking cranes at ECT', Freight international, 1992, p31-33.

[90] Wang Z., 'Adaptive self-organizing neural network method for tracking problems of nonlinear dynamic systems, 1994 IEEE International Conference on Neural Networks - IEEE World Congress on Computational Intelligence, Vol.5, pp2793-2796.

[91] Waddicor P.J. 'Development of automatic positioning control of freight container lifting equipment', 1996, Dept. of Mechanical Engineering, Bolton Institute of Higher Education.

[92] Wen W., Durrant-Whyte H.F., 'Model based active object localisation using multiple sensors', IEEE International Workshop on Intelligent Robots and Systems IROS '91, , Osaka, Japan, Nov. 3-5, 1991pp1448-1452.

[93] Woods J.D., Clements D.J., 'Nonlinear control of overhead cranes', National conference publication- Institution of engineers, Australia', No. 92, 1992, pp29-40.

[94] Yamada S., Fujikawa H., 'Fuzzy control of the roof crane', Proceeding of 15th annual conference of IEEE industrial electronics society - IECON'89, Philadelphia, Pennsylvania, Vol.4, November 1989, pp709-714

[95] Yong H.K., 'Localization of a mobile using a laser range finder in a hierarchical navigation system', Proceeding IEEE Southeastcon '93, 1993, Charlotte, USA, p. 4p.

[96] Yoon J.S., Park B.S., Lee J.S., Park H.S., 'Development of anti-swing control algorithm for the overhead crane' Proceedings of the conference on remote system technology, 1991, pp89-94.

[97] Zadeh L.A., 'Outline of a new approach to the analysis of complex systems and decision processes', IEEE transcations on systems man and cybernetics, 1973 SMC-3, pp28-44.

Appendix B : The proof of the statement in section 3.4

The **x**-coordinate is chosen as an example for the following proof.

The gain h_x in x-coordinate is determined according to equation (3.75) with an initial position error ε'_{x0}, which is smaller than the maximum span $\varepsilon_{x0,\max}$. That is,

$$|\varepsilon'_{x0}| < |\varepsilon_{x0,\max}| \qquad \text{------- (B.1)}$$

From equation (3.75), the corresponding gain h'_x is,

$$h'_x = 2\frac{e \cdot \dot{x}_{\max}}{|\varepsilon'_{x0}|} \qquad \text{------- (B.2)}$$

where e is the natural logarithm exponent and $\dot{x}_{\max}$ is the physical maximum velocity of the x- actuator (equation (3.70)).

From equation (3.68), the corresponding velocity in the x-coordinate, $\dot{\varepsilon}'_x$, with ε'_{x0} will be,

$$\dot{\varepsilon}'_x = -a^2 \varepsilon_{x0,\max} t e^{-at} \qquad \text{------- (B.3)}$$

where

$$a = -\frac{h'_x}{2} \qquad \text{------- (B.4)}$$

According to equations (3.71) and (3.72) , the maximum value of $\dot{\varepsilon}'_x$ in equation (B.3) will occur when $t = \frac{1}{a}$.

Therefore,

$$\left(\dot{\varepsilon}'_x\right)_{\max} = -a\varepsilon_{x0,\max}e^{-1} \qquad \text{-------} \quad \text{(B.5)}$$

With equations (B.2) and (B.4),

$$\left(\dot{\varepsilon}'_x\right)_{\max} = -\frac{h'_x}{2}\varepsilon_{x0,\max}e^{-1} \qquad \text{-------} \quad \text{(B.6)}$$

On substituting equation (B.4) to (B.6),

$$\left(\dot{\varepsilon}'_x\right)_{\max} = \frac{\varepsilon_{x0,\max}}{\left|\varepsilon'_{x0}\right|}\dot{x}_{\max} \qquad \text{-------} \quad \text{(B.7)}$$

From equation (B.1),

$$\frac{\left|\varepsilon_{x0,\max}\right|}{\left|\varepsilon'_{x0}\right|} > 1 \qquad \text{-------} \quad \text{(B.7)}$$

By taking the *absolute value* of both sides of equation (B.6), it is concluded that,

$$\left|\left(\dot{\varepsilon}'_x\right)_{\max}\right| > \dot{x}_{\max} \qquad \text{-------} \quad \text{(B.8)}$$

Appendix C : The description of the configuration file of the VSS

[a] The 'data' file

The 'data' file, which holds all the essential parameters, is downloaded to the FM960 to setup all the hardware. The filename defaults to "data", but may be specified in the command line. Each of these parameters takes the form "name=value", where "name" is a text string identifying the parameter being defined and "value" is the desired value. It should be noted that no spaces are allowed in a parameter definition, but "value" must be followed by a space. Any characters after this space are ignored, thus allowing comments to be inserted in the file for readability. An entire line may be commented by a "*" in the left-hand column.

