National Library of Canada
Acquisitions and Bibliographic Services Branch
395 Wellington Street
Ottawa, Ontario
K1A 0N4

Bibliothèque nationale du Canada
Direction des acquisitions et des services bibliographiques
395, rue Wellington
Ottawa (Ontario)
K1A 0N4

Your file Votre référence

Our file Notre référence

NOTICE

The quality of this microform is heavily dependent upon the quality of the original thesis submitted for microfilming. Every effort has been made to ensure the highest quality of reproduction possible.

If pages are missing, contact the university which granted the degree.

Some pages may have indistinct print especially if the original pages were typed with a poor typewriter ribbon or if the university sent us an inferior photocopy.

AVIS

La qualité de cette microforme dépend grandement de la qualité de la thèse soumise au microfilmage. Nous avons tout fait pour assurer une qualité supérieure de reproduction.

S'il manque des pages, veuillez communiquer avec l'université qui a conféré le grade.

La qualité d'impression de certaines pages peut laisser à désirer, surtout si les pages originales ont été dactylographiées à l'aide d'un ruban usé ou si l'université nous a fait parvenir une photocopie de qualité inférieure.

Canada

The Development and Analysis of a Multi-processor Based Intelligent Robotic Workcell

Simon Chi-leung Poon

A Thesis

in

The Department

of

Mechanical Engineering

Presented in Partial Fulfillment of the Requirements
for the Degree of Master of Engineering at
Concordia University
Montreal, Quebec, Canada

1990

ISBN 0-315-97707-8

ABSTRACT

The Development and Analysis of an Multiprocessor-Based Intelligent Robotic Workcell

Simon C.L. Poon

This thesis describes a phase of development to the final goal - integrated workcell controller in the CIC. Several typical tasks in an industrial robotic workcell have been implemented in the controller and are described in this thesis.

The objective of the workcell is to determine the location/orientation of 2D components lying on a conveyor and to direct a manipulator to synchronize the end-effector and a moving workpiece for material handling or on-line assembly. The control mechanism of the workcell is governed by a transputer-based multiprocessor hierarchical controller. This controller executes the tasks of primary image analysis, path generation and overall position feedback. The primary image analysis takes the image data from a stationary camera at the upstream of a conveyor and determines its location and orientaion. The path generation task determines the motion of the end-effector so that the end-effector may be able to intercept and then synchronise accurately with the workpiece. Two path generation control schemes - the linear path and the elliptical path are described in this thesis. The linear path provides a minimium rendez-vous time and the elliptical path reduces the transient acceleration. An overall position feedback task is incorporated in the controller so that the synchronisational offset can be compensated in realtime operation.

ACKNOWLEDGEMENTS

The author would like to thank all the colleagues of the Centre for Industrial Control for their constant support during the time of development of this project. I am also deeply indebted to my supervisor, Dr. R.M.H. Cheng for his invaluable guidance and encouragement, especially during the difficult hours of this project

I would like to acknowledge my wife Mythu and my parents, to whom this thesis is dedicated, for the continued support and encouragement they have provided me through these years of study.

Simon C.L. Poon

TABLE OF CONTENTS

Page

Nomenclature. x

List of Figures xiv

List of Tables. xx

PART I : BACKGROUND AND OBJECTIVES. 1

Chapter 1 : INTRODUCTION. 1

1.1 BACKGROUND AND SYSTEM DESCRIPTIONS 1

1.2 OBJECTIVES AND SCOPE 5

1.3 THESIS OUTLINE 8

Chapter 2 : TASKS OF THE INTELLIGENT ROBOTIC WORKCELL 9

2.1 PRIMARY IMAGE PROCESSING 9

2.1.1 Primary Image Acquisition System 9

2.1.1.1 review 9

2.1.1.2 description of the image acquisition test bench 11

2.1.2 Image Processing 15

2.1.2.1 image processing review. 15

2.1.2.2 image processing methodology 18

2.1.2.3 speed performance of the image acquisition. 22

2.2 GOBAL SPATIAL PATH GENERATION. 23

2.3 ROBOT CONTROL. 26

page

2.4 OVERALL FEEDBACK 29

2.4.1 Motivation 29

2.4.2 Description of the Joint Data Acquisition. . . 29

PART II : CONTROLLER CONFIGURATION AND TASK IMPLEMENTATION. . 32

Chapter 3 : MULTIPROCESSOR HIERARCHICAL CONTROLLER SYSTEM HARDWARE AND SOFTWARE 32

3.1 GOAL 32

3.2 DESIGN CRITERION 33

3.3 DEVELOPMENT BACKGROUND OF THE HIERARCHICAL CONTROLLER 35

3.4 REVIEW OF COMMON CONTROLLER STRUCTURES 38

3.5 HARDWARE AND SOFTWARE OF THE HIERARCHICAL CONTROLLER

3.5.1 Assumptions. 46

3.5.2 Controller Architecture. 46

3.5.3 Controller Hardware. 47

3.5.3.1 processing element.. 47

3.5.3.2 memory 49

3.5.3.3 transputer module. 49

3.5.3.4 network. 50

3.5.3.5 motherboard. 52

3.5.3.6 transputer I/O.. 53

3.5.4 Controller Software. 56

3.5.4.1 compiler 56

3.5.4.2 communication drivers. 57

page

3.5.5 Task Implementation. 63

3.5.5.1 task distribution. 63

3.5.5.2 task communication and scheduling. . . 65

3.5.6 Performance. 68

3.6 SUMMARY. 69

Chapter 4 : PATH GENERATION TASK. 71

4.1 INTRODUCTION 71

4.2 DESIGN OF ON-LINE MOTION CONTROLLERS OF ROBOTIC MANIPULATORS 71

4.3 PATH CONTROL 75

4.3.1 "Minimium Time" Discrete Control Model 76

4.3.1.1 interception phase model 80

4.3.1.2 synchronisation phase model. 84

4.3.2 "Reduced Torque" Discrete Control Model. . . . 87

4.3.2.1 interception phase model 87

4.3.2.1.1 elliptical integral linearisation. 95

4.3.2.1.2 elliptical path position commands 102

4.3.2.2 transition and synchronisation phase model. 108

4.4 PERFORMANCE. 113

4.4.1 Productive Space Comparsion. 113

4.4.2 Torque Comparsion. 122

page

4.4.3 Synchronisation Performance. 129

4.4.3.1 simulation technique 131

4.4.3.2 MT model synchronisation performance . 133

4.4.3.3 RT model synchronisation performance . 134

4.5 SUMMARY. 138

Chapter 5 : OUTERLOOP FEEDBACK TASK 140

5.1 INTRODUCTION 140

5.2 CONTROL MODEL. 142

5.2.1 Overall Feedback Control Loop. 142

5.2.2 Overall Feedback Controller Algorithm. 143

5.3 EXPERIMENTAL RESULTS 146

5.3.1 Controller Tuning. 146

5.3.2 Performance of the MT Model. 149

5.3.3 Performance of the RT Model. 150

5.4 SUMMARY. 155

PART III : CONCLUSION 156

Chapter 6 : CONCLUSION AND SUGGESTIONS FOR FUTURE WORK. . . . 156

REFERENCES. 159

Appendix. 170

Appendix A : Register Address and Flowchart of the Idetix Camera Software Driver 171

Appendix B : Joint Feedback Data Acquisition Circuitry. . . . 180

Appendix C : Comparison of Common Multiprocessor Systems. . . 189

Appendix D : Limiting Speed Ratios between Robot and Workpiece of MT Model. 192

Appendix E : Numerical Solution of the Elliptical Integral. . 195

Page

Appendix F : Numerical Differentiation to Evaluate Joint Velocities and Accelerations 201

Appendix G : Newton-Euler Inverse Dynamics Formulations . . . 203

Appendix H : Forward Kinematics of the PUMA 560 Robot 206

Appendix I : Incremental Image Processing 210

Appendix J : Architecture of an Intelligent Robotic Workcell for Synchronisation Control 241

Appendix K : Auxiliary Camera Transeceiver/Strobe Light Circuit. 253

NOMENCLATURE

A_s	silhoutte area per frame
a	major axis of the elliptical path
a_{rx}	acceleration of the effector in x-direction
b	minor axis of the elliptical path
d_{m1}	distance of the interception point measuring from the camera station (MT model)
d_{m2}	distance of the interception point measuring from the camera station (RT model)
d_{s1}	maximum allowable workspace of the RT model
d_{s2}	maximum allowable workspace for the MT model
d_t	distance travel during the transition phrase
$E(\alpha, \varphi)$	value of the elliptical integral with the parameters α and φ
e_x	error betwwen the robot effector and the workpiece
e_{xo}	error control output of the overall feedback controller
i, j, k	unit vectors of displacement
i_ϕ, k_ϕ	unit vectors of the inclined plane of the interception phase in RT model
K_c	proportional gain
k	number of iteration
k_i	number of iterations required to reach the interception point
k_s	number of iterations required to reach the synchronisation phrase
s	specified distance between the true intercepion point to the virtual interception point
L_1	workspace upstream limit
L_e	workspace downstream limit
l_f	frame lenght (mm)

$m[k]$	theoretical number of sample intervals to accomplish interception phase
m_i, c_i	parameters of the linearised function for segment i
n	total number of iterations required for the robot to move from the home position to the (virtual) interception position
n_f	number of frames
n_s	maximum number of segments
n_{tr}	number of iteration at the end of transition phase
$r[k]$	incremental positional command at k iterations (components: $r_x[k]$, $r_y[k]$, $r_z[k]$)
$S[k]$	lenght of remaining elliptical path from kth iteration to nth iteration (components s_x, s_y, s_z)
s	specified distance between the true intercepion point to the virtual interception point
p_k	desired robot location at time k
$p_r[k]$	accumulative positional commandend-effector, relative to starting position of robot, at time k (components : $p_{rx}[k]$, $p_{ry}[k]$, $p_{rz}[k]$)
$P_r[k]$	robot location at time k
$P_w[k]$	measured position of workpiece at time k (components: $p_{wx}[k]$, $p_{wy}[k]$, $p_{wz}[k]$)
Q_k	best-fit polynomial for path segment
$q_r[k]$	position vector of the robot effector on the elliptical path w.r.t. the imaginary centre
$\Delta q_r[k]$	change of the vector $q_r[k]$
$\dot{\Delta q}_r[k]$	rate of change of the vector $q_r[k]$
T_d	derivative time
T_i	integral time
t_e	time overhead to compare the error for each segment and decide which segments should be used

t_{co}	time to compare the elliptical integral errors
t_{i1}	intercept time of the MT model
t_{i2}	intercept time of the RT model
t_{io}	time overhead for the hierarchical controller to handshake with the robot controller
t_{it}	image transmission overhead
t_{if}	time left for image processing
t_{p1}	productive time of the MT model
t_{p2}	productive time of the RT model
t_{pf}	average time to process one frame
t_{seg}	time overhead to calculate the path data of one segment
t_t	time spent in the transition phase
t_{tr}	communication overhead for one transputer talking with another transputer
$v_c[k]$	measured speed of workpiece & conveyor at time k
$v_r[k]$	velocity of robot at time k (components : $v_{rx}[k]$, $v_{ry}[k]$, $v_{rz}[k]$)
V_r	prescribed magnitude of robot velocity (constant)
v_{rx}	robot velocity in x-direction
V_z	designated robot velocity in z-direction
X_c	X centroid position
X_j	position X of j pixel
X_s	X optical scale
$x_{act}[k]$	actual robot x-cartesian position i
x_o, y_o, z_o	starting position of robot end-effector, relative t, stationary camera
x_m, y_m, z_m	interception point

x_{mo}, y_{mo}, z_{mo}	computed distance from starting position to interception point
x_{moi}	computed x-distance from the starting position to interception point by using i segment
Y_c	Y centroid position
Y_i	position Y of i pixel
Y_s	Y optical scale
Z_d	vertical position of centroid of workpiece
$\beta[k]$	angle between the vector $p_r[k]$ and the axis of the elliptical path in xy-plane
ΔA_{ij}	the area of the pixel at position i, j of a frame
λ	number of iteration periods before robot command is accomplished
Σ	$\sum_{i=1}^{k-1}$
τ_a, τ_d	implementation delays of motion controller
τ_s	sampling delay of motion controller
θ	workpiece orientation

LIST OF FIGURES

		Page
Figure 1.1	Schematics of the Experiemental Workcell	3
Figure 1.2a	Photograph of the Test Rig	4
Figure 1.2b	Photograph of the Test Rig	4
Figure 1.3	Development and Scope of the Robotic Workcell Research	7
Figure 2.1	Binary Image	10
Figure 2.2a	First Frame is Being Taken	12
Figure 2.2b	Second Frame is Being Taken.	13
Figure 2.3	Centroidal Profile	16
Figure 2.4	Bribiesca's Shape Number	17
Figure 2.5	Dimensionless Parameters	20
Figure 2.6	Procedures of Primary Image Processing Implementation	21
Figure 2.7	Spatial Path in Realtime Operation	25
Figure 2.8	PUMA 560 Robot	26
Figure 2.9	Robot Controller Structure	28
Figure 2.10	Joint Data Acquisition Schematics.	31
Figure 3.1	Tightly-Coupled Multiprocessor System.	42
Figure 3.2	Harmony Multiprocessor System.	42
Figure 3.3	NYMPH Multiprocessor System.	43
Figure 3.4	Butner Multiprocessor System	43
Figure 3.5	Loosely-Coupled Multiprocessor System.	44
Figure 3.6	Tightly-Loosely Coupled Multiprocessor System. .	45
Figure 3.7	Architecture of Multi-Device Workcell Controller	48
Figure 3.8	B404-3 Transputer Module	50

		Page
Figure 3.9a	Cube Network	50
Figure 3.9b	Hypercube Network.	51
Figure 3.9c	Torus Network.	51
Figure 3.10	Inmos B008 Transputer Mother Board	54
Figure 3.11	Hierarchical Structure of Multi-Transputer Network.	55
Figure 3.12a	Transputer Host Dependent I/O.	58
Figure 3.12b	Transputer Local I/O	59
Figure 3.13	Parallel C Configurer.	61
Figure 3.14	Operation of Interrupt Driven File Server. . . .	64
Figure 3.15	Task Distribution of the Multiprocessor Workcell Controller	66
Figure 3.16	Schedule and Communications of the Tasks	67
Figure 4.1	Time Delay Between Position Command and Actual Position of Robot	74
Figure 4.2	Path Control Scheme Developed by Tom Montor. . .	77
Figure 4.3	Path of the "Minimum Time" Model	78
Figure 4.4	Path of the "Reduced Torque" Model	79
Figure 4.5	Relationship of Salient Position Vectors During the Intercept Phase (MT Model).	81
Figure 4.6	Relationship of Salient Position Vectors During the Synchronisation Phase (MT Model). . .	86
Figure 4.7	Centre and Vertices of the Elliptical Segment. .	90
Figure 4.8	Path of the Reduced Torque Model with Virtual Interception Point	91
Figure 4.9	Linearinsation of the Elliptical Integral. . . .	97

Page

Figure 4.10 Position Vectors and Commands of the RT Model. . 107

Figure 4.11 Transition Phase of the RT Model 109

Figure 4.12 Linear Velocity Profile of the Transition Phase. 110

Figure 4.13 Upstream and Downstream Limit of the workspace . 114

Figure 4.14a Productive Space of MT Model (v_c=100mm/sec v_r=200mm/sec). 117

Figure 4.14b Productive Space of RT Model (v_c=100mm/sec v_r=200mm/sec s=10mm). 117

Figure 4.14c Productive Space of RT Model (v_c=100mm/sec v_r=200mm/sec s=25mm). 118

Figure 4.14d Productive Space of RT Model (v_c=100mm/sec v_r=200mm/sec s=50mm). 118

Figure 4.15a Productive Space of MT Model (v_c=150mm/sec v_r=200mm/sec). 119

Figure 4.15b Productive Space of RT Model (v_c=150mm/sec v_r=200mm/sec s=10mm). 119

Figure 4.15c Productive Space of RT Model (v_c=150mm/sec v_r=200mm/sec s=25mm). 120

Figure 4.15d Productive Space of RT Model (v_c=150mm/sec v_r=200mm/sec s=50mm). 120

Figure 4.16a Velocity Profile (MT Model). 124

Figure 4.16b Velocity Profile (RT Model). 124

Figure 4.17a Joint #1 Computed Torque Profile (MT Model). . . 125

Figure 4.17b Joint #1 Computed Torque Profile (RT Model s = 0mm) 125

Page

Figure 4.17c Joint #1 Computed Torque Profile
(RT Model s = 5mm) 126

Figure 4.17d Joint #1 Computed Torque Profile
(RT Model s = 10mm). 126

Figure 4.17e Joint #1 Computed Torque Profile
(RT Model s = 15mm). 127

Figure 4.17f Joint #1 Computed Torque Profile
(RT Model s = 25mm). 127

Figure 4.17g Joint #1 Computed Torque Profile
(RT Model s = 50mm). 128

Figure 4.18 MT Synchronisation Performance (Delays Ignored). 130

Figure 4.19 Path Simulation Model. 132

Figure 4.20 Simulation and Experiment of the Synchronisation Performance of the MT Model. 135

Figure 4.21 Experiements with Disturbances on Conveyor Speed 136

Figure 4.22 Simulation and Experiment of the Synchronisation Performance of the RT Model. 137

Figure 4.23 RT Model Synchronisation Error 138

Figure 5.1a Synchronisation Performance without Overall Feedback 141

Figure 5.1b Synchronisation Error. 141

Figure 5.2 Schematic of the Overall Feedback Controller . . 144

Figure 5.3a Synchronisation Response of the MT Model
(K_c=0.5 T_i=100000 T_d=0.0). 151

Page

Figure 5.3b Synchronisation Response of the MT Model (K_c=1.0 T_i=100000 T_d=0.0). 152

Figure 5.3c Synchronisation Response of the MT Model (K_c=1.25 T_i=100000 T_d=0.0). 152

Figure 5.4 Synchronisation Response of the MT Model (K_c=0.5625 T_i=0.3333 T_d=0.0). 153

Figure 5.5 Synchronisation Response of the MT Model (K_c=0.75 T_i=100000 T_d=0.05) 153

Figure 5.6 Synchronisation Response of the MT Model (K_c=0.6 T_i=0.4 T_d=0.0). 154

Figure 5.7 Synchronisation Response of the MT Model (K_c=0.6 T_i=0.4 T_d=0.0). 154

Figure A.1 Software Driver to Activate the Idextix Camera to Take Image. 175

Figure A.2 Software Driver to Reset the Camera Controller. . 176

Figure A.3 Software Driver to Download Parameters to the Camera Controller 177

Figure A.4 Software to Check Buffer. 178

Figure A.5 Software to Initialise the DMA Channel

Figure B.1 Pulse Conditioning Circuit. 179

Figure B.2 Direction Decoding Circuit. 184

Figure B.3a Pulse Train of A and B in One Direction 185

Figure B.3b Pulse Train of A and B in Opposite Direction 185

Figure B.4 Counting Circuit. 186

Figure B.5 Address Decoding Circuit. 188

		Page
Figure C.1	Comparison of Common Multiprocessor Systems . . .	191
Figure D.1	Conditions for Existence of Real-time Solution.	194
Figure E.1	Ellipse with a Major Axis 'a' and Minor Axis 'b'	197
Figure E.2	$E(\alpha,\varphi)$, α=constant	197
Figure E.3	$E(\alpha,\varphi)$, φ=constant	198
Figure I.1	Incremental Image Acquisition System Schematic.	215
Figure I.2a	Physical Critical Points (Two Straight Lines). . .	219
Figure I.2b	Virtual Critical Points (Two Straight Lines) . . .	219
Figure I.3	Physical Critical Point (Two Polynomial Curves). .	222
Figure I.4	Physical Critical Point (A Straight Line & a Polynomial Curve).	223
Figure I.5	Coordinate Systems of the Incremental Image processing Technique	225
Figure I.6	Auxiliary Camera Mounting	228
Figure I.7	Resultant Offset Vector r_e for non-drifted Workpiece.	229
Figure I.8	Resultant Offset Vector r'_e for drifted Workpiece.	230
Figure I.9	Original Workpiece	231
Figure I.10	Drifted Workpiece.	232
Figure I.11	Auxiliary Camera Taking the Secondary Image. . . .	233
Figure K.1	Transmitter Circuit for the Camera Buffer.	257
Figure K.2	Receiver Circuit for the Camera Buffer	258
Figure K.3	Flash Control Circuit.	259

LIST OF TABLES

page

Table 2.1 Table of the Matching Process Timings. 23

Table 2.2 Joint Resolution 30

Table 4.1 Error of the Linearisation of the Elliptical Integral 97

Table 4.2 Productive Space Comparison of the MT and RT Models. 121

Table 4.3 Torque Transient Comparison. 123

Table A.1 Idetix Camera Control Parameters 173

Table A.2 Idetix Camera - Command Registers and Data Set . . . 173

Table A.3 Idetix Camera - Status Registers 173

Table B.1 Joint Pulse Train Indicate Positive Motion. 187

Table B.2 Resolution of the Joint Counter for PUMA 560 Robot. . 187

Table E.1 Values of the Elliptical Integral 199

Table I.1 Experimental Results of the Incremental Image Processing (Position Drifts) 238

Table I.2 Experimental Results of the Incremental Image Processing (Orientation Drifts). 239

CHAPTER 1

INTRODUCTION

1.1 BACKGROUND AND SYSTEM DESCRIPTION

A robotic workcell is often used to reduce the production time for many manufacturing activities. In order to adapt the changing environment, e.g. the velocity and position of the moving workpiece or orientation of a randomly despatched workpiece, most of the workcells [9,15,29,36,65,102] usually consist of a vision system to detect the position, orientation or other features of a workpiece. However, for the automated operation to be economically attractive, the features must be computed quickly at a moderate system cost and operating cost. The trend towards high resolution cameras with specialised computer and electronics interface usually price such systems beyond what most small and medium industries can afford. Therefore, a robotic workcell [21] with a low cost vision system is developed in the Centre for Industrial Control.