Sample data file.

---

*Grabber setup parameters:	
IOBase=300	Card I/O address (in hex)
FrameX=200	Left margin
FrameWidth=400	Width
FrameY=40	Top margin
FrameHeight=300	Height
FrameDepth=1	Can be 8 or 1. 8=grey scale image, 1 = black & white
FrameCount=1	Set this to the number of images to be captured on the card each time a grab operation is carried out (This number is clipped to the max. possible in the available RAM).
FrameReduceFactor=2	Specifies the amount of reduction of each image when displaying it on the screen (This does not affect the actual size of images captured in the memory).
Interlace=1	Interlace (1=interlace, 0=non-interlace)
InvertInterlace=0	Interlace order (0=lower line first, 1=upper line first)
SyncSource=1	Sync source (1, 2 or 3)

SyncSliceLevel=125	Sync slice level (25, 50, 100 or 125)
HiRefLevel=1.2	Upper voltage reference level (0.0 - 1.2)
LoRefLevel=0.3	Lower voltage reference level (0.0 - 1.2)
ClockBase60=0	Clock base (0=57.272 MHz, 1=60 MHz)
ClockSourceExternal=0	External clock (1=external, 0=internal)
ExtClockInvert=0	Invert external clock (1=inverted, 0=non-inverted)
LineuSecs=64.0	Line length (microsecs)
ClockDiv=4	Clock divide value (e.g. 57.272MHz / 4 = 14.318MHz)
ClockFreq=14.31818	Clock freq. (MHz)
HSyncTime=4.7	Horizontal sync position (microsecs)
VSyncLines=0	Vertical offset in lines (ignoring interlace)
ClampStartTime=8.5	Clamp start time (microsecs)
ClampStopTime=9.0	Clamp stop time (microsecs)
ZeroStartTime=8.5	Zero start time (microsecs)
ZeroStopTime=8.75	Zero stop time (microsecs)
Analog=1	Analogue input (1=analogue, 0=digital)
ExtTrigger=1	Capture waits for low-high trigger pulse on pin 11
ThresholdLevel	Threshold level used for black & white images
960Filename=	Name of 960 programs to download & run on the FM960 (normally main.4cs. Leave blank if flash programmed).

LUT=256	256 values for the input LUT
0	First value
1	Second value
2	Third value
.	etc.
.	
.	
255	

[b] The introduction of FM960 Developers Pack.

The FM960 Developers Kit consists of a GRABS960 application that continuously captures images & displays them on the PCs screen. GRABS960 is in two parts, GRABS960.PC & GRABS960.960. GRABS960.PC is a DOS program that runs on a VESA compatible card capable of displaying 800x600x256 colours. GRABS960.960 is a 960 program that GRABS960.PC downloads to the FM960 on start-up. GRABS960.960 is a program run on the 960. This should be used as the foundation for the software, and it contains a message loop that waits for commands from the PC which are then executed.

1. Installing the FM960 Developers Pack

Before installing the source code, install the Intel GNU960 compiler on your system.

Copy the \FM960 directory onto your hard disk. This will create the following directory structure :-

```
e.g.    \FM960 ------- GRABS960.960------
                |                    |
                |                    ------ OBJ
                |
                -------- GRABS960.PC
```

PC Code - \GRABS960.PC

GRABS960.PC compiles under Microsoft Visual C++ using the supplied .MAK file, but should compile under any compiler with a few minor modifications (screen & I/O function names).
The program is made up from several source files :-

GRABS.C The main program loop.

SCREEN.C	Functions for setting display modes and displaying images on the screen.
CONTROL.C	This module contains FM960 command calls. Calls are made to Command960() in FUNCS960.C. This function sends the command code & any parameters to the FM960 as well as receiving any return values.
FUNCS960.C	Contains the Command960() command handler function as well as functions for downloading a 960 program.
COMMS960.C	This file contains communication functions for sending & receiving data to the 960.