The developed workcell[67] is based on two objectives:

> "Sufficient intelligence must be provided in order to identify and determine the position and orientation of parts randomly placed on a flat surface. The information must be gathered real-time, perferably by non-contact means.
>
> Position and velocity synchronisation must be achieved between an end-effector and a part randomly situated on a conveyor line and subject to velocity distubances."

The first development stage of the CIC-workcell[68] mainly consists of six components. These are :

(1) a workpiece detector

(2) a stationary (MicronEye) camera

(3) a PUMA 560 robot and its controller

(4) a conveyor belt

(5) workpiece position/velocity tracking circuit

(6) a hierarchical controller (IBM-XT)

Fig.1.1 shows the schematic of the system test rig and fig.1.2 shows the photographs of the system. The workpiece detector consists of a pair of infra-red LEDs (1) installing at either sides of the conveyor belt to generate an interrupt signal to activate the overhead stationary camera (2) for image analysis. The position and velocity of the moving workpiece are monitored by a tracking circuitry(5). The stationary camera with resolution 128x64 pixels/frame is installed at upstream of the conveyor (4) to detect the random arrival of a workpiece. An image of a workpiece is gathered by concatenating several frames. The hierarchical controller (6) analyses the image and extracts the identity, position and the orientation of the workpiece. Situated downstream of the conveyor is an industrial robot (3). After the image analysis has been finished, the Real Time Path Control (RTPC) mode of the robot is then activated. The hierarchical controller works out the trajectory commands and gives the commands to the robot controller through serial communication. The end-effector is then directed to a position immediately above the travelling workpiece (interception phase) and is eventually led to a stage of position/velocity synchronisation (synchronistaion phase) with the moving workpiece, taking possible speed variations in conveyor speed

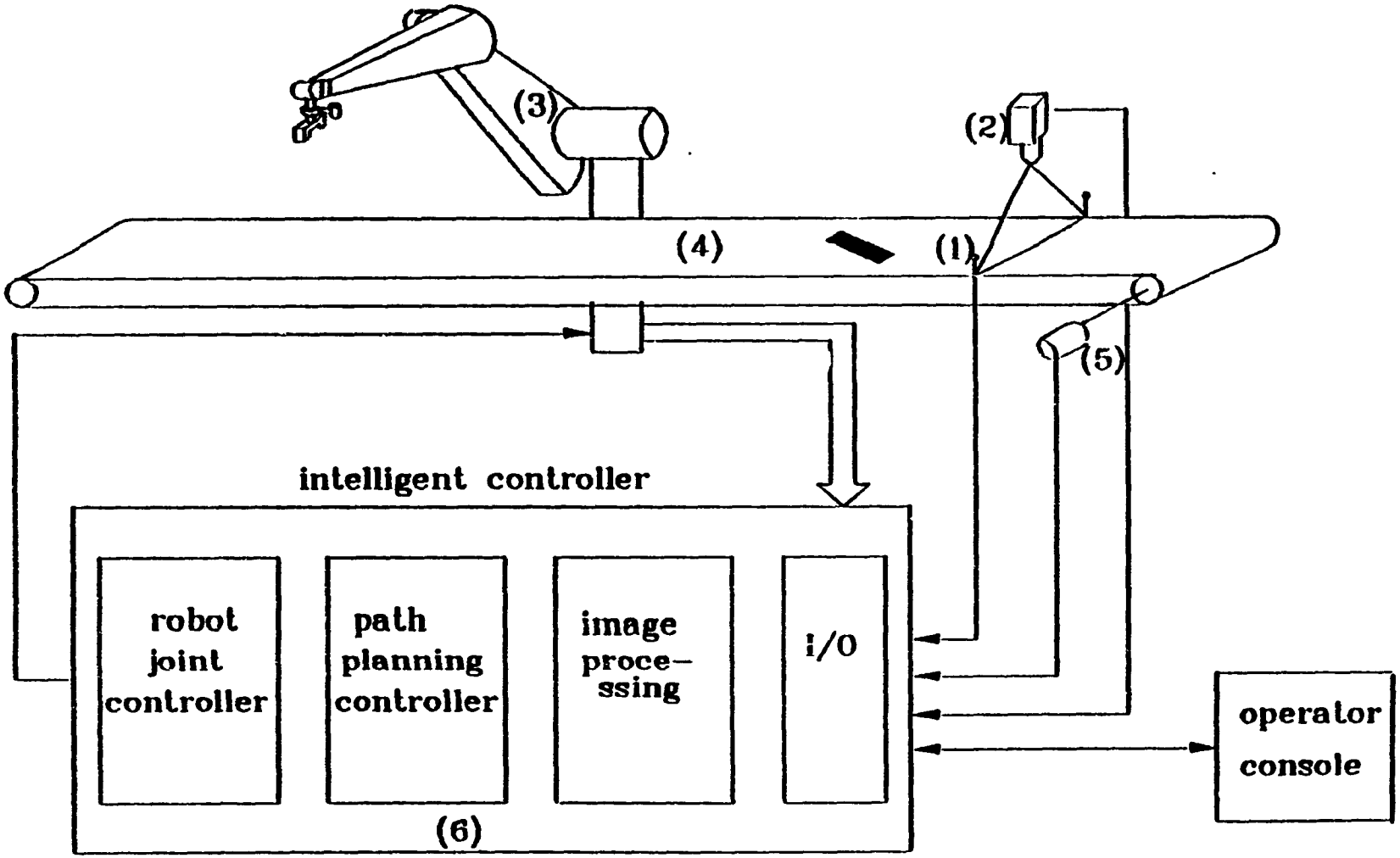


Fig.1.1 Schematics of the Experimental Workcell

Fig. 1.2a Photograph of the Test Rig

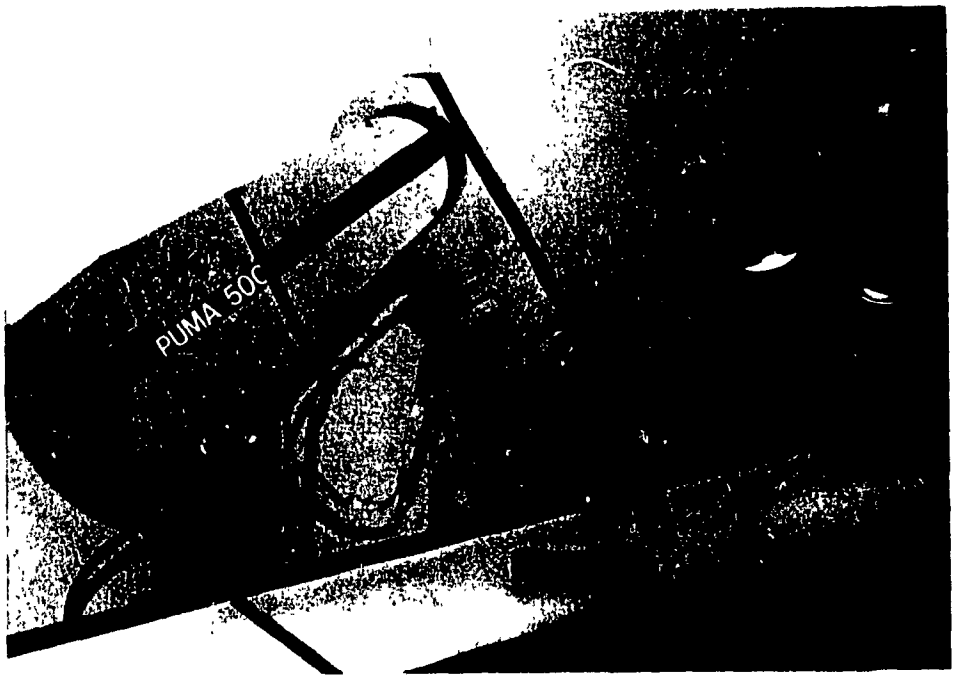


Fig. 1.2b Photograph of the Test Rig

into consideration. Thus it is possible to perform productive operations while the workpiece is continually travelling.

1.2 OBJECTIVES AND SCOPE

During the past few years, there has been a growing demands for the various applications of robotic workcell[33,47,52,78,101]. More mechanical devices, e.g. robots, control system, indexing/rotary tables and sensory devices, e.g. camera, force transducer, are required to be incorporated in workcells to carry out specific automation tasks in manufacturing, assembly or materials handling. Usually, the devices have their own dedicated controllers. As a result, a robotic workcell ends up with several controllers, different hardware interface and different programming environments. Therefore, the CIC final goal is to develop a relative low cost and flexible multiprocessor workcell controller which integrates all the dedicated controllers into one compact unit. The advantages associated with the concept of a single integrated controller as distinct from the sprawling architecture are :

1. it eliminates unnecessary duplication of hardware and software, offering a simplier system and greater economy;
2. it offers a uniformity in hardware and software (host and multiprocessor), eliminating differences in communicating protocols, especially for different robots or other devices;
3. it provides a modularity in design, expandable (upwards and downwards,) and adjustable to system needs at any time;
4. the internal inter-device communication means faster responses;
5. it eliminates high level control languages (e.g. VAL II on PUMAs) which do not allow users to change or modify the control algorithms.

They are bound to be ponderous since they are general-purposed while workcells are specific;

6. it offers more efficient system maintenance;

7. it provides a powerful tool for implementing robotic control research;

8. the approach is novel, and has tremendous potential for industrial application.

Various robotic workcell applications result different design criteria. In case of the industrial material handling workcell, the robot should be able to carry higher payload. An on-line assembly workcell may require larger productive space. Therefore, the additional objective of this thesis is to formulate two spatial path control models. The first model provides a minimum rendezvous time for the purpose of giving the maximum productive space. The second model is to smoothen the global path such that the robot can sustain higher payload running at the same robot velocity.

On-line material handling and assembly operations require good synchronisation performance to minmise the error between the end-effector and the moving workpiece. In order to achieve better synchronisation performance, another objective is to incorporate an overall position feedback inside the hierarchical controller.

Finally, higher adaptive capacity may be achieved if additional intelligence is incorporated in the workcell to correct the error of the primary image analysis and the unpredictable relative drifts between the moving workpiece and the conveyor. A 'incremental image processing' technique is then developed for such purpose. However, this technique is beyond the scope of this thesis. The reader is suggested to refer details of the developed technique in Appendix I.

The development and scope of the robotic workcell research in CIC is summarised in fig.1.3.

Task	first reported in 1986	intermediate development	latest development
primary image processing (section 2.1.2.2)	/	/	/
simple spatial path planning (section 4.3)	/	/	/
multiprocessor controller configuration (ch.3)	x	/ (IBM-AT plus 68000)	/ (IBM-AT plus transputers)
spatial path planning - *Minimum Time Model* (section 4.3.1)	x	/	/
spatial path planning - *Reduced Torque Model* (section 4.3.2)	x	x	/
overall position feedback to improve synchronisation performance (ch.5)	x	x	/
incremental image processing (appendix i)	x	x	/

x - not included
/ - included

fig.1.3 Development and Scope of the Robotic Workcell Research

1.3 THESIS OUTLINES

Chapter 2 outlines the implemented tasks of the developing workcell. Chapter 3 describes the history of the development, architecture and communications of the multiprocessor controller. Chapter 4 describes two spatial path control models and they are also compared in term of productive space, transient torque and synchronisation performance. Chapter 5 presents the formulation and tuning of the overall feedback controller. Finally, the conclusion in chapter 6 summaries the project and the recommendations for future works are included.

Chapter 2

TASKS OF THE INTELLIGENT ROBOTIC WORKCELL

The test rig consists of the following tasks to form an experimental robotic workcell :

- primary image processing
- spatial path generation
- robot motion control
- overall feedback

2.1 PRIMARY IMAGE PROCESSING

The primary image processing task analyses the workpiece image to obtain the position and orientation. Section 2.1.1 describes the hardware of the vision system. Section 2.1.2 describes the image processing techniques. A brief review of some common image processing techniques are discussed in section 2.1.2.1. The implemented image processing technique is outlined in section 2.1.2.2 and finally the performance of the implemented technique is presented in section 2.1.2.3.

2.1.1 Primary Image Acquisition System

2.1.1.1 review

In the past, linear array scanner and vidicon camera are used as the industrial vision input devices[15,59,65,78,102]. However, their short duration life ,fragility, image time lag and image distortion reduce its attraction. In addition, extra procedures are required to convert the analog signals into digital data for

processing.

Lately, the solid state charge-coupled device (CCD) camera become more popular in the industrial application [47,52,105]. There are in the form of optic RAM with various resolution 128 x 128, 512 x 512 , 2048 x 2048 elements, etc.. The elements are arranged in rows and columns. Each element represents an analog value as the color of a point on an object. When light falls on the RAM, the charge of the elements varies with the light intensities in each location. These kinds of camera are sensitive over a wide spectral range. They have little lag time so that it is more suitable for moving line application[15] and have more positional accuracy.

For certain industrial applications [21,30,106] which particularly concentrate on simple geometric properties such as position, orientation. Binary image(fig.2.1) can be sufficient and economical instead of using the grey level image. The binary image, each pixel is represented by 1 or 0, is more easier to be stored and processed.

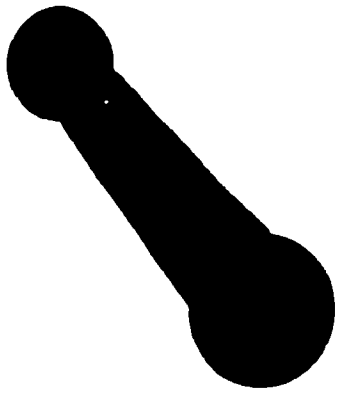


Fig.2.1 Binary Image

2.1.1.2 description of the image acquisition test bench

During the initial stage of development, a MicronEye binary camera[64] with resolution 128x64 was used as the image acquisition unit. The corresponding window on the conveyor surface is 265mm x 46mm(fig.2.2a). The camera is aligned with the conveyor such that the '64' pixel is along the longitudal direction of the conveyor. Once the workpiece detector is activated, the camera is triggered to take the first frame of the workpiece. The complete image of the workpiece is composed of a number of concetenated frames taken (fig.2.2b) by the primary camera. The primary camera is activated through hardware interrupt for each 46 mm conveyor belt displacement. With a typical 8 msec exposure and a 67 msec transmission time per frame, the total image acquisition overhead amounts to approximately t_{it} = 75 msec.[71]. Therefore, the time left t_{lf} for image processing at a conveyor speed with v_c is

$$t_{lf} = l_f / v_c - t_{it}$$

For a typical conveyor speed 150 mm/sec.,

$$t_{lf} = 46 / 150 - 0.075$$
$$= 0.0232 \text{ sec.}$$

Presently, the MicronEye camera is replaced by a new Idetix camera with a DMA electronic interface board and the IBM-PC is also replaced by a 12 MHz IBM-AT. The software of the camera driver and image processing is written in C language to replace the previous Fortran and Assembly codes.

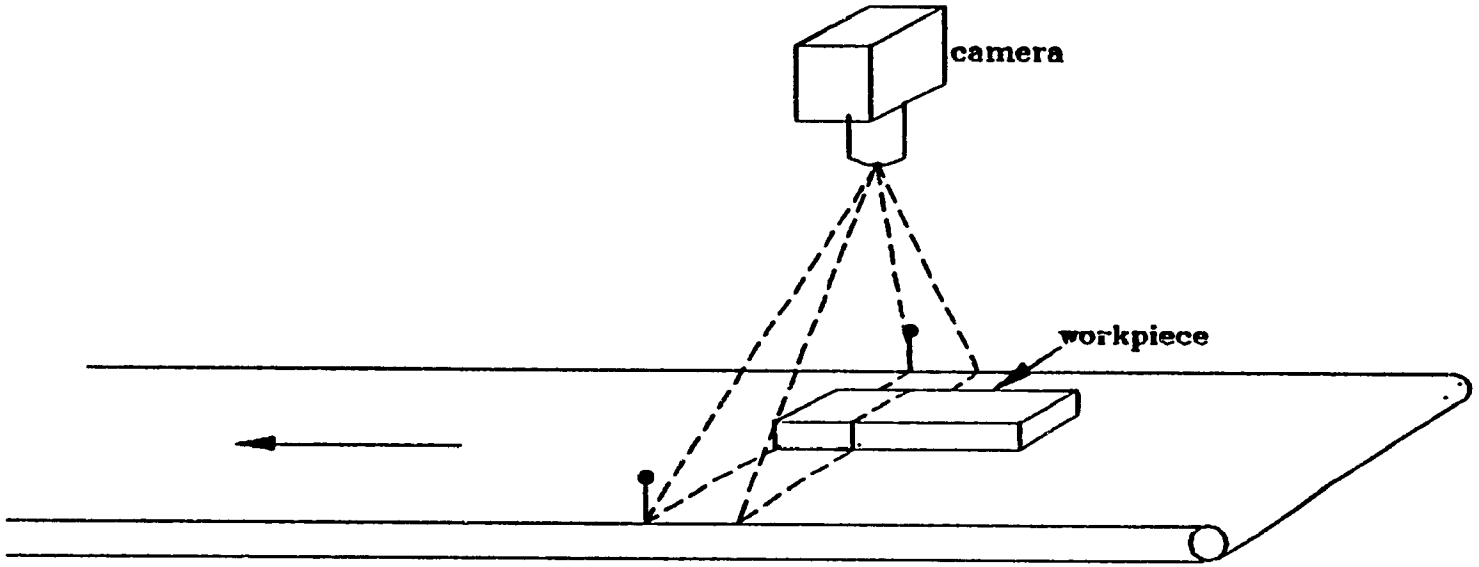


Fig.2.2a First Frame is Being Taken

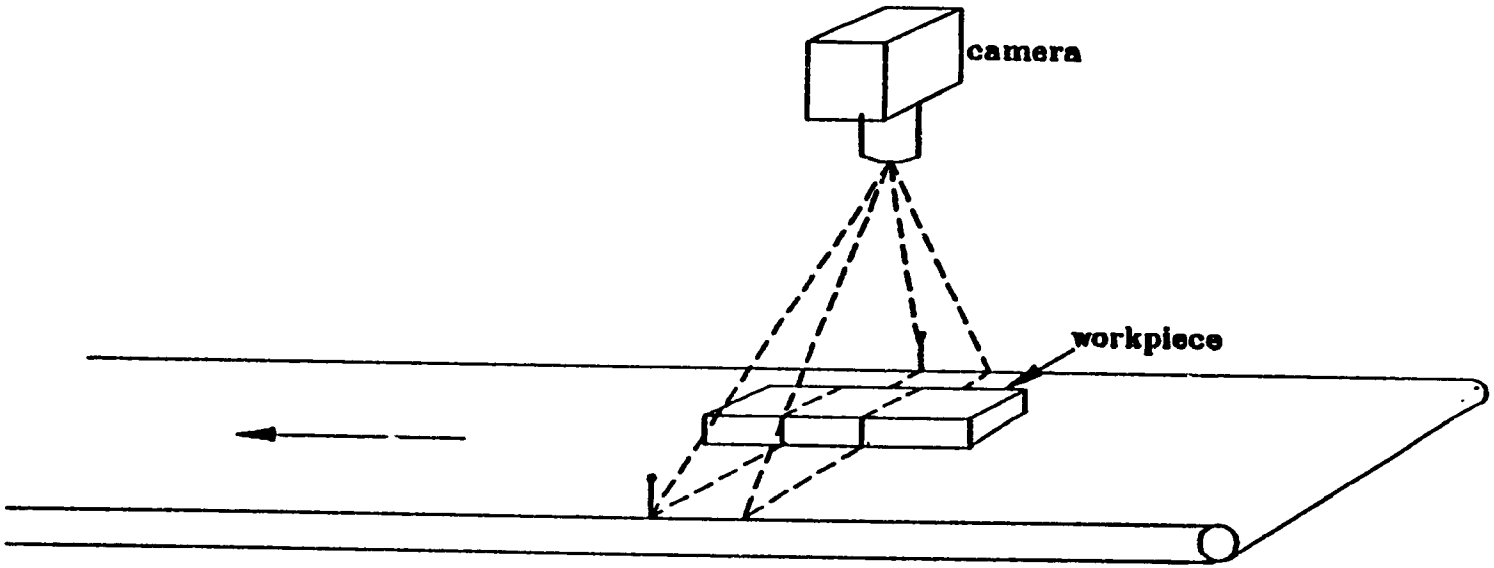


Fig.2.2b Second Frame is Being Taken

The Idetix controller board[63] communicates with the IBM host through the I/O slots on the mother board. Pixel information is transferred from the controller to IBM memory via Direct Memory Access. Pixel data from the image sensor is sampled and shifted into an eight bit shift register. DMAREQ sets the selected DRQ high and latches the contents of the shift register. The controller clock is inhibited while DRQ is high. When the IBM host acknowledges the DRQ with a /DACK, a /IOR will occur generating a /IORW and reading the latched pixel data onto the data bus. This data will be stored automatically into the IBM host memory by the DMA controller. All the transferred operation is actually scheduled by a 63701 microcontroller on the Idetix interface board. The communication between the IBM host and the microcontroller is via an 18 byte Command and Data register set. Commands and data are sent to the microcontroller by writing each byte sequentially to a single address. This address can be between 200H and 2F0H on 10 byte boundaries. The default setting 260H is used in this research project. The register address and the flowchart of the Idetix camera driver are shown in the Appendix A. The DMA operation provides a faster transmission rate compared to the MicronEye camera. The image transmission overhead is only 6 msec per frame. Including the 8 msec exposure time, it only requires 14 msec overhead. The reduction of the image transmission overhead gives more time for the image processing. For a typical conveyor speed 150 mm/sec.,

$$t_{lf} = 46 / 150 - 0.014$$

$$= 0.0293 \text{ sec.}$$

Comparing to the MicronEye camera, there are 61 msec more for the image processing analysis per frame.

2.1.2 Image Processing

2.1.2.1 image processing review

Image processing has been developed 15 years and there are many different image processing techniques for different applications, such as part identification, quality inspection, motion guidance, etc.. However, the following brief review is concentrated on the following objectives :

- . identification of the 2D industrial workpieces
- . detection of the workpiece position
- . detection of the workpiece orientation

The earlier method to identify the industrial parts is the centroidal profile technique[31]. The method calculates the centroid of the workpiece first. Then, it calculates all the distances between the critical points and the centroid(fig.2.3) to form a curve of a centroidal profile. The identification of the workpiece can be obtained by matching with the pre-stored profiles. This method requires to store a number of series and compare each of them in real-time. Better precision of the centroidal requires considerable increase of computing power. Bribiesca[13] used the method of finding out different shape numbers by calculating the length of the perimeter of a 'discrete shape' closely corresponding to the curve. The determination of the shape is to compare the similarity between the regions of silhoutte(fig.2.4). There are two difficulties in using this method. Firstly, it has to identify the order of the shape in order to have an unique shape number. Secondly, the inclined basic rectangle is not easy to be figured out. Some researchers[106] used

the invariant moment to recognise workpiece recognition. Besides the invariant moment, Hoard[38] uses additional features, e.g. area, inertia moments, the minimum, maximium and average radii and the length of the perimeter for part recognition. Nagao[75] developed a trail and error process to use a variable size slit to obtain the characteristic features of a workpiece for identification. This method is very flexible. On the other hand, it requires several trials and errors to get the optimium slit position.

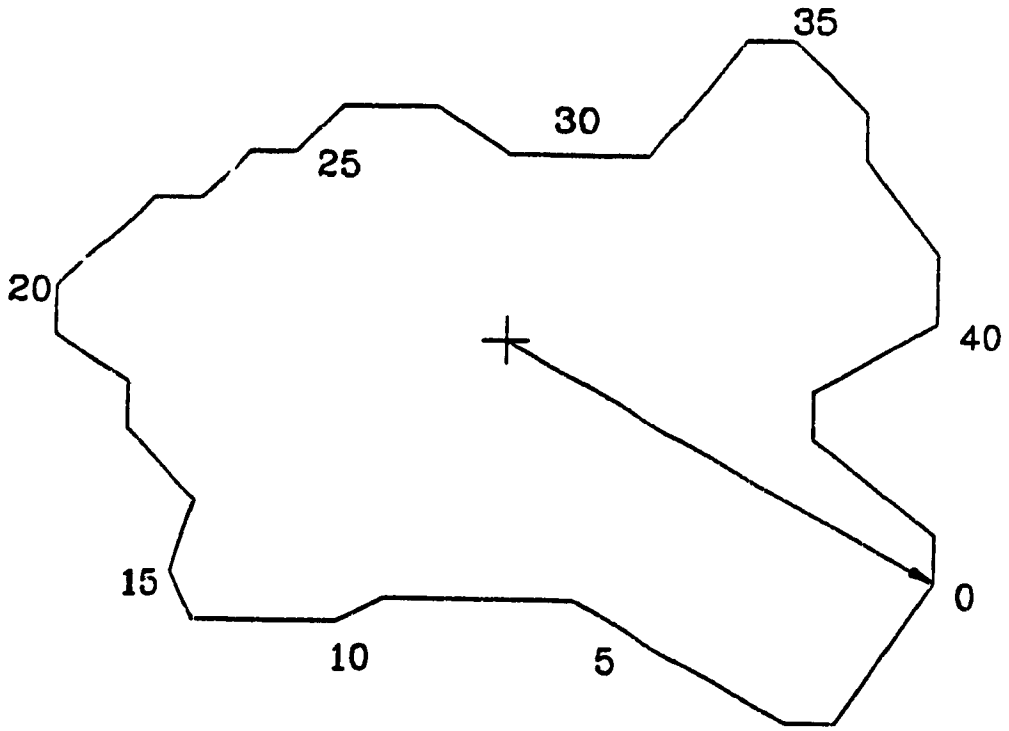


Fig.2.3 Centroidal Profile

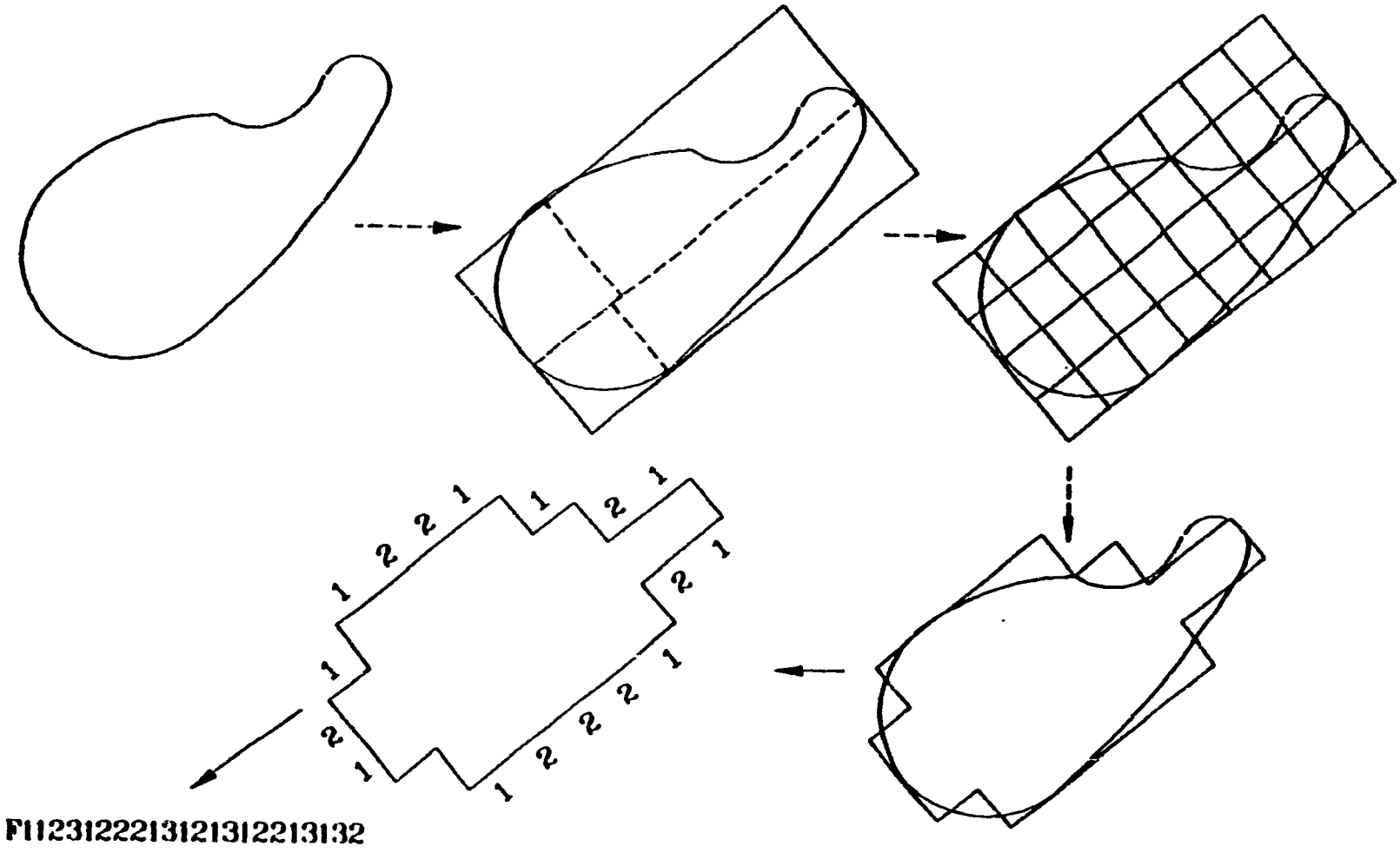


Fig.2.4 Bribiesca's Shape Number

To find the orientation, Armbruster[8] used the chain code to establish a profile series by quantising the curvature of the workpiece. A number of series with a corresponding reference orientation are stored offline. The orientation can be obtained by comparing the chain code of the random oriented workpiece. Besides the chain code, Kammenos[44] generates a sequence of radial distances from the centroid by comparing each pre-stored sequence with the corresponding orientation. Driel[29] uses the method of the 2nd moment to find out the orientation. The above methods require intense computing power to match the correct profile series.

Concerning the position calculation, some researchers calculate the position of an industrial parts by analysing the features[62, 85]. The more simple approach[4,44,103] is to take the centroid of the workpiece as the position.

2.1.2.2 image processing methodology

The primary image processing algorithm was first devised in their original form by the authors of [19] in 1980 and [20] in 1983. The algorithm was then implemented in real-time control by Tom Montor[22] on the robotic workcell.

As discussed in the reference[20], the position , i.e. the centroid, of a workpiece may be calculated by taking the 1st moment about a mutually prependicular reference lines. This method is simple, fast and particularly good to a binary environment. The centroid can be described by the following equations :

$$X_c = \frac{\sum_{i=M}^{N} \sum_{j=P}^{R} X_j \, \Delta A_{ij}}{\sum_{i=M}^{N} \sum_{j=P}^{R} \Delta A_{ij}} \cdot X_s$$

$$Y_c = \frac{\sum_{j=P}^{R} \sum_{i=M}^{N} Y_i \, \Delta A_{ij}}{\sum_{i=M}^{N} \sum_{j=P}^{R} \Delta A_{ij}} \cdot Y_s$$

where X_c is the x-centroid position

Y_c is the y-centroid position

X_j is the position x of j pixel

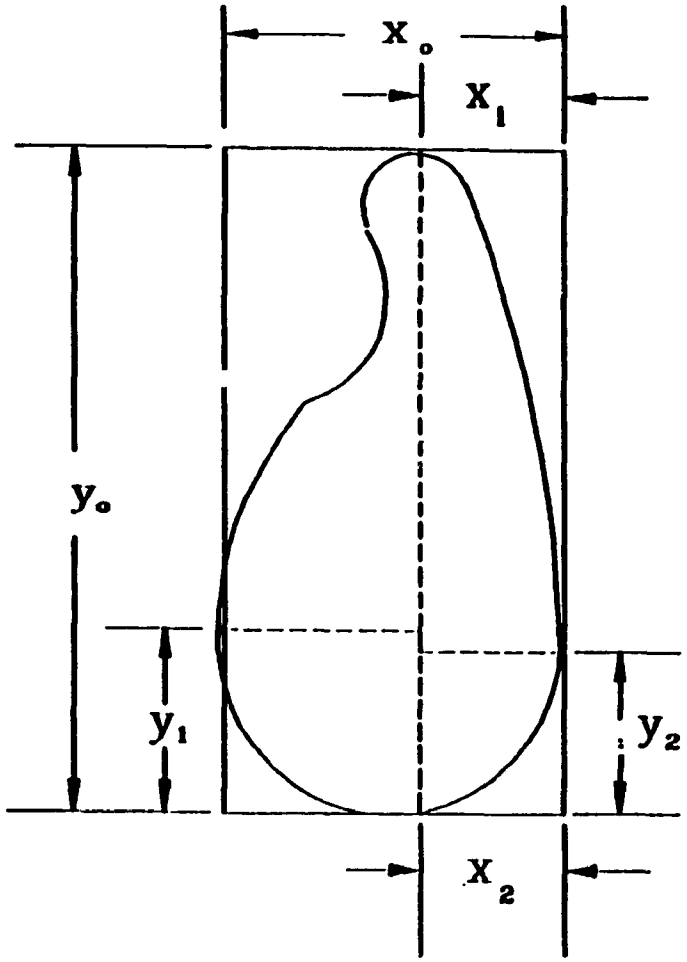
Y_i is the position y of i pixel

ΔA_{ij} is the area of the pixel at position (i, j) of a frame

X_s is the x-optical scale (mm/pixel)

Y_s is the y-optical scale (mm/pixel)

The procedures to determine the orientation of a workpiece consists of two phases - set up phase and operational phase[69]. The set-up phase generates a look-up table composed of a selected number of geometric parameters extracted from the silhouette. Fig.2.5 shows how the dimensionless geometric parameters of a workpiece at angle θ is determined in a circumscribing rectangle. With every incremental rotation of the profile, the non-dimensionless values are generated and inserted with the angle in which they belong. In the real-time operational phase, the angular rotation is recognised by matching the parameters extracted from the silhouette of the workpiece with the parameters generated in the set-up phase. The procedures of the image processing implementation shown in fig.2.6. The error of this method to obtain the orientation may be up to 4 to 6 degrees[70].



$$p_1 = \frac{x_o}{y_o}$$

$$p_2 = \frac{x_1}{x_o}$$

$$p_3 = \frac{y_1}{y_o}$$

$$p_4 = \frac{x_2}{x_o}$$

$$p_5 = \frac{y_2}{y_o}$$

Fig.2.5 Dimensionless Parameters

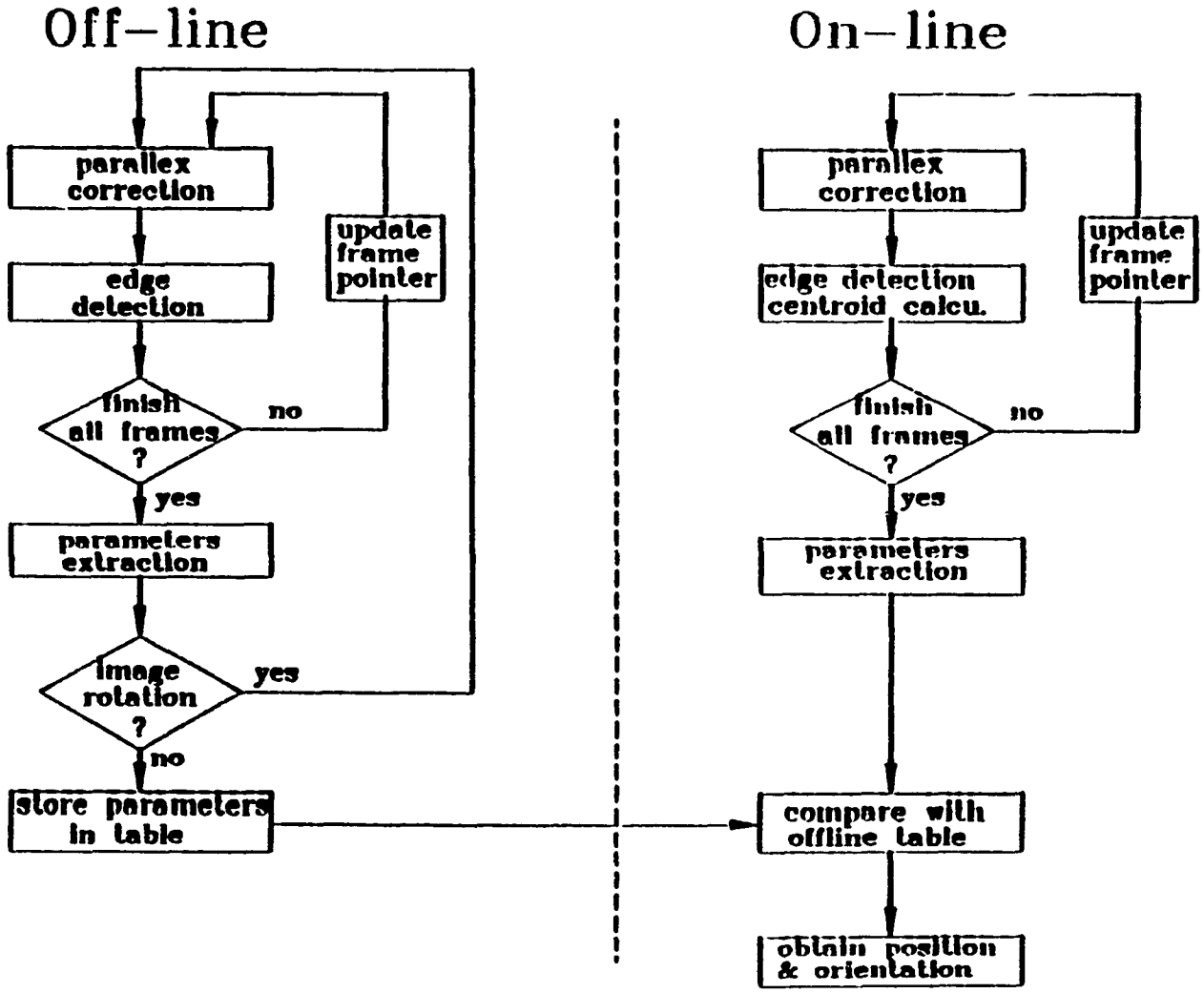


Fig.2.6 Procedures of Primary Image Processing Implementation

2.1.2.3 speed performance of the image acquisition

The performance of the vision system has been improved. There are several factors affecting the speed of the image acquisition. They are :

. time of image transmissions rate

. time of image processing per frame

. time of matching the parameters

The image transmission rate has been discussed in section 2.1.1.2. The image processing algorithm has been modified to process a unit of byte instead of a unit of bit. Thus, the time for the image parallax correction, centroid calculation and parameter extraction has been speeded up. The average processing speed per frame is represented by the following equation :

$$t_{pf} = 86 * A_s + 26.2$$

where t_{pf} is the average processing speed in msec

A_s is the percentage of the silhoutte area per frame

The image processing time of the new vision module is about 10 to 15 times faster compared to the MicronEye system. There are two factors affecting the time taken in the process of parameters matching - 1) the number of parameters in the pre-stored table; 2) the number of the extracted parameters in real-time. The following table shows how those factors affect the timings,

TABLE 2.1 : TABLE OF THE MATCHING PROCESS TIMINGS

resolution of pre-stored parameters for 180°	resolution (degrees)	number of real-time parameters 1	2	3	4
18	10	1 msec	1 msec	1 msec	2 msec
36	5	6 msec	8 msec	12 msec	15 msec
360	1	40msec	70msec	102msec	130msec

The above table shows the proportional relationship of the resolution of the pre-stored parameters and the number of the real-time parameters. For matching 4 parameters, it is shown that it only takes about 15 msec. to match a table with 5° resolution.