960 Code - \GRABS960.960

GRABS960.960 uses the Intel GNU960 Compiler. A set of batch files are supplied in the GRABS960.960 directory to compile, assemble & link the source files. These batch files are :-

C.BAT	Compiles & assembles a C source code file into object code.
CNO.BAT	Compiles & assembles without optimizations
CS.BAT	Compiles & assembles a C source code and also creates a .L file with the generated assembler code in-line with the C for hand-optimization purposes.
A.BAT	Assembles a .ASM file into object code.
L.BAT	Build the .4CS file ready to be run on the FM960. This process involves linking all object code files listed in GLNK into a .COF file & then converting this to a .4CS file using COFF4CS.EXE.

All the above bat files require a filename to be passed as a parameter (i.e. C MAIN to compile MAIN.C). The batch files also assume that the GNU960 compiler environment variables have been set up correctly (COMPILE.BAT contains an example of the required settings. You can alter the directory locations and use them to set up the variables).

Finally, ALL.BAT re-compiles all source files for GRABS960.960 and links the file together.

The program contains the following source code files :-

MAIN.C	Contains the main message loop & functions for initializing any variables, clearing memory etc.
COMMSPC.C	Contains functions for communicating with the PC (Sending & receiving data)
GRAB.C	Functions for capturing images into memory.
SEQUENCE.C	Functions for allocating space for sequences of images are stored here.
ERR.C	Contains a function required by the Intel GNU compiler that returns the address of errno.

Object code files are created in the OBJ sub-directory.

In-order for any newly compiled versions of GRABS960.960 to be sent to the FM960 card, it is necessary either copy the new MAIN.4CS into the GRABS960.PC directory or change the 960Filename line of the file DATA in GRABS960.PC to include the path for the MAIN.4CS file to point at the GRABS960.960\MAIN.4CS file.