2.2 GOBAL SPATIAL PATH GENERATION

The cycle of the real-time operation has three sequential steps :

. workpiece detection

. image processing

. productive action

The workpiece detection and the image processing are briefly explained in the previous sections. At the time of productive action, the robot directs its end-effector onto the position of the moving workpiece to perform the material handling, on-line assembly, etc.. Directing the robot end-effector onto the moving workpiece has two phases (fig.2.7) :

. Interception Phase (A to B)

. Synchronisation Phase (B to C)

Interception Phase - With the robot assuming some position downstreams along the conveyor, the end-effector is directed to travel upstreams to meet an on-coming workpiece, making a preliminary match with the predetermined feature which is located at the identified workpiece position.

Synchronisation Phase - Subsequent to a successful stages it is required to develop an accurate matching in position, velocity and orientation between the end-effector and the workpiece before meaningful operation is carried out. The challenge lies in the fact that in practice, disturbances will vary the velocity and position of the workpiece along the direction of the conveyor. To a less significant extent, relative displacements between the workpiece and the conveyor are also possible.

The discrete control models of the paths will be described in chapter 4 in details. It takes into full account the servo time delay element , and disturbances which adversely modifies the velocity and position of the workpiece and the conveyor. Two path control models have been formulated and implemented. The first control model, called the *minimum time model*, takes the shortest geometrical path to intercept the moving workpiece. The objective is to provide larger productive space. The second control model, called the *reduced torque model*, takes an elliptical spatial path to intercept the moving workpiece. The objective is to reduce the acceleration of the robot end-effector.

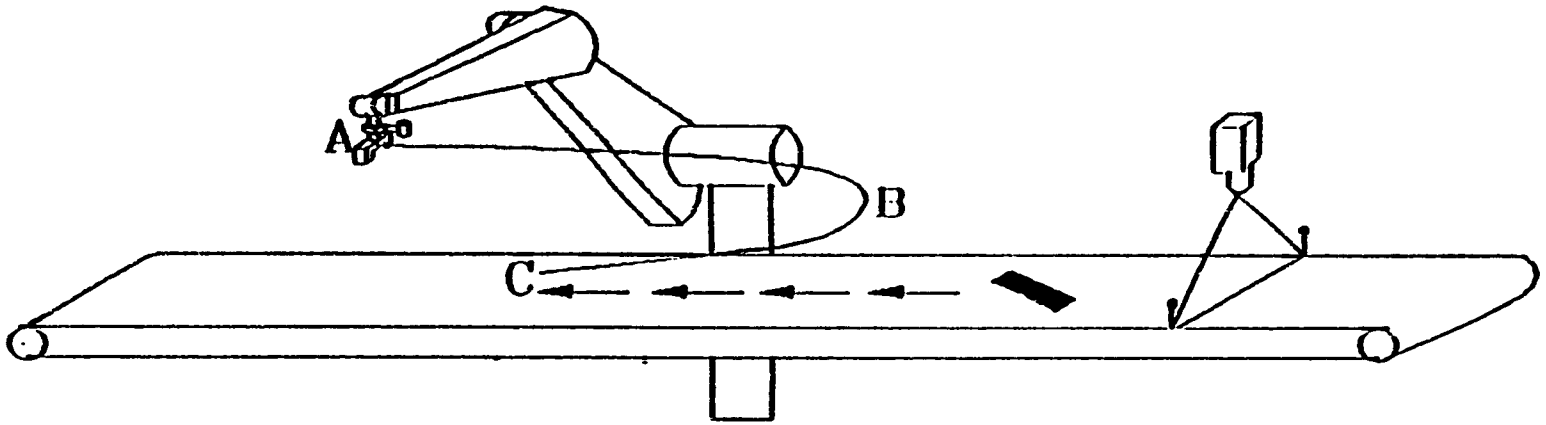


Fig.2.7 Spatial Path in Realtime Operation

2.3 ROBOT CONTROL

A 6-dof PUMA 560 robot(fig.2.8) is used in this research project. The robot comes with a vendor-provided motion controller and is controlled by using the proprietory robot language VAL-II. The motion controller is limited to be accessed and modified by the users.

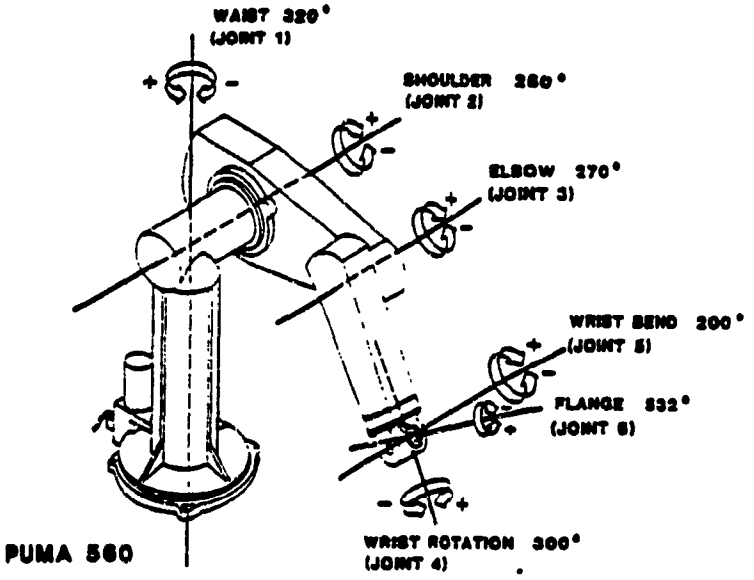


Fig.2.8 PUMA 560 Robot

The motion controller structure shown in fig.2.9 has a master CPU LSI-11. It accesses the peripherals, e.g. terminal, floppy disk, teach pendant, etc. through the Q-bus and the DLV-11J interface. The LSI-11 executes the VAL program and performs the kinematics calculations[77]. It also connects six joint servo controller boards through the DRV-11 interface. Each of them has a 6503 microprocessor as the central processing element. These independent servo boards receive the position command from the LSI-II for every 28 msec. The 6503 microprocessor generates a segment of trajectory for every 28 ms.

Then, it sends signals to the DAC and power amplifier to drive the joint motor. The joint servo controller board updates the joint current to the power amplifier for every 0.876 msec[77] within the 28 ms period.

The path of the robot is usually directed by the user application VAL-II programs which have been written offline. All the positions and orientations have been prescribed and this method is only suitable for the predictable applications. However, for those applications with sensory input, e.g. workpiece position, workpiece velocity which involve some sort of uncertainty and adaptive capability require the modification of the robot path in real-time (real-time path control). This can be achieved by using the ALTER port of the robot controller. The real-time path of the robot is dictated by the cartesian input [Δx, Δy, Δz, Δo, Δa, Δt] to the ALTER port. The end-effector position will be updated by the summation of the incremental cartesian commands.

A hierarchical controller is used to communicate with the ALTER port of the robot motion controller for every 28 msec. With respect to the workpiece orientation and currect position/velocity, it generates incremental position commands for every 28 msec. The trajectory generation, forward kinematics, inverse kinematics and joint motion controls will be the responsibility of the motion controller. It should be noted that the scope of this thesis is by and large independent of the design and operation of the motion controller. The methodlogies developed in this thesis are also not limited to a particular robot motion controller.

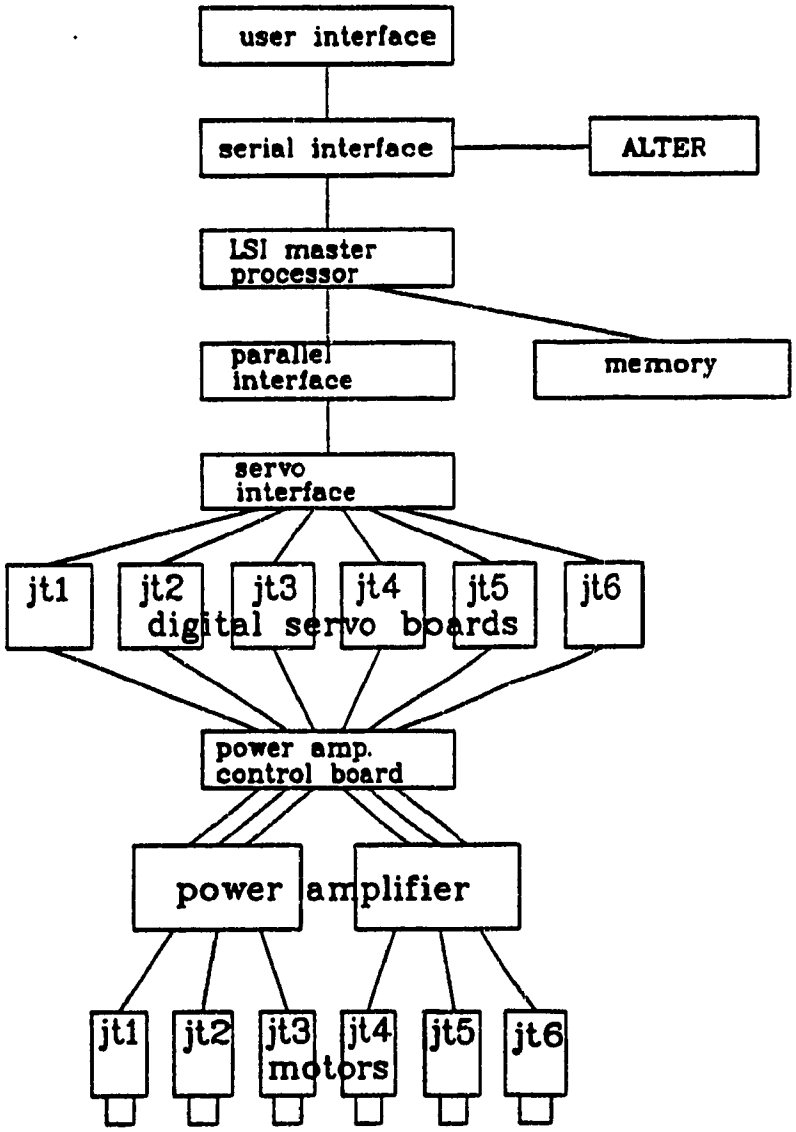


Fig.2.9 Robot Controller Structure

2.4 OVERLOOP FEEDBACK

2.4.1 Motivation

The open-loop spatial path control models described in section 2.2 is not adequate to achieve good synchronisation performance. There are several possible reasons :

. The robot motion controller is limited to be accessed by the users. Thus, the time delay of the control loops inside the controller may not be exactly the same as investigated;

. Error are accumulated when computing the inverse/forward kinematics inside the motion controller;

. Error are accumulated when the hierarchical controller computes the spatial path trajectory;

. Since the ALTER port only accepts integer data, truncated floating point errors may be accumlated;

. In the case of the *reduced torque model*, the approximation technique to generate the elliptical path may also cause certain errors.

Therefore, an overall feedback is incorporated to compare the current end-effector and the workpiece positions. The joint angle feedback are obtained by a joint data acquisition circuitry in the hierarchical controller. The controller executes the forward kinematics to calculate the actual cartesian coordinate. Any offset will be compensated by an overall PID feedback controller.

2.4.2 Description of the Joint Data Acquisition

Because of the limited accessibility of the robot motion controller, a data acquisition circuitry situated inside the hierarchical controller is used to keep track the joint positions. The

circuitry shown in fig.2.10 consists of four modules :

. signal conditioning circuit

. direction determination circuit[76]

. counting circuit

. address decoding

Each joint encoder has A and B inputs. The 6-dof robot requires totally 12 signal conditioning modules. The signal conditioning circuit creates a high impedance to isolate the encoder and the data acquisition. Another function is to filter the unnecessary noise due to inferences and disturbances.

There are six direction determination circuits. It detects the phase difference of the A and B encoder inputs to determine the direction of the motor movements.

There are six 16-bit up/down counters to monitor the joint positions. The resolutions of each joint are listed in Table 2.2. The results are adopted from the reference[76] and has been verified experimentally.

TABLE 2.2 : JOINT RESOLUTION

joint number	resolution (degrees/count)
1	0.0230
2	0.0167
3	0.0268
4	0.0189
5	0.0200
6	0.0188

The address decording circuit selects a particular counter to performs the READ or WRITE operations. Details of the joint data acquisition circuitry can be referred to Appendix B.

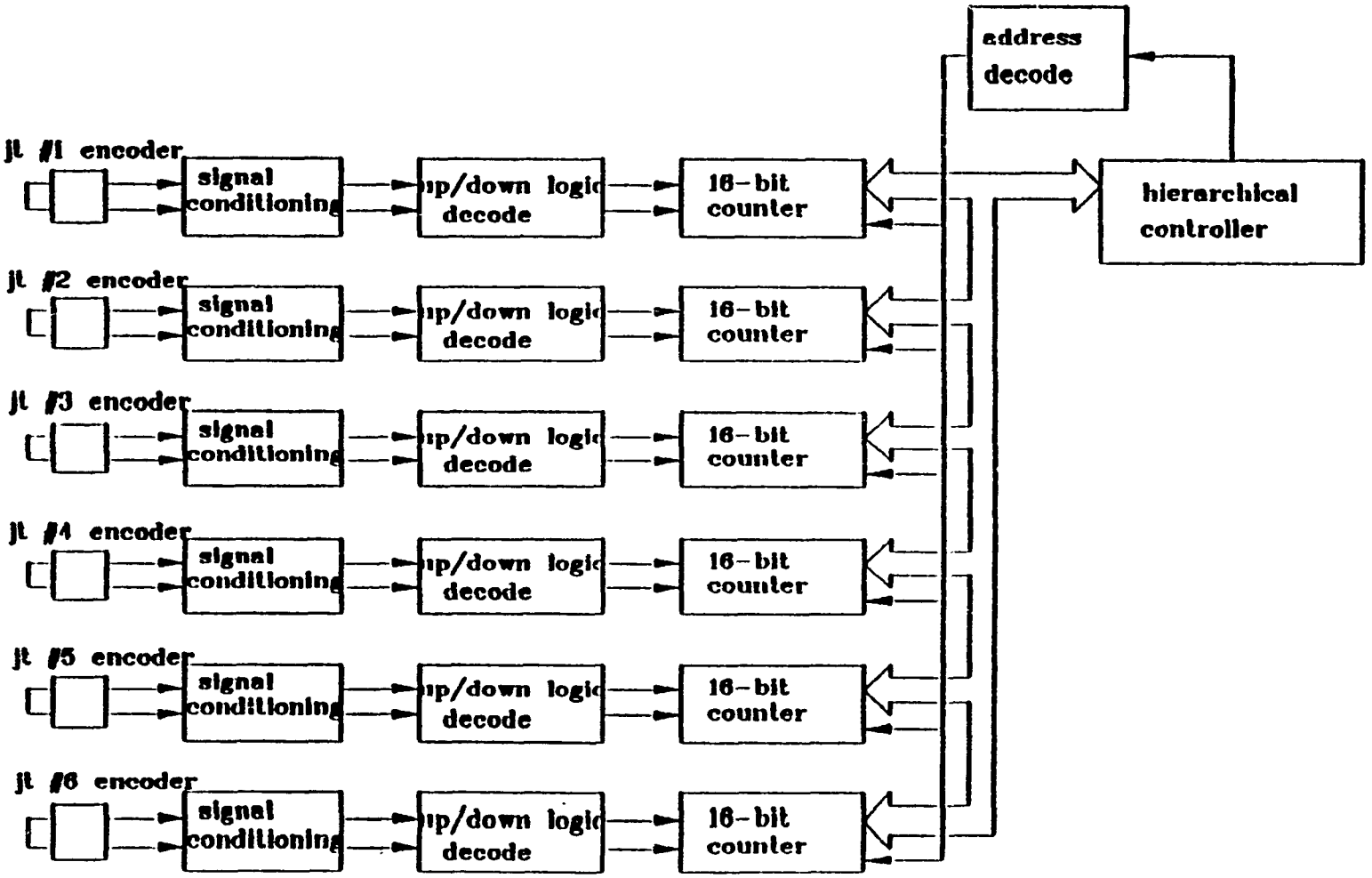


Fig.2.10 Joint Data Acquisition Schematics

Chapter 3

Multiprocessor Hierarchical Controller System Hardware and Software

This chapter discusses the necessities, requirements and development of a low cost and compact real-time workcell hierarchical controller. Its function includes the coordination and execution of the following tasks :

. image processing

. spatial path generation

. overall feedback

Section 3.2 outlines the design criterion of the controller. The background of the development is described in section 3.3. A review of some common controller structures is presented in section 3.4. The multiprocessor architecture is chosen to be the suitable configuration. Existing multiprocessor systems available in market have also been compared and evaluated in section 3.4. The transputer is chosen as the processing element of the controller. In section 3.5, the hardware and software of the transputer-based controller is described. The implemented structure of the controller is presented. Merits and demerits of the developing transputer based real-time workcell controller are then outlined. Finally, the performance of the implemented controller is described.

3.1 GOAL

With the pace of the growth of technology and sophisticated industrial application, it is increasingly necessary for a workcell controller to be able to control and monitor multiple devices in an

industrial robotic workcell. Therefore, the main objective is to design a flexible architectural structure for an integrated workcell, so that it can handle multiple tasks simultaneously. More importantly, the same flexibility can be extended to different tasks specifications, by suitably assembling 'device drivers' whenever they exist or develop new ones to add to the list of drivers. The concept of drivers is not a novelty in the present generation of microcomputers. A computer can handle a variety of I/O devices like printers, video adaptors by the provisions of software drivers. This concept is merely extended to provide drivers which may also include a set of collection of computing hardware. For example, a device can be a robot, force sensor, camera, or another robot, etc.

3.2 DESIGN CRITERION

The design criterion of an integrated workcell controller are as follows :

High Precision - High precision may not be so significant to the integer operations, such as the image analysis. However, the inaccurate floating point operations in the application of robot control or sensor input analysis may generate significant errors. If the data bandwidth is narrow, the errors may probably be accumulated in consecutive iterations. Higher accuracy in floating point calculations can be achieved with wider data bandwidth, e.g. 32 bits.

High Speed - Most of the real-time automated systems are required to have the input updated within a designated sampling time. In the case of the PUMA 560 robot, the master processor LSI-11 has to update the joint servo position commands for every 28 msec[77]. The servo digital

controllers give commands to the power amplifiers for every 0.9 msec to maintain a smooth trajectory. Therefore, the processing element inside the controller should be able to provide sufficient computational power for real-time calculations, such as image analysis, kinematic model, dynamic model and joint motion control algorithm.

Sensory Interactive - The system should be able to interface with sensors in order to detect any changes of the environment parameters. The system/sensors interface should be popular and available in the market.

Extensible - The high speed engines inside the controller should provide sufficient computational power for complex robot control algorithms, intensive image and sensor data analysis. However, it is impossible to predict the upper bound of the computational power since more and more complex algorithms are being developed for real-time control. Therefore, it is desirable to have a controller with extensible architecture which is easily to be upgraded in computational power.

Modularity - The system can be partitioned in modules so that different layers and tasks can be easily managed. Failure of one processor can be interchanged by another processor under the process of switching the software and I/O interface.

Parallelism - Tasks can be assigned to run in parallel to control different devices. The degree of parallelism should also be flexible

in terms of job level, task level or code level.

Task Interaction - Fast communication should be provided between the tasks. The task can be scheduled in parallel processors and the execution time is not slowed down by the communication overhead.

Uniformity - Uniformity of hardware and software makes the maintenance and installation easier. 'Uniformity of hardware' represents that the same kind of processing elements are used in all levels and 'uniformity of software' refers that the same kind of programming language is used in all levels of the system.

Software Environment - A high level language compiler should be provided for easier software development.

3.3 DEVELOPMENT BACKGROUND OF THE HIERARCHICAL CONTROLLER

This section traces the different phases of development of the intelligent robotic workcell at the Centre for Industrial Control. It describes the tasks that have been incrementally added to the controller at the various phases, leading to the long term objective of a hierarchical controller for multi-arm, multi-device workcells. Alongside with the ever increasingly demanding load on the computing facility which the controller must provide, the choice of hardware and software architecture and specific components takes on a process of evolution.

In 1986, the first configuration of the CONCIC workcell[21] was developed. It relies on an IBM-PC (running at 4.7 MHz) to serve as the hierarchical controller, communicating via 2 serial links with the

robot motion-controller provided by the vendor. Specifically, the CONCIC workcell scans a workpiece travelling along a conveyor system when it comes under a stationary video camera. The latter transmits the image data to the IBM-PC, which also carries out the necessary image processing to compute the position and orientation of the workpiece. The computer also charts out an appropriate path in the world coordinate to direct the robot to move "upstream" of the conveyor to meet the travelling workpiece. This rendezvous is actually carried out by the motion-controller of the robot, which receives the path-commands in packets, at time intervals of 28 ms as dictated by the protocol of the motion controller. Thus the architecture was in line with most other developments [15,47,52,101] in which the overall control assumes two linked modules. The performance of this first configuration was satisfactory for simple tasks of pick and place operations, with the 4.7 MHz INTEL 8088/8087 providing all the computing capability. This is primarily because of the efficient image processes algorithm used in the control software [20,22], partly because much of the codes were written in assembler. The latter requirement demands consideration effort in software maintenance and system enhancement.

The second phase of the CONCIC workcell came about as a decision to upgrade the performance of the synchronous control of the robot by providing an optimized path planning. This is an adaptive process, depending on the location of the robot (in all 6 DOF) at the time the workpiece is recognized by the camera. The workpiece position and orientation is of course another set of variables. This new specifications brought about a review of the control architecture, by incorporating microprocessors which are able to function in parallel

with the host computer, and which physically are accommodated within the host itself, sharing the I/O facility of the latter. The parallel processor chosen is the PC68K board, with a single MC68000, running at 10 MHz under its own operating system of OS9. At this time, most of the hierarchical control software was redeveloped, using the C-language, gradually phasing out the use of assembler. The result was an upgraded system performance, with the robotic path being optimized into two straight line segments taken minimum time for the rendezvous. The architecture upgrade is reported in reference [23], while the algorithm of the optimized path planning and its digital implementation is described in another paper [24].

The supplier of the processing board PC68K was never able to live up to their original specifications of (1) providing for multi-copy of the MC68000 within the same PC-bus, (2) interfacing with a floating point daughter board. Another strategy has to be chosen in order to implement an growing list of task demand on the workcell.

The third phase of the CONCIC development includes a further optimization of the robotic path to ensure smoother transition resulting in lesser demands of the accelerating capability of the joint motors of the robot. Put in another word, given the tasks of a workcell, online optimized path planning puts less demand on the motor rating at the specification stage of the robot. Analytical research leads to an elliptic path planning. In fact, the entire path between the starting position of the robot and its initial rendezvous position with the travelling workpiece is split into one elliptical segment and one parabolic segment, each in a separate plane that is normal to the plane in which the workpiece travels. At the same time, another refinement requires capturing the joint position feedback so as to

adjust the path planning in order to improve on the accuracy of rendezvous. While the motion controller is provided within position feedback of the joint motors for the proper functioning of the loop-controls, access to these feedback signals are generally denied to the user of the robot. Consequently, a joint data acquisition circuit is hooked up to the 6 joint motors (or more specifically their respective shaft encoders) in order to obtain the feedback signals necessary to carry out this refinement.

This leads us to a second and overall review of the control architecture and the system requirements, culminating in a compact controller which provides all necessary control for the robot motion as well as the workcell functions such as image processing, and path planning. In fact, an architecture emerges, which caters for multi-robot control in a single workcell.

3.4 REVIEW OF COMMON CONTROLLER STRUCTURES

Various controllers have been developed by many researchers to handle complex control algorithm in the past. General speaking, the common possibilities are as follows :

- multitasking uniprocessor
- special processors - Vector, Array
- multiprocessors

Multitasking Uniprocessor - It is a common sight that an uniprocessor system can execute several tasks simultaneously. The conventional approach is to use the technique of pipelining. After each tasks has been scheduled and prioritised in the system, each tasks may be executed with a slice of the CPU time. All these multitasking

processes are transparent to the operator. This approach is very common and has successfully implemented for certain period of time[35,50]. Although the uniprocessor can execute various tasks simultaneously, the switching of the CPU from one task to another task actually spend more time in saving and calling the environment. In addition, the system performance may be degraded when additional tasks are assigned to the system. Therefore, the actual system performance is difficult to evaluate in the beginning. Finally, the cost of an uniprocessor system to have such multitasking capability is not cost effective.

Special Processor - Specific processors[6,7,42,55,98] may also be used for the applications in the robotic control and image analysis. The speed of these processors are very fast since their hardware design are particularly for the specific numeric calculations. The speed of the DSP μPD77230 [98] can have 13.4 Mflops. However, the ALU of the DSP has been heavily pipelined and it cannot have full executing speed for random robotic algorithms[7]. Therefore, it can actually operate at about 6.5 Mflops. Some researchers[6,55] use array processor as the processing element which can be pushed above 10 Mflops. Unfortunately, array processor also relies the regularity of algorithms to achieve its maximum performance. Therefore, its uni-tasking characteristic and the downgrade performance become the bottleneck to these special processors in controlling real-time control multi-devices . Even if it is fast enough to handle the control of multiple devices in sequential order, such as the JIFFE[7] can deliver 20 Mflops, the tasks cannot be easily partitioned and the flexibility is limited for interchangeability or future expansion. The multi-device software all

in one processor is hard to debug and maintain. It is suggested that a controller with a versatile configuration would be invaluable. In the long run, Anderson[7] suggests that distributed processing seems to be more practical. Fast scalar processor will be gradually replaced by parallel processors.

Parallel Processor - An alternative choice is the multiprocessor configuration. This configuration consists of a number of parallel processors to execute individual tasks simultaneously. This concept has already been implemented in large systems for number of years. However, it has not been widely implemented in the microcomputer real-time control. Many researchers have developed various multiprocessor systems to carry out intensive computations in the application of real-time controls. There are also many multiprocessor systems available in market as shown in the Appendix C. The multiprocessor systems are compared on the basis of a performance index($/mip). Generally speaking, the multiprocessors architecture may be mainly classified into three categories[45] :

tightly-coupled -Tightly-coupled configuration (fig.3.1) is very common in the early days of parallel processors development [18,42,48,60,79,80,104]. It consists of a global memory to be accessed by all processors for communication via a single bus. Examples listed in the Appendix C are the PC4000, D64180LP, TMS320C25 , the Harmony and the DSI-780. The PC4000 and HD64180 have the highest performance index, however, that have no floating point operations. The other TMS320c25 with high performance index has no high level language. Its low level assembly ASM25 may make the development more difficult. From

the view point of capability, the Harmony system shown in fig.3.2 seems very eligible. However, it is not that cost effective. Other researchers also have developed some multiprocessor systems. Ozguner's system[80] consists of four Intel 8086 based computers which communicate with each others through multi-bus to control a hexpod walking machine. The data between processors are exchanged through sections of common memory. NYMPH[18] shown in fig.3.3 was developed as a multiprocessor system which composed of 3 layers. The first layer VAX connects the second layer Sun Station through a Ethernet link. Seven 32081 floating point coprocessors each with 32k RAM at the bottom layer communicate with each others. They are all supervised by the Sun station through a multi-bus. The I/O and data acquisition are also sent/received via the bus. This system is used to control the Stanford Arm which has three fingers with three degree of freedom each. If there are more processors added to this system, problem of bus contention may occur to reduce the efficiency of the inter-processor communication. Ish-Shalom[42] has a similiar multiprocessor structure to NYMPH. The first layer has an IBM mainframe. The second layer consists of an IBM PC(XT or AT) and the bottom layer is composes of several signal processors instead of the 32081 floating processor. Its performance is better than the NYMPH because each signal processor has its own local I/O. The I/O data transfer on bus is totally eliminated and thus there is less chance to have problem of bus contention. Butner[60] has an improved configuration shown in fig.3.4. Instead of having one bus, he separates the system into more number of levels and each level has its own bus for inter-processor communication. Butner's multiple bus system may be a solution to the problem of bus contention and to

efficiency of a multiprocessor network, however, it may not be feasible from the economical point of view.

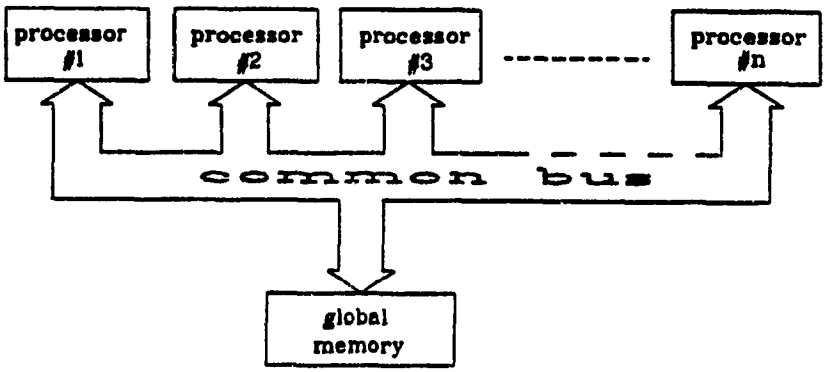


Fig.3.1 Tightly-Coupled Multiprocessor System

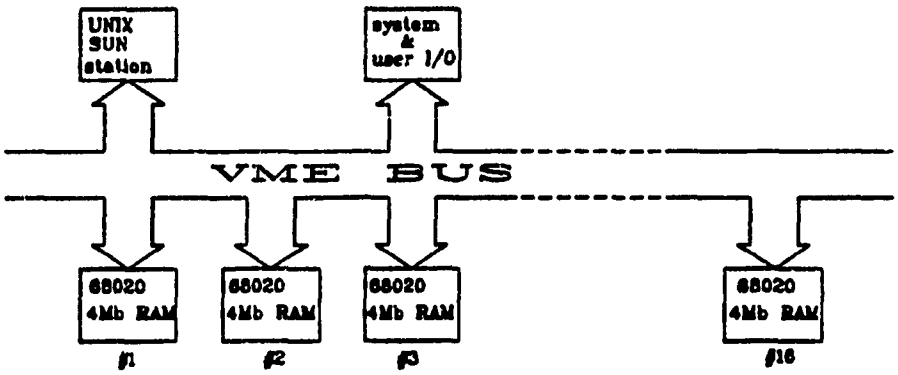


Fig.3.2 Harmony Multiprocessor System

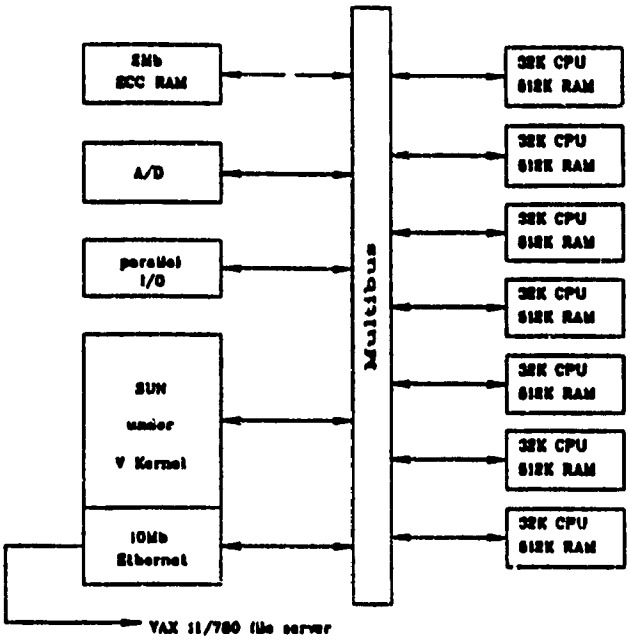


Fig.3.3 NYMPH Multiprocessor System

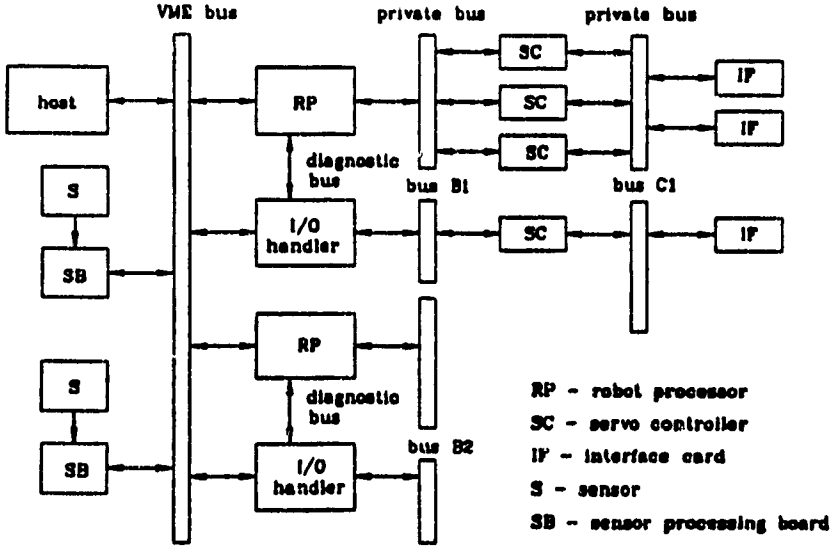


Fig.3.4 Butner Multiprocessor System

loosely coupled - An alternate architecture[45,107] is the distributed memory system shown in fig.3.5. A typical example is the transputer. Each processor is provided with its local memory and the data is shared by transmitting data in the form of message between processors. This configuration does not have the bottleneck of bus contention. It allows more number of processors in a system. However, the bottleneck usually are the communication overhead in transferring messages between processors. Therefore, this configuration may only be feasible if a system provides fast communication speed between processors.

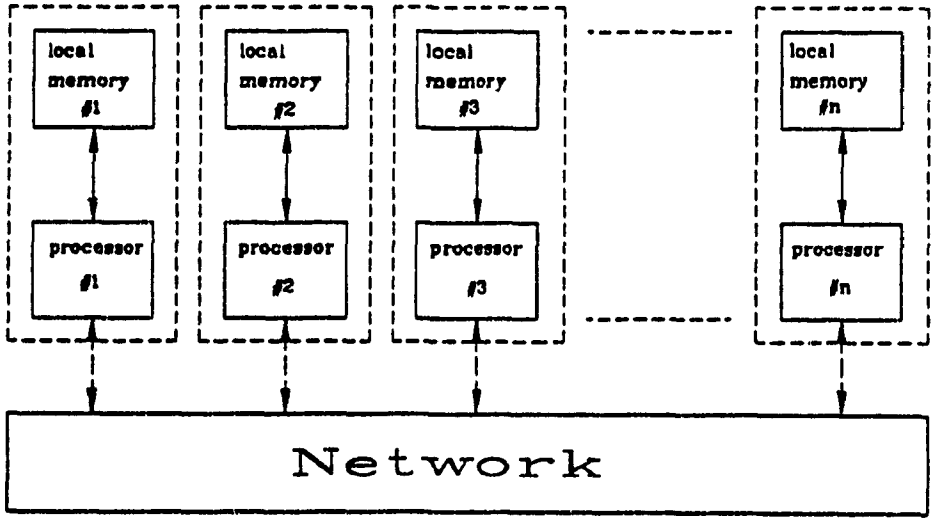


Fig.3.5 Loosely-Coupled Multiprocessor System

tightly-loosely coupled - Another configuration shown in fig.3.6 is the distributed memory multiprocessor system. The individual processor has its own local memory. The system also consists of a global memory and a network for the inter-processor communications. Any asynchronised inter-processor communication may be achieved by accessing the global memory. Other synchronised inter-processor communication may go through the message channels. It provides fast communication speed and eliminates the problem of bus contention. However, this configuration may not be cost effective because the architecture are more complicated than the formers.

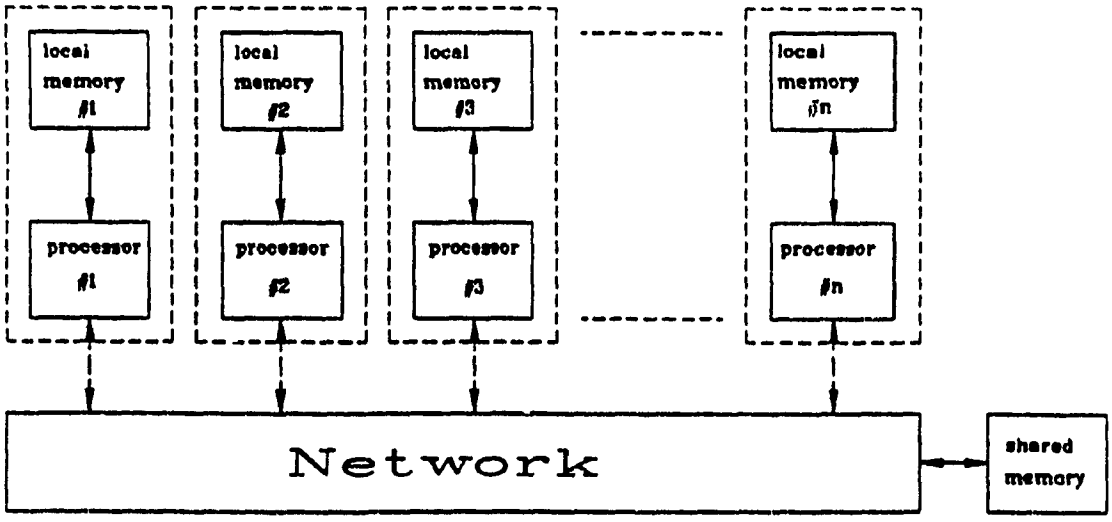


Fig.3.6 Tightly-Loosely Coupled Multiprocessor System

3.5 HARDWARE AND SOFTWARE OF THE HIERARCHICAL CONTROLLER

3.5.1 Assumptions

The discussions in section 3.4 provides the following directions for the selection and design of the workcell controller :

. The controller should be a multiprocessor system to provide substantial computional power.

. The architecture of the multiprocessor system should be a loosely-coupled system

3.5.2 Controller Architecture

A general architecture of a workcell controller for multi-devices will look like fig.3.7 which is the ultimate stage of development of the CONCIC program. It is shown schematically that the host accommodates a number of clusters of processing elements. The host can also have direct interface with its own set of devices, including the standard console devices. Some clusters are configured as device drivers and have I/O interfaces; others simply carrying computing task, and do not directly control any external device. However, the results of computing can be routed directly to other clusters as appropriate.

By carefully assigning clusters of small numbers of suitably interconnected processing element to look after the computing requirement of a device as well as communicate with the hierarchical control module(s), one achieves the kind of sophisticated device driver as stipulated above.

Thus for the CONCIC workcell, one may assign

1. 1 processing element for the primary image analysis;
2. 1 processing element for the path planning with elliptical

optimization;

3. 1 processing element for processing the joint feedback and the necessary compensation to the path planning

4. 2 or more processing elements for the joint control of the robot, depending on whether one employs the simple independent loop-servo controls via inverse kinematics (commonly implemented in the controller of most industrial robots, or take advantage of sophisticated control strategies involving inverse dynamics to computer joint torques and adaptive algorithms).

3.5.3 Controller Hardware

3.5.3.1 processing element

The processing element is an Inmos T800-20 transputer. The T800-20 transputer is a 20MHz 32-bit processor which consists of a memory, processor and communications system and a 64 bit floating point unit(FPU) via a 32-bit bus. The 32 bit processor average executes 10 million instruction per second. The FPU is able to perform floating point operation concurrently with the processor and it provides at a rate of 1.5 Mflops[40]. The fast computation speed is presently sufficient to run the individual task in each transputer.