Appendix D : An example program of the device driver block

```
/*dac08.c*********************************************/
/* dac08da.c
 *
 *  S-Function device driver for the Computer Boards
 *  CIO-DAC08 analog output of the board.
 *
 *  Created by K Huang in 26/03/97
 */
/* Revised 230697 */
#undef S_FUNCTION_NAME
#define S_FUNCTION_NAME dac08da       /* Do NOT change it */

#include <stddef.h> /* For NULL  */
#include <stdlib.h> /* For min   */
#include <math.h>   /* For pow() */

#include "simstruc.h" /* Where simulation structure, S, is defined */

#ifdef MATLAB_MEX_FILE
#include "mex.h"
#endif

#ifndef min
#define min(x, y)      ((x) < (y) ? (x) : (y))
#endif

/* compiler dependant low level hardware calls */
#ifdef __HIGHC__
     #include <conio.h>
     #define   hw_outportb( portid, value )    _outp( portid, value )
     #define   hw_inportb( portid )            _inp( portid )
     #define   hw_outport( portid, value )     _outpw( portid, value )
     #define   hw_inport( portid )             _inpw( portid )
#elif __WATCOMC__
     #include <conio.h>
     #define   hw_outportb( portid, value )    outp( portid, value )
     #define   hw_inportb( portid )            inp( portid )
     #define   hw_outport( portid, value )     outpw( portid, value )
     #define   hw_inport( portid )             inpw( portid )
#else
static void
hw_outportb(portid, value)
int portid;
int value;
{
}

static int
hw_inportb(portid)
int portid;
{
     return(0);
}
```

```
static void
hw_outport(portid, value)
int portid;
int value;
{
}

static int
hw_inport(portid)
int portid;
{
    return(0);
}
#endif

/* Input Arguments */
#define BASE_ADDRESS_ARG        ssGetArg(S,0)
#define OUTPUT_REFERENCE_ARG    ssGetArg(S,1)
#define NUM_CHANNELS_BITS_ARG   ssGetArg(S,2)
#define SAMPLE_TIME_ARG         ssGetArg(S,3)
#define ACCESS_HW_ARG           ssGetArg(S,4)
#define NUMBER_OF_ARGS          (5)

/* Indeces into vectored input arguments */  /* for ssGetArg(S,2) */
#define CHANNELS                (0)
#define NUM_CHANNELS_BITS       (1)

#define NSAMPLE_TIMES           (1)         /* for ssGetArg(S,3) */

#define MAX_OUTPUTS             (8)         /* 8 channels for DAC08 */

/*********************************************************************/
/* Storage Allocation */                    /* for ssSetIWorkValue() */
/* Each output channel's resolution and number of bits is stored in an integer
  work vector.  Then add the other storage space needed. */

#define BASE_ADDRESS            (16)
#define NUM_CHANNELS            (17)
#define ACCESS_HW               (18)
#define NUMBER_OF_IWORKS        ((MAX_OUTPUTS * 2) + 3)

/* Storage Allocation */                    /* for ssSetRWorkValue() */
#define OUTPUT_REFERENCE        (0)
#define NUMBER_OF_RWORKS        (1)
/*********************************************************************/

#define DAC_ZERO        (0)
#define DAC_MAX_VOLTAGE (4095)
#define DAC_MIN_VOLTAGE (0)
#define DAC_MAX_BITS    (12)
#define BIT_DIFFERENCE  (0)


static void mdlInitializeSizes(S)
    SimStruct *S;
{
    int num_channels;
```

```
    if (ssGetNumArgs(S) != NUMBER_OF_ARGS) {
#ifdef MATLAB_MEX_FILE
        mexErrMsgTxt("Wrong number of input arguments passed.\nFive arguments are
expected\n");
#endif
    }
    /* Check the size of the number of channels,
     * number of bits per channel input.
     */
    else if (mxGetN(NUM_CHANNELS_BITS_ARG) < NUM_CHANNELS_BITS)
    {
#ifdef MATLAB_MEX_FILE
        mexErrMsgTxt("The size of the Number of Channels, Bits Per Channel vector must be at
least 1\n");
#endif
    } else {
        num_channels = mxGetPr(NUM_CHANNELS_BITS_ARG)[CHANNELS];
#ifdef MATLAB_MEX_FILE
        if ((num_channels < 1) || (num_channels > MAX_OUTPUTS)) {
            mexErrMsgTxt("Number of output channels must be between 1 and 8\n");
        }
#endif
        /* Set-up size information */
        ssSetNumContStates(S, 0);
        ssSetNumDiscStates(S, 0);
        ssSetNumOutputs(S, num_channels);
        ssSetNumInputs(S, num_channels);
        ssSetDirectFeedThrough(S, 0); /* No direct feedthrough */
        ssSetNumSampleTimes(S, NSAMPLE_TIMES);
        ssSetNumInputArgs(S, NUMBER_OF_ARGS);
        ssSetNumIWork(S, NUMBER_OF_IWORKS); /* D/A card hardware register pointers */
        ssSetNumRWork(S, NUMBER_OF_RWORKS); /* D/A card output reference voltage */
    }
}

/* Function to initialize sample times */
static void mdlInitializeSampleTimes(S)
    SimStruct *S;