There are six registers inside the transputer for execution of a sequential process. They are :

. The workspace pointer which points to an area of store where local variables are kept

. The instruction pointer to the next instruction to be executed

. The operand register which is used in the formation of instruction operands

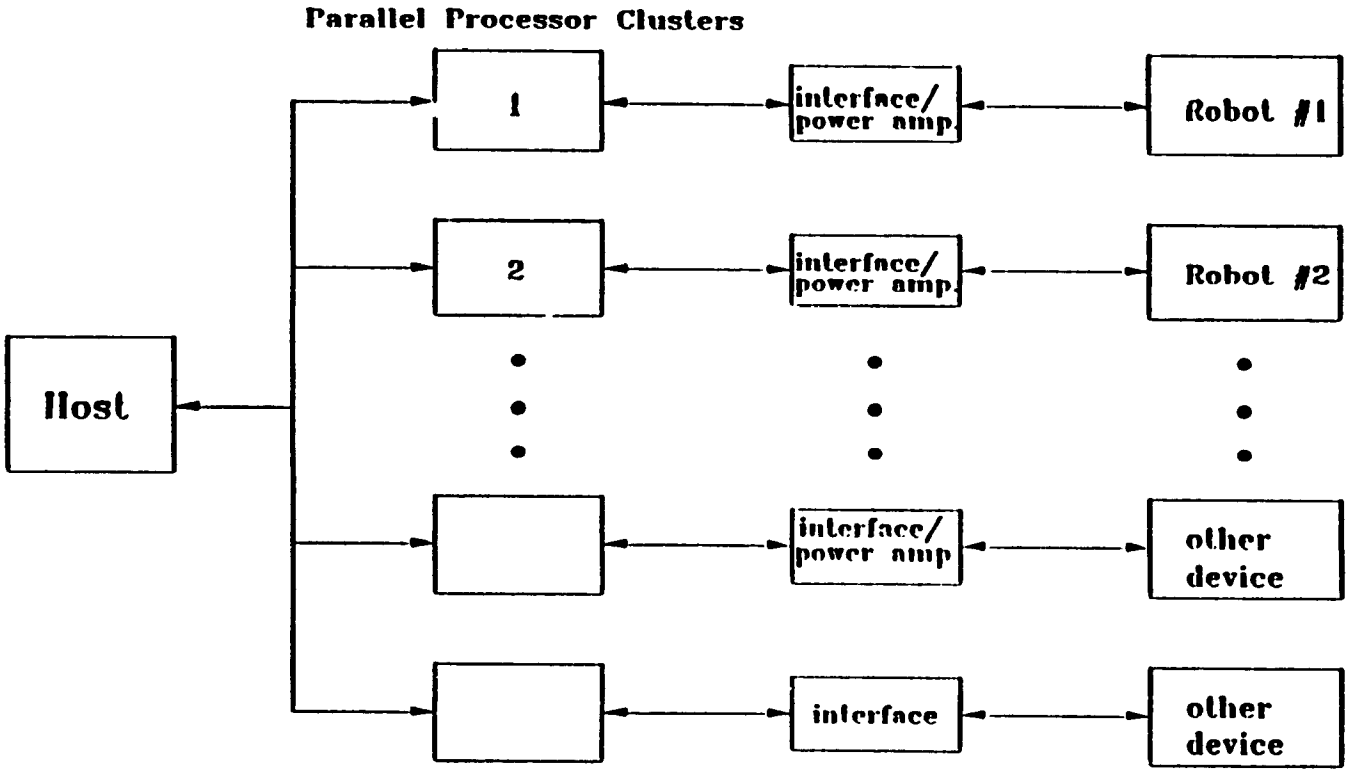


Fig.3.7 Architecture of Multi-Device Workcell Controller

The registers (A, B and C) as a hardware stack which is used for most integer and address arithmetic. The small number of register, together with the simple instruction, enables the processor to have relatively simple and fast control logic.

Each processor has four bi-directional serial links to provide communications between processors with the speed up to 20 Mbits/sec. The high speed of the serial link also reduces interprocessor communication overhead. Each transmitting data will be acknowledged, thus, a sending link will not send another data until it receives the acknowledge signal. Acknowledge signal is indicated by a start bit followed by a stop bit. The protocol permits an acknowledge signal to be generated as soon as the receiver has identified the data package. This helps the processor to receive an acknowledge signal before all the data in the package has been transmitted. In this case, the sending and receiving process are synchronised automatically by the hardware architecture[37].

3.5.3.2 memory

Each T800 transputer has 4k bytes of on-chip (internal static) memory for fast operation and up to 8M bytes of external local memory space available in market. The local memory architecture can also eliminate the problem of bus contention.

3.5.3.3 transputer module

Transputer modules (TRAMs) are the board level transputers with simple and standardised interface. It consists of a section of zero wait static RAM, one wait state static RAM, subsystem controller circuitry, and four serial links for interprocessor communication. A

typical transputer module B404-3 (fig.3.8) has a T800-20 transputer and 2 Mbytes of dynamic memory. A transputer network can be formed by connecting a number of transputer modules together.

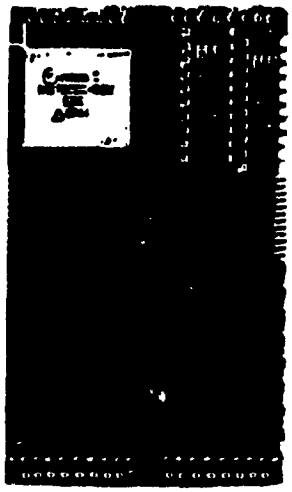


Fig.3.8 B404-3 Transputer Module

3.5.3.4 network

The transputers can be connected and expanded to become a multi-transputer stwork. Typical configurations may be the Cube(8 transputers), Hypercube(16 transputers) and Torus(64 transputers) as shown in fig.3.9.

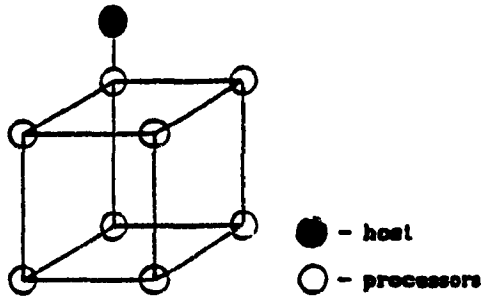


Fig.3.9a Cube Network

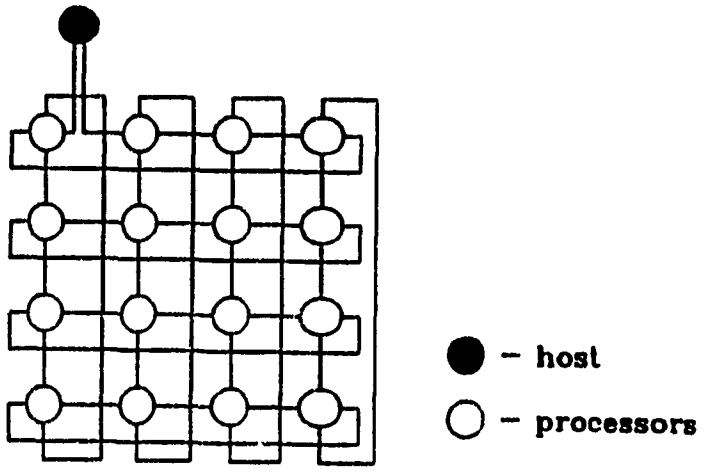


Fig.3.9b Hypercube Network

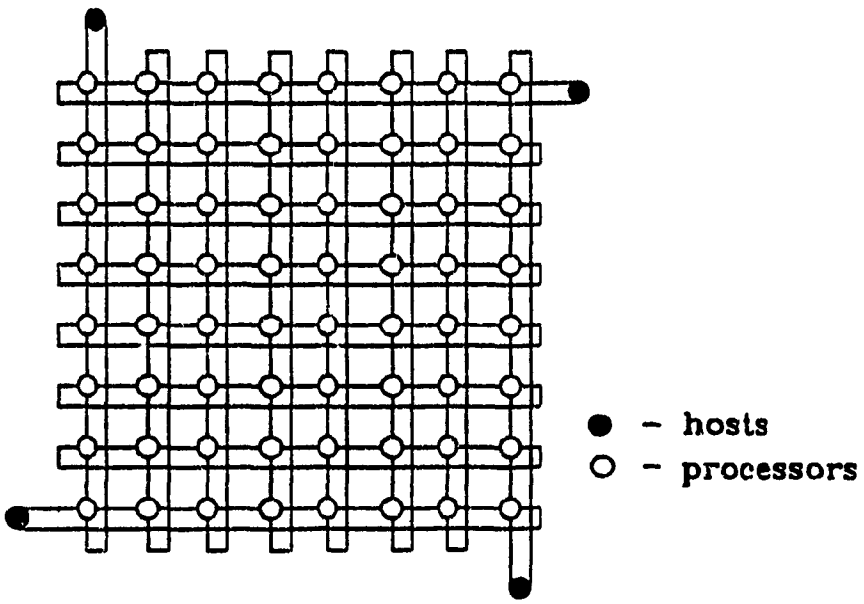


Fig.3.9c Torus Network

3.5.3.5 motherboard

The transputer motherboard is a board to be seated with a number of transputer modules to form a multi-transputer network. The motherboard is then plugged into a host computer. The host may be a Sun, VAX, Mac or IBM compatible computer. An IBM is preferable because of its cost effectiveness and its popularity of interface to various real world equipment, e.g. camera interface, data acquisition. One of the IBM bus transputer motherboard is the Inmos B008 shown in fig.3.10. It can hold maximum up to four B404-3 transputer modules. It consists of the interrupt logic, DMA logic and the reset logic interfaced to the IBM PC bus. The B008 board can connect other B008 boards to expand the transputer network in a form of hierarchical structure (fig.3.11). A T212 transputer and a Programmable Link Switch(C004) can be programmed for the dynamic network configuration. If one wants to over-ride the switch and use the hardwire connection, one has to take the C004 out from the B008. Otherwise, the modules may be burnt. In a multi-transputer network, the transputer connected to the host may be called as the Root Transputer and other transputers are the Slave Transputers. In a multi-transputer network, only the root transputer has the privilege to communicate with the host. The communication between the root transputer and the host is through a link adaptor. A link adaptor converts the transputer link serial signal into parallel data or vice versa[39]. Communications are activated between root transputer and the host computer under the following conditions :

. standard I/O (e.g. keyboard, console, etc.)

. data transfer to/from host parallel port

3.5.3.6 transputer I/O

Transputer input/output can be classified into two types which are shown in fig.3.12 :

. host dependent I/O

. local I/O

host dependent I/O

The host dependent I/O depends on a file server running in the host to transfer data to/from the root transputer. This type of I/O is very slow because of three reasons :

1. The slave transputers have to pass through a pipeline to reach the root transputer for transferring data. In addition, such transferring process also disturbs the efficiency of parallel processing because other transputers have to carry an extra responsibility to transfer I/O data for other transputers.

2. The I/O speed is slowed down because of the handshaking procedures of the file server.

3.0 Since the host bus can only handle one data at a time, the I/O operation has to be performed sequentially.

local I/O

The local I/O is directly interfaced to the transputer links. It does not require the file server to transfer the data. Each transputer may take care of its I/O. The local I/O can be A/D, D/A, RS232 or parallel ports. The local I/O interface transforms the analog or digital signal into transputer serial link. This I/O type has three advantages :

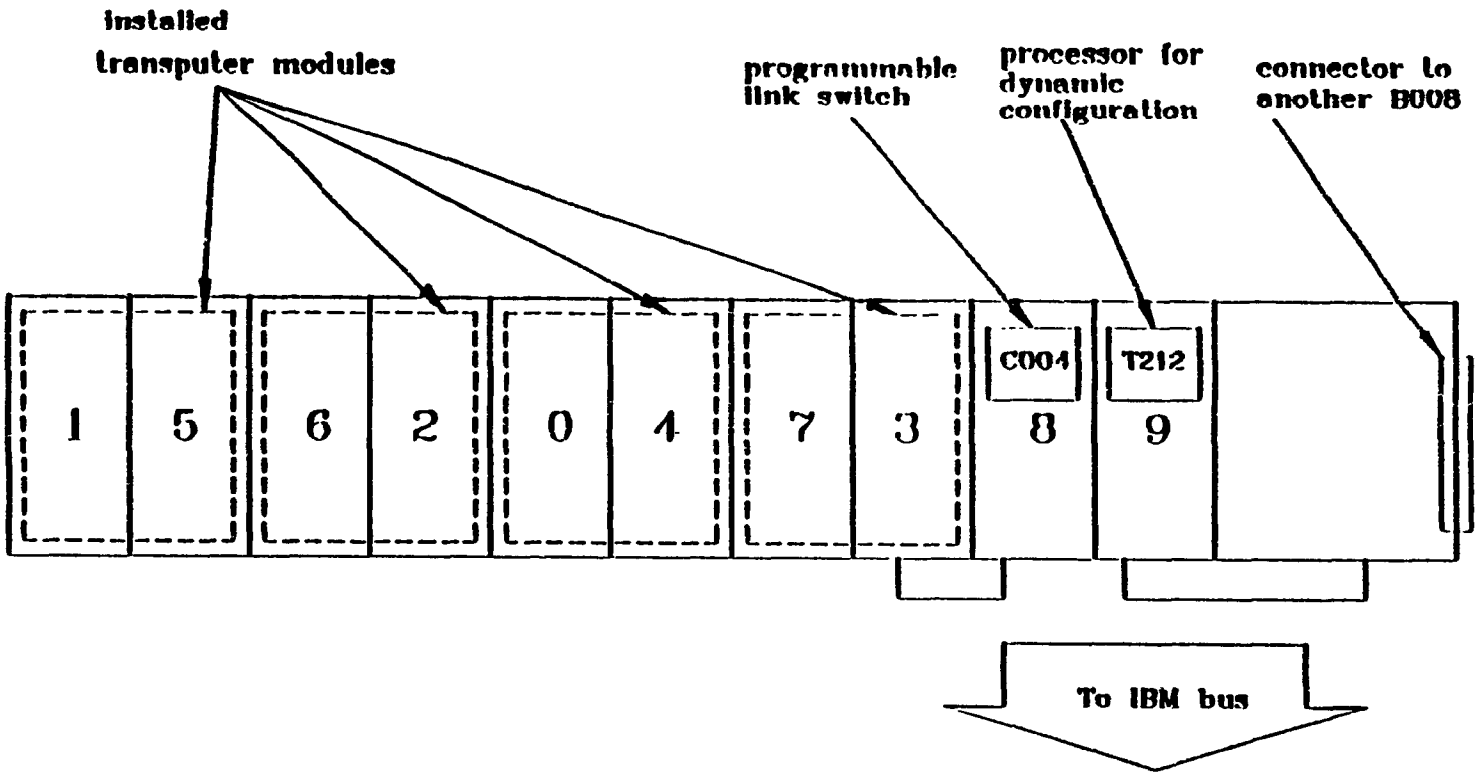


Fig.3.10 Inmos B008 Transputer Mother Board

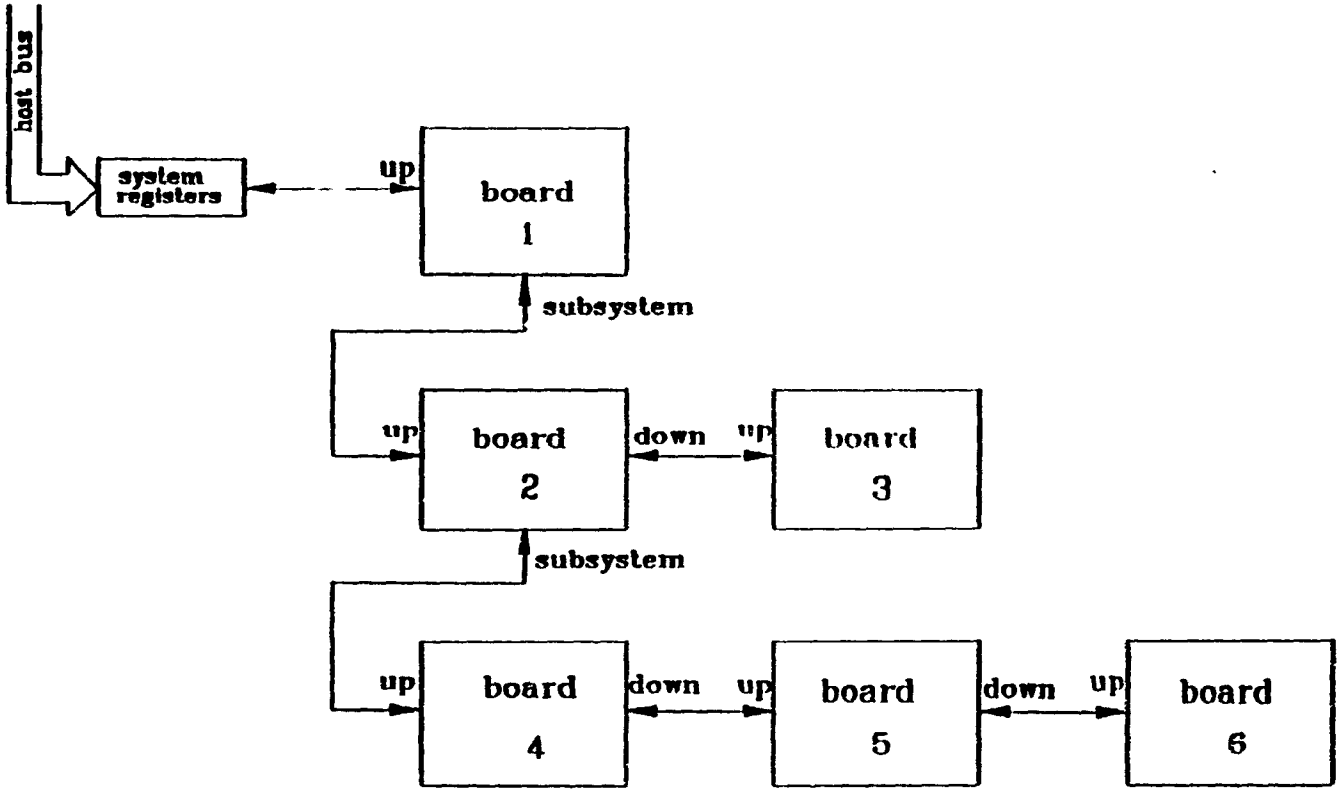


Fig.3.11 Hierarchical Structure of Multi-Transputer Network

1. Each slave transputer has its own I/O. The necessity to depend on the root transputer and the host to transfer I/O data can be eliminated.

2. The I/O speed is much faster because of eliminating the handshaking process of the file server.

3. Slave transputers may control their device independently and simultaneously.

4. The problem of bus contention can be eliminated.

The disadvantage of this type is that one of the links will be occupied for the I/O channel.

3.5.4 Controller Software

3.5.4.1 compiler

The OCCAM language[41] was developed by Inmos particularly for the application of the concurrency and communication. OCCAM may be recognised as a good programming model for the transputers, especially for concurrency and parallelism[61]. However, its inadequacy in pointer operation, data structure, recursion and dynamic allocation[91] sometimes cause inconveniences for the software development. It is preferable to use a high level language, such as parallel C or parallel Fortran. Entire parallel system can be programmed and developed using these high level compilers without the need for the conventional transputer development tools, such as OCCAM tool set. There are several parallel C compilers available in market, Definicon parallel C, Logical system and the 3L. Besides the similarity of common C compiler, they also consists of the ability for the users easily to access the DOS function and perform the interprocessor communication. For example, the 3L compiler [1]

accesses the DOS function 16h as following.

```
#include <dos.h>
static int my_int86(n, r1, r2)
int n;
union REGS *r1, *r2;
{       int86(n, r1, r2);
        return(r2->x.ax);
        }
#define int86 my_int86
main()
{
        char c;
        c = get_key();
        }

get_key()
{
        union REGS r;
        r.h.ah = 0;
        return int86(0x16, &r, &r);
  }
```

A configurer shown in fig.3.13 provides the user to declare the physical processors, link connections between transputers, task assignment to transputers and communication channels of processes inside the transputers[2]. It also provides a convenient way for the programmer to re-configure the system tasks in a multi-transputer network if one of the processors fails or one of the tasks is abandoned.

3.5.4.2 communication drivers

There are three types of communication drivers required to operate this controller. They are :

. interprocessor communication driver

. host/root transputer communication driver

. local I/O driver

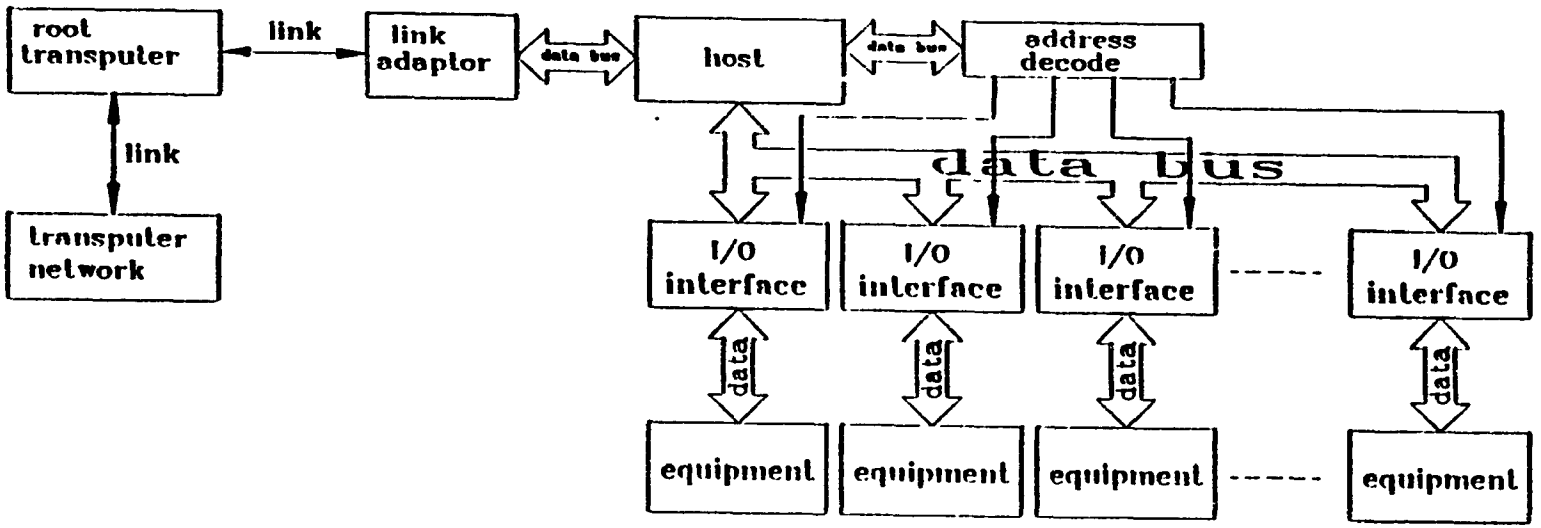


Fig.3.12a Transputer Host Dependent I/O

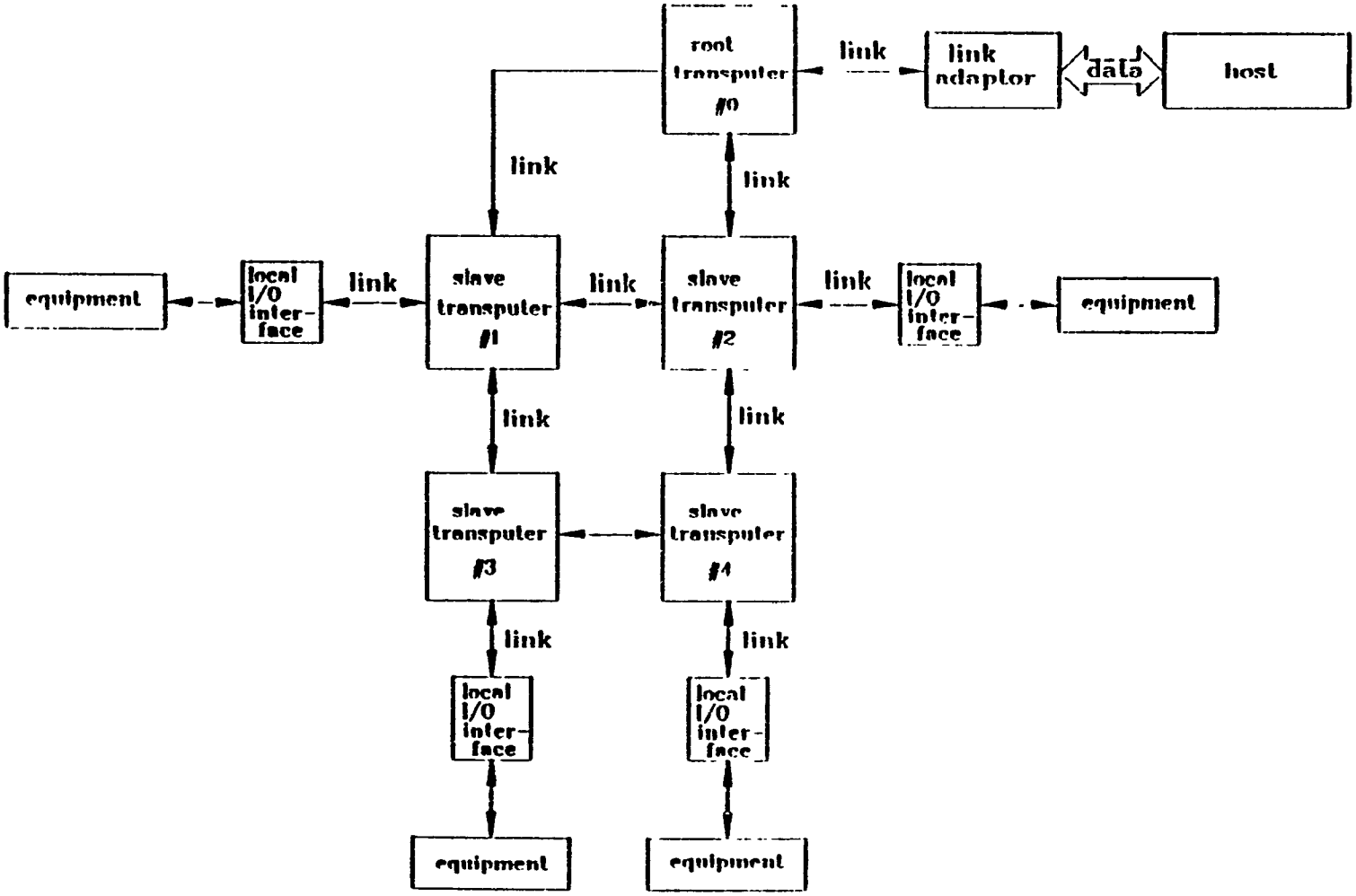


Fig.3.12b Transputer Local I/O

interprocessor communication:

The 3L parallel C compiler provides the basic communication facility, chan_in_x() and chan_out_x() to transfer data between transputers via the links. The 'x' can be a byte, word or message[3] . For example,

```
#include <chan.h>   /* declare the address of link      */
main()
{
  unsigned char c;

 /* input a byte via link 1 from other transputer    */
  chan_in_byte(&a, Link1Input);

  /* output a byte via link 2 to other transputer    */
  chan_out_byte(a, Link2Output);
  }
```

host/root transputer communication driver:

The motherboard B008 itself does not support any peripheral equipment. Therefore, any standard or host dependent I/Os have to depend on the file server running inside the host to serve the I/O request from the root transputer. Unfortunately, the one provided by 3L or other vendors is a polling basis file server. If the root transputer has any request, the server executes the requested task. After the server has finished the requested task, it goes to poll again until the next request occurs. Therefore, the IBM host is totally tied down and not able to support other equipment, e.g camera.

```
!
! example.cfg       configurer example of a two transputers network
!
processor host      ! declare the host processor
processor root      ! declare the root transputer
processor second    ! declare the second transputer
!
! declare the hardwire wire link connection
wire ? host[0] root[0]  ! link 0 host connected to root transputer link 0
wire ? root[2] second[1]! root transputer link 2 connected to second
                        ! transputer link 1
!
! task declarations and channel I/O ports and memory requirements
!
task afserver  ins=1 outs=1
task filter    ins=2 outs=2 data=10k
task task1     ins=3 outs=3 data=80k
task task2     ins=1 outs=1 data=20k

!assign software tasks to physical processors
!
place afserver host
place filter   root
place task1    root
place task2    second
```

Fig.3.13 3L Parallel C Configurer

In order to retain the capability of the IBM host, a customised Interrupt Driven File Server (IDFS) is developed to replace the the polling basis file server. The IDFS consists of two sections of codes - initialisation code and resident code. The initialisation code is to load the resident code inside the host memory. The resident code is actually to serve the root transputer requests. After the resident code has been loaded by the initialisation code, the resident code is turned in a sleep state. If the root transputer has any request (e.g. printf, port_in, etc.), it enables the hardware interrupt to wake up the file server(fig.3.14) to perform the requested task. After the server has finished the request, it goes to sleep again until the next request. In this case, the host can still use the time-slice between the requests to execute its DOS program while the transputers are running their tasks.

Concerning the data transfer between the host and root transputer, a Virtual Dual Port RAM is assigned for transferring data between the host and root transputer. It can be accomplished by the transputer functions mem_write() and mem_read(). Before using these functions, the memory in transputer and host should be allocated. Otherwise, program codes may be overwritten to cause unpredictable results. The codes of the memory write and memory read tasks are shown as the followings.

```
#define mem_write_cmd    38
#define mem_read_cmd     39

mem_write(offset, value)
int offset;
int value;
{
  int status;
  _put_int(mem_write_cmd); /* send command to host  */
  _put_int(offset);        /* send offset to host   */
  _put_int(value);         /* send data to host     */
  _get_int(&status);
  }

mem_read(offset, value)
int offset;
int *value;
{
  int status;
  _put_int(mem_read_cmd);  /* send command to host  */
  _put_int(offset);        /* send offset to host   */
  _get_int(value);         /* get data to host      */
  _get_int(&status);
  }
```

3.5.5 Task Implementation

3.5.5.1 task distribution

Fig.3.15 shows the current implementation using a host(12 MHz INTEL 80286/80287) with two transputers. The host is mainly responsible for the management user interface and image processing. The forward kinematics is assigned to the root transputer for position feedback and the second transputer performs a high level robot path generation. The reason to use the host for the image processing task is mainly because of the IBM bus camera. In the experience of the CONCIC workcell, the supervisory control functions as well as the image processing is adequately performed by the host. At this present stage of development, the actual motion joint control of the robot is still the responsibility of the controller that came with the robot.

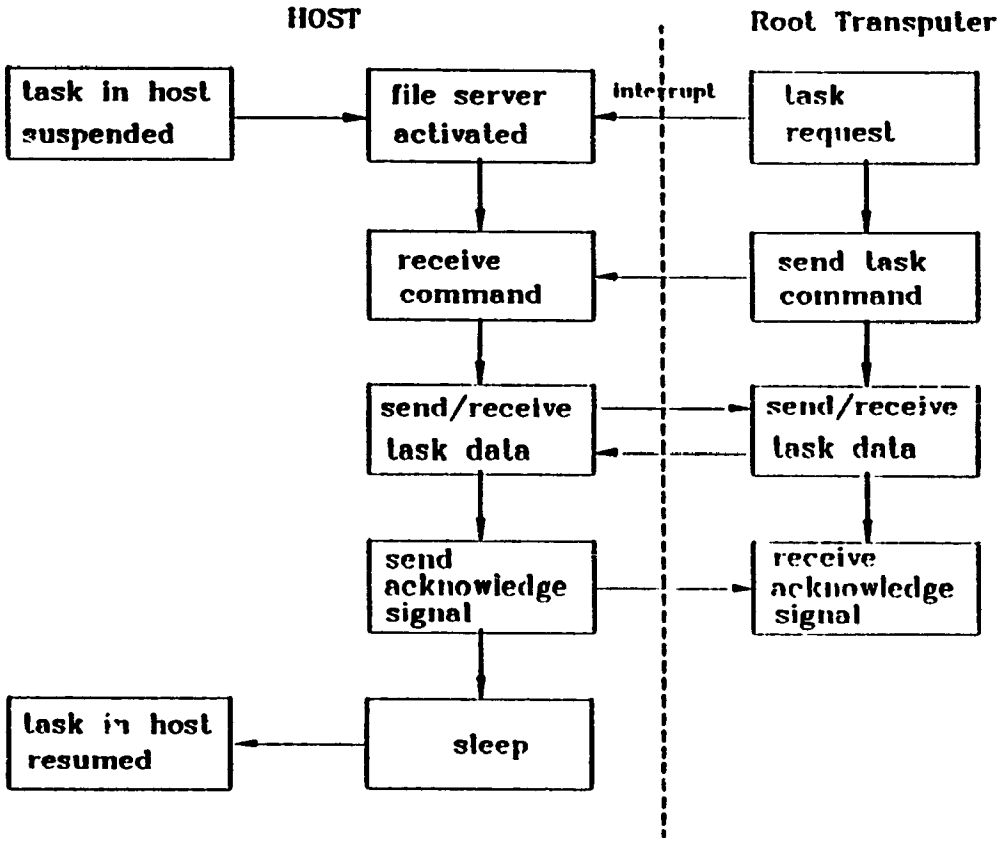


Fig.3.14 Operation of Interrupt Driven File Server

3.5.5.2 task communication and scheduling

Fig.3.16 shows the schedules and communication of the implemented tasks in the robotic workcell. An operation cycle has two stages - image processing and real-time path control. At stage 1, only the host is activated to perform the image processing task. The transputers and the robot controller are ready for the stage 2. After the workpiece geometrical data have been identified by the primary image processing in stage 1, the data management puts the data in the VDPR. In stage 2, i.e. real time path control stage, the root transputer takes the workpiece geometric data, conveyor position and velocity from the VDPR through a link adaptor. The root transputer transfers the data to the slave transputer through a serial link. According to the geometrical data, the slave transputer calculates the incremental position commands for each iterations. It sends the commands to the robot motion controller through a transputer local I/O, link/RS232 adaptor[81]. At the same time, the root transputer also activates the IDFS inside the host to input the workpiece position, conveyor velocity and the joint angles for every 28 msec. It also calculates the current cartesian coordinate of the end-effector. The feedback position is then transferred to the slave transputer to readjust any synchronisation offset between the moving workpiece and the robot end-effector.

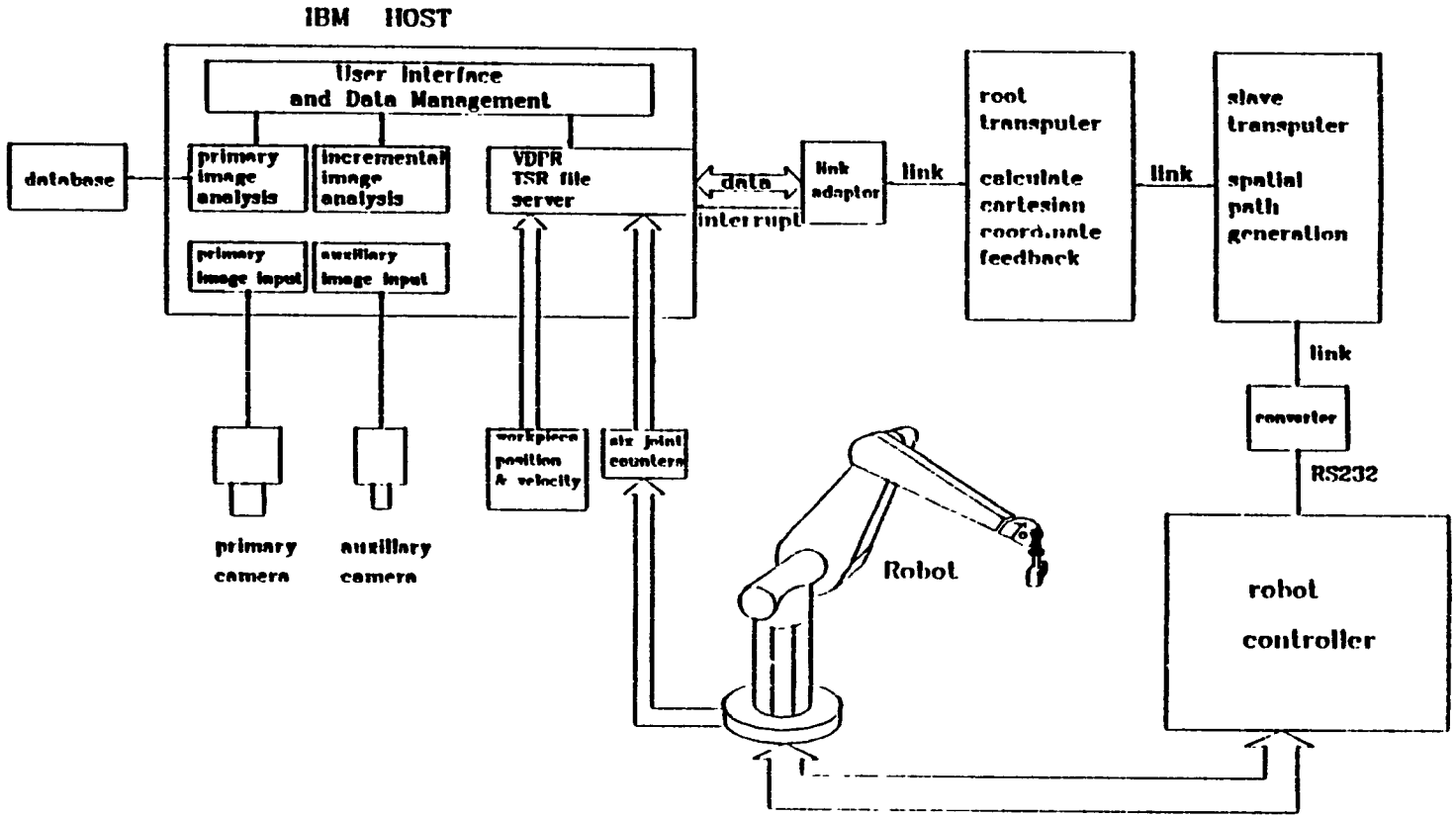


Fig.3.15 Task DIstribution of the Multiprocessor Workcell Controller

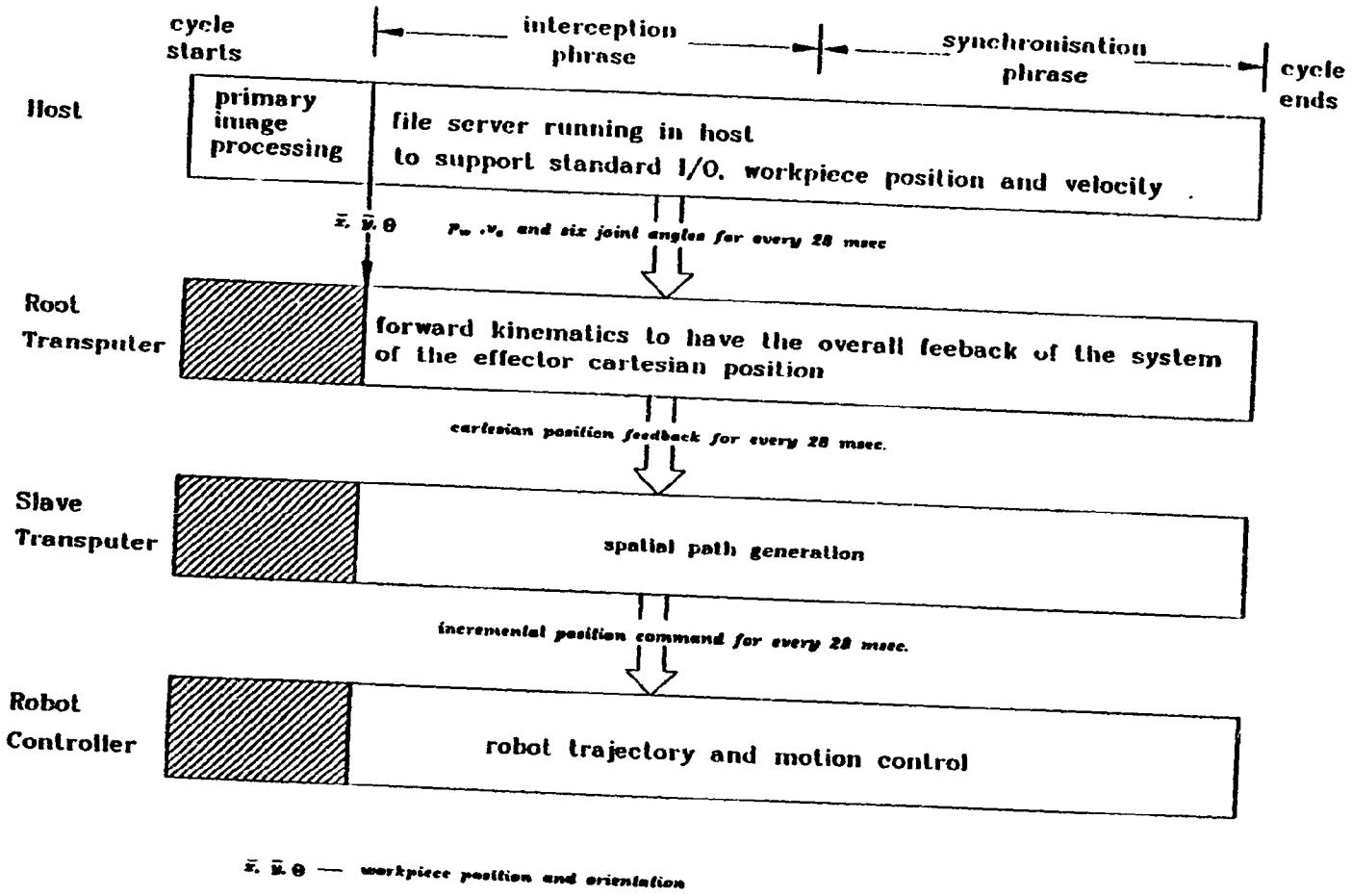


Fig.3.16 Schedule and Communications of the Tasks

3.5.6 Performance

The performance of the transputer and the host processor have been measured and compared. The performance are outlined as the following groups :

- general mathematical operation
- 6-dof robot kinematics
- spatial path generation
- interprocessor communication
- system I/O

By comparing the transputer benchmark with the IBM host (12 MHz AT), the transputer is 3 times faster in the integer operation, 26 times faster in the floating point operation and 4 times faster in the transcendental function execution.