{
    ssSetSampleTimeEvent(S, 0, mxGetPr(SAMPLE_TIME_ARG)[0]);
    ssSetOffsetTimeEvent(S, 0, 0);
}

static void mdlInitializeConditions(x0, S)
    double *x0;
    SimStruct *S;
{
    int arg_str_len = 128;
    char arg_str[128];
    unsigned int base_addr;
    int num_channels;
    double output_reference;
    int access_hw;
    int num_bits;
    int dac_resolution[MAX_OUTPUTS];
    int num_dacs;
    register i;
```

```
    mxGetString(BASE_ADDRESS_ARG, arg_str, arg_str_len);
    base_addr = (unsigned long)strtol(arg_str, NULL, 0);
    ssSetIWorkValue(S, BASE_ADDRESS, (int)base_addr);

    num_channels                    =                    min(MAX_OUTPUTS,
(int)mxGetPr(NUM_CHANNELS_BITS_ARG)[CHANNELS]);
    ssSetIWorkValue(S, NUM_CHANNELS, num_channels);

    /* Initialize the ADC resolution and number
     * of bits to their respective maximums.
     */
    for (i=0; i < num_channels; i++)
    {
        ssSetIWorkValue(S, i, DAC_MAX_VOLTAGE);
        ssSetIWorkValue(S, i + MAX_OUTPUTS, BIT_DIFFERENCE);
    }

    num_dacs = min(num_channels, (int)mxGetN(NUM_CHANNELS_BITS_ARG) - 1);

    /* Now save the bit difference and the new resolution
     * to integer storage space.
     */
    if (num_dacs > 0)
    {
        for (i=0; i < num_dacs; i++)
        {
            num_bits = (int)mxGetPr(NUM_CHANNELS_BITS_ARG)[i + 1];
#ifdef MATLAB_MEX_FILE
            if ((num_bits < 1) || (num_bits > DAC_MAX_BITS)) {
                mexErrMsgTxt("The number of bits must be between 1 and 12\n");
            }
#endif
            dac_resolution[i] = (int)pow(2.0, (double)num_bits) - 1;
            ssSetIWorkValue(S, i, dac_resolution[i]);
            ssSetIWorkValue(S, i + MAX_OUTPUTS, DAC_MAX_BITS - num_bits);
        }
    }

    output_reference = mxGetPr(OUTPUT_REFERENCE_ARG)[0];
#ifdef MATLAB_MEX_FILE
    if ((output_reference < 0.0) || (output_reference > 10.0)) {
        mexErrMsgTxt("The output reference voltage must be LARGER than 0.0 ,and SMALLER
or EQUAL to 10.0 volts\n");
    }
#endif
    ssSetRWorkValue(S, OUTPUT_REFERENCE, output_reference);

    access_hw = (int)(mxGetPr(ACCESS_HW_ARG)[0]);
    ssSetIWorkValue(S, ACCESS_HW, access_hw);

    /* If this is a MEX-file, then accessing the hardware
     * depends on the value of the access hardware flag in the
     * S-function's dialog box.
     */
#ifdef MATLAB_MEX_FILE
    if (!access_hw)
        return;
```

```
#endif /* MATLAB_MEX_FILE */

    /*
     * Initialize the board outputs to be zero volts (0.0).
     */
    for(i=0;i<num_dacs;i++){
        hw_outportb(base_addr+ 2*i , DAC_ZERO);
        hw_outportb(base_addr+ (2*i) + 1 , DAC_ZERO);
        }

}

/* Function to compute outputs */
static void mdlOutputs(y, x, u, S, tid)
    double *y;
    double *x;
    double *u;
    SimStruct *S;
    int tid;
{
    /* This function writes to the DAC on board.  First, the value is
     * truncated between -4.95 volts and +4.95 volts.  Then the voltage
     * value is converted to binary, broken into one byte and one nibble,
     * and written out to the DAC.
     */
    unsigned int base_addr = ssGetIWorkValue(S, BASE_ADDRESS);
    int num_channels = ssGetIWorkValue(S, NUM_CHANNELS);
    double output_reference = ssGetRWorkValue(S, OUTPUT_REFERENCE);
    int low_byte;
    int high_byte;
    register i;
    int value;
    int dac_resolution;
    int bit_diff;

    /* If this is a MEX-file, then accessing the hardware
     * depends on the value of the access hardware flag in the
     * S-function's dialog box.
     */
#ifdef MATLAB_MEX_FILE
    int access_hw = ssGetIWorkValue(S, ACCESS_HW);
    if (!access_hw)
    {
        /* Allow the inputs to be directly passed to the
         * outputs for a loopback situtation.  Note that
         * to use this capability, an output port needs to
         * be added to the block.
         */
        for (i = 0; i < num_channels; i++) {
            y[i] = u[i];
        }
        return;
    }
#endif /* MATLAB_MEX_FILE */

    for (i = 0; i < num_channels; i++) {
        dac_resolution = ssGetIWorkValue(S, i);
        bit_diff = ssGetIWorkValue(S, i + MAX_OUTPUTS);
```

```
    for(i=0;i<num_channels;i++){
        hw_outportb(base_addr+ 2*i , DAC_ZERO);
        hw_outportb(base_addr+ (2*i) + 1 , DAC_ZERO);
        }
}

#ifdef MATLAB_MEX_FILE  /* Is this file being compiled as a MEX-file? */
#include "simulink.c"   /* Mex glue */
#else
#include "cg_sfun.h"    /* Code generation glue */
#endif
```