The optimised 6-dof robot (PUMA 560) kinematics control codes takes 1.394 msec runnning in one T800-20 transputer. The time is reduced to 914.3 μsec with 2 transputers and 785.0 μsec with three transputers[88]. For real-time application, it is found that one transputer is sufficient to calculate the cartesian end-effector position within the PUMA sampling time (28 msec).

The spatial path generation task running in the slave transputer communicates with the robot controller within 28 msec. The slave transputer solves the linearised elliptical integral in 1.11 msec, compared to 8.5 msec for the IBM-AT.

The inter-communication between the tasks can be a byte, word, single precision or double floating point. Although the standard transmission rate between two transputers is specified to be 20 Mbits/sec, the speed is much reduced in actual operation. The

following figures are obtained in experimentation :

format	time (μsec)
byte (1 byte)	7.373
word (4 bytes)	9.293
float (4 bytes)	9.638
double(8 bytes)	11.750

It can be found that the inter-processor communication speeds are not really proportional to the number of bytes. It may be due to a portion of time to be spent on the communication set-up process.

The joint angles, workpiece position and velocity are obtained from the parallel ports of the IBM host. The root transputer depends on the file server to input these data for calculation. Because of the handshake procedures between the file server and the root transputer, it takes about 0.6 msec to input one byte of data from the parallel ports. Such slow I/O speed becomes the bottleneck of the system.

3.6 SUMMARY

This chapter has analysed the common controller architectures. The multiprocessor configuration is selected as the most suitable configuration of the workcell controller because of its intensive computation power, higher flexibility and its uniformity. A brief review of the common multiprocessor system available in market has been discussed. The transputer is ultimately selected as the better processing element for the workcell controller implementation. The hardware and software capability of the transputer has been described briefly. This chapter has also described a phase of development of a transputer-based hierarchical controller. An interrupt driven file server has been developed in order to retain the capability of the host and to provide the I/O interface of some special equipment. A Virtual Dual Port RAM has also been assigned in order to have a

common buffer between the host and the transputer network for inter-memory communication. It is found that the bottleneck of the system is the slow speed of the host dependent I/O.

Chapter 4

Path Generation Task

4.1 INTRODUCTION.

The primary functions of this controller in the experimental robotic workcell are

(i) collect and process image data, obtain conveyor speed and displacement

(ii) execute a path-planning algorithm for synchronization control

(iii) physically move the robot

Very often, the last function (iii) is assumed to be the responsibility of the motion controller supplied by the robot manufacturer. It is there a matter of understanding the operation of this controller, and integrating function (ii) with it by suitable real-time communication facilities.

This chapter focuses on the design of the path-planning algorithm, which takes into account the characteristics of the motion controller. It is therefore appropriate to quickly outline in section 4.2 the characteristics of the motion controllers in most industrial robots, followed by the formulation of two control models in sections 4.3.1 and 4.3.2. The results of implementation and the comparsion of two control models will be discussed in section 4.4. Section 4.5 provides a summary of the research work in this section.

4.2 DESIGN OF ON-LINE MOTION CONTROLLERS OF ROBOTIC MANIPULATORS

Only continuous path control is considered here, since a point-to-point motion controller is not suitable for synchronizational

control.

In most applications of industrial robots, the location (i.e. position and orientation) of the end-effector may be pre-determined as a trajectory between some initial locations and final locations. The actual trajectory is constructed as a sequence of discrete intermediate locations, the spacing of which is dictated by the required resolution and other considerations. Curve-fitting techniques of one type or another are employed so as to regenerate the trajectory. Because of the repetitiveness of most automations in which these robots are employed, it is possible to customize and optimize the intermediate locations, using such techniques as those proposed by Lin and Chang[53]. These off-line techniques are however not suitable for the kind of intelligent applications that are the subject matter of this chapter, since all robotic locations are to be decided upon in real-time.

For real-time path planning, it is usual to assign a fixed time-interval between intermediate locations, since the controller is an essential discrete controller. This time interval, designated as τ_s is the sampling period of the controller. Its magnitude is dictated by the maximum speed of the end-effector, the complexity of the curve-fitting and interpolating techniques used, the computing power available, etc.

Real-time path generation works with a limited knowledge of the trajectory that lies ahead. There are unavoidable delays between the generation of a desirable location and the actual arrival of that location. These delays are contributed by several factors, such as the number of locations ahead in order to generate a smooth path segment, the time required to apply the curve-fitting techniques on the

available input data, the time required for the interpolation and servo delays.

Many motion controllers implement path segments with constant velocities between locations, and with constant accelerations between segments. Others rely on numerical curve fitting methods, yielding a series of polynomials that satisfy continuity constraints. The reader is referred to the literature for abundant examples of these techniques [5,16,17,25,26,28,46,51,54,57,82,83,84,93,101].

To illustrate the schemes used by a sector of industrial robotic controllers (including the PUMA family), refer to fig.4.1, in which all events are referenced to the sampling interval τ_s.

In order that the robot moves from a position p_{k-1} to the next p_k, a cubic polynomial Q_k is to be determined, requiring as data the knowledge of desired locations p_{k-1}, p_k, and p_{k+1}, as well as the previous polynomial Q_{k-1} for continuity.

So much for the data to generate the polynomial Q_k. But its implementation takes time, as summarized by a preparatory delay τ_d, and an execution delay that is equal to 1 sample interval τ_s. This gives rise to a total delay of

$$\tau_a = \tau_d + \tau_s$$

In the case of the PUMA 560 the following has been determined experimentally[72] :

$$\tau_s = 28.8 \text{ ms},$$

$$\tau_d = 67.3 \text{ ms } (\cong 2.4 \times \tau_s)$$

$$\tau_a = 96.1 \text{ ms.} (\cong 3.4 \times \tau_s)$$

It should be obvious that real-time path generation with adaptive capability is complicated by the necessity of predicting the desired position of the robot in view of disturbances in the working

environment. It is the objective of this section to focus on a predictive technique to take care of this complication.

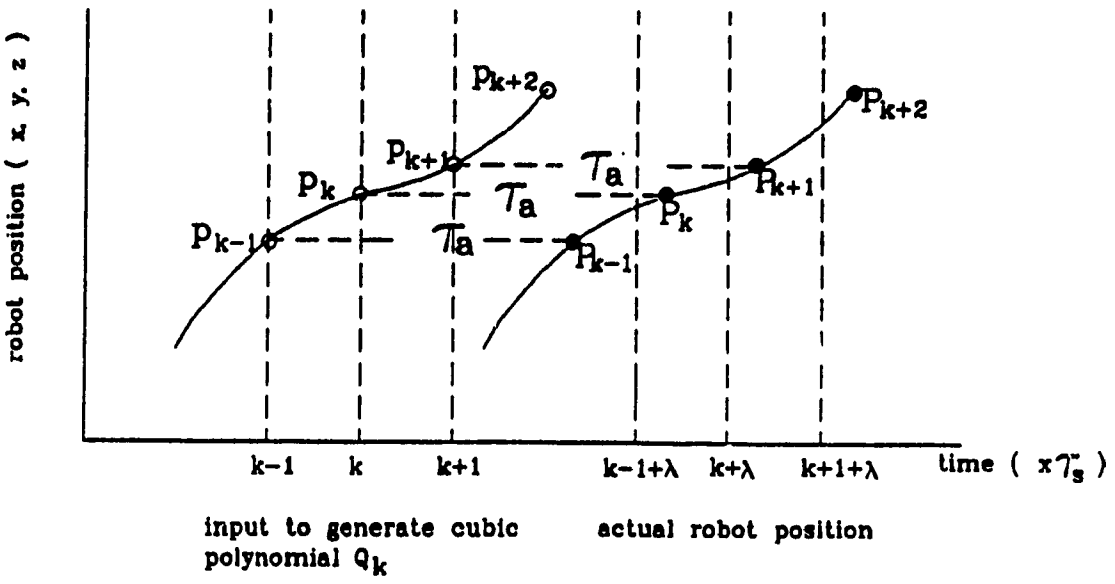


Fig. 4.1 Time Delay Between Position Command and Actual Position of Robot

4.3 PATH CONTROL

Path control for a static environment is quite simple but if the environment is changable, then the control become more complicated. First of all, a concept has to be clarified between the terms *trajectory* and *path*. The terms are defined as follows by Brady[10]

> *"A trajectory is the time sequence of intermediate configurations of the arm between the source* P_o *and the destination* P_1.P_1.. *The space curve traced by the effector is called the path of the trajectory "*

In the past, many researchers[56,83,86,94] have developed theories and algorithms to specify the robot trajectories. Some of them[53,56] emphasise on minimising the total time spent on moving between two points in space. Other researchers[92,94,95] develop theories to find the minimum travelling time under certain dynamic constraints. Kyriakopoulos and Saridis[49] minimise the jerk as the goal of optimisation. All the theories are mainly aimed for the trajectories. It is rare to have control models to describe the gobal path control, especially deal with the changable environment such as a moving workpiece on a conveyor. Therefore, this section will discuss two global path control models.

In the past, a path control model shown in fig.4.2 was developed by Tom Montor[73]. The robot travels at the xy plane as the interception phase and then travels at the xz plane to approach the moving workpiece as the synchronisation phase. This control model can be modified to intercept the workpiece in a shorter period of time and thus the productive space can be increased. Two improved control

models are proposed, formulated and implemented on the experimental workcell. The first control model shown in fig.4.3 takes a linear path to intercept the workpiece with a constant velocity. It is found that the acceleration and deceleration distances are insignificant w.r.t. the whole global path. It can be approximated that the linear path is more or less the shortest path, i.e. takes minimum time. Once it arrives the interception point, it starts the synchronisation stage to perform designated operation. It is called to be the "Minimum Time" model which will be abbreviated as the MT model. The second control model takes an elliptical path in order to eliminiate the abrupt change of motion at the interception point (fig.4.3). It is thus called the "Reduced Torque" model which will be abbreviated as the RT model..

While kinematic relationships between the end-effector and the conveyor have been developed and discussed previously, this section develops two discrete control models which lend themselves to digital implementation. In particular, it takes into full account the time delay element discussed in section 4.2, and disturbances which adversely modifies the velocity and position of the workpiece and the conveyor.

4.3.1 "Minimum Time" Discrete Control Model

This section deals with developing a simple real-time strategy to control the end-effector travelling in a straight line, so as to most accurately intercept the on-coming workpiece travelling on a variable-speed conveyor in minimum time. The formulation will be divided into two parts - 1) interception phase; 2) synchronisation phase.

AB – interception phrase

BC – synchronisation phrase (approaching)

CD – synchronisation phrase (productive)

A – home position

B – intercept point

C – arrival position

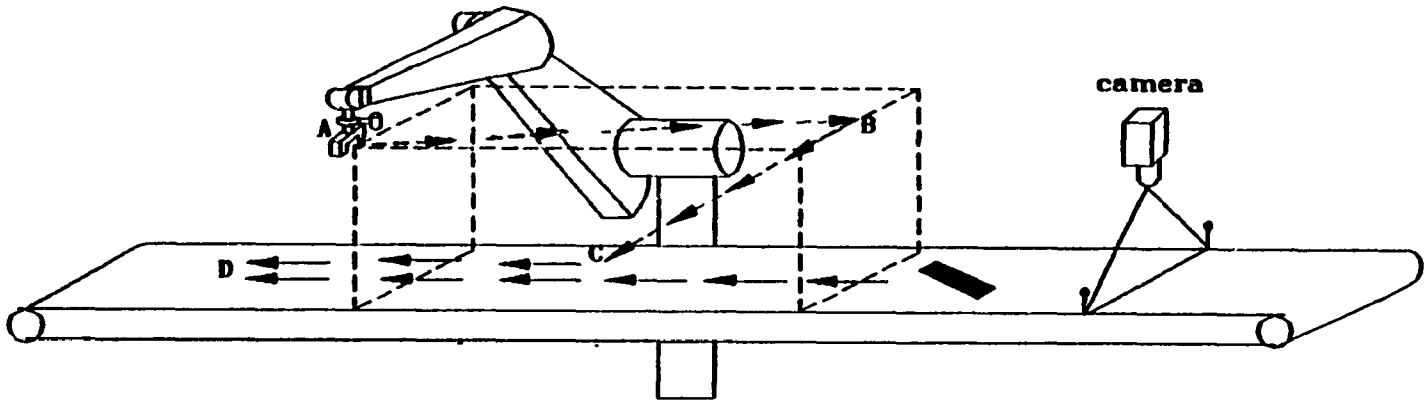


Fig.4.2 Path Control Scheme Developed by Tom Montor

AB – interception phrase

BC – synchronisation phrase (approaching)

CD – synchronisation phrase (productive)

A – home position

B – intercept point

C – arrival position

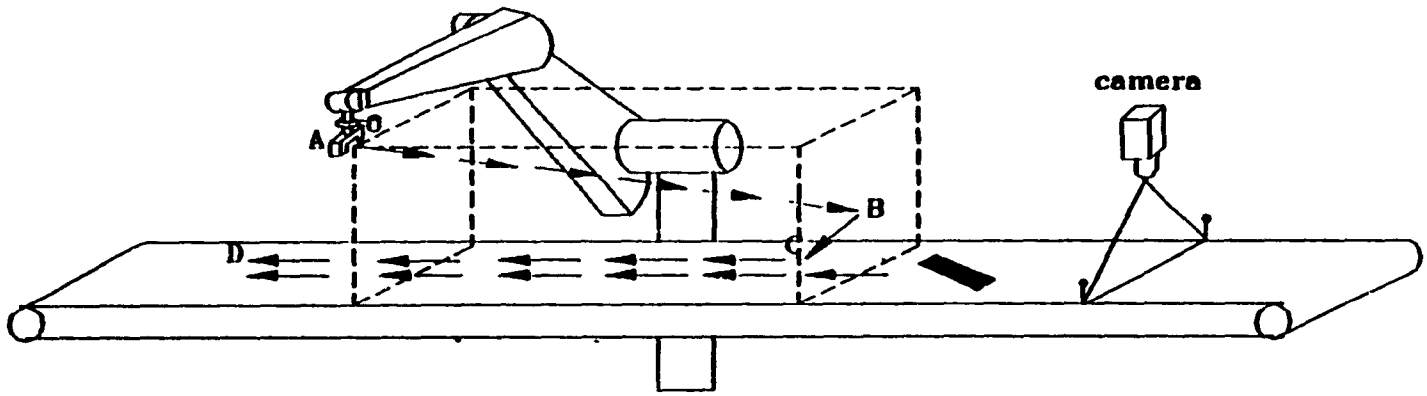


Fig.4.3 Path of the "Minumum Time" Model

AB – interception phrase

BC – trasnition phrase

CD – synchronisation phrase (productive)

A – home position

B – intercept point

C – arrival position

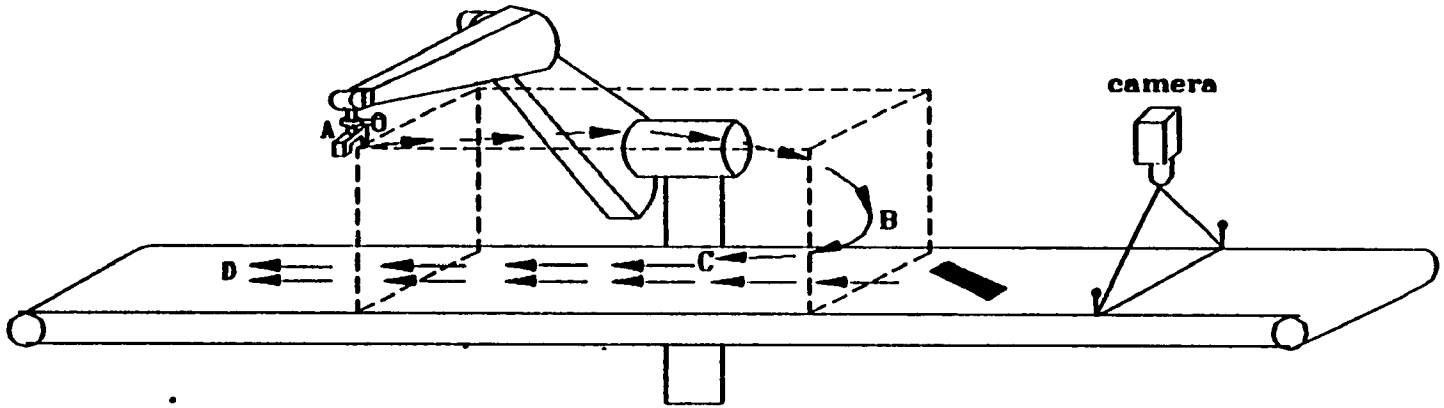


Fig.4.4 Path of the "Reduced Torque" Model

4.3.1.1 interception phase

Referring to fig.4.5, let the starting position, i.e. home position, of the robot to be A (x_o ,y_o ,z_o). Two displacement vectors are defined with respect to the starting position : $p_r[k]$, the accumulative positional command at time k, and $P_w[k]$ the measured position of the workpiece.

Now recall from fig.4.1 that for a positional command issued at time [k-1], the robot will be expected to arrive there at a time τ_a later,

$$t = (k-1)\tau_s + \tau_a$$
$$= k\tau_s + \tau_d$$

At time k, let us look into the future by τ_d ahead, i.e. at $t = k\tau_s + \tau_d$. The position D of the robot is given by the previous command p[k-1], while the position F is given by the measured position $P_w[k]$ plus a distance traversed by the conveyor at the measured speed $v_c[k]$ for the duration τ_d ($v_c[k]$ being assumed constant during τ_d).

From these positions, assuming an appropriate robot velocity $v_r[k]$ can be found and maintained constant for the next m[k] periods of time, it is conceivable that the robot will meet the workpiece in position B. The vector equation can be expressed as follows,

$$\mathbf{AD + DB = AE + EF + FB}$$

$$\text{i.e. } p_r[k-1] + v_r[k]m[k]\tau_s = P_w[k] + v_c[k]\tau_d i + v_c[k]m[k]\tau_s i \quad ...(4.1)$$

This equation enables one to determine the robot velocity command vector $v_r[k]$, and therefore the incremental positional command r[k]. The relationship between r and the accumulative command p_r is given as follows:

$$p_r[k-1] = \Sigma r[i] = p_{rx}[k-1]i + p_{ry}[k-1]j + p_{rz}[k-1]k \quad ...(4.2)$$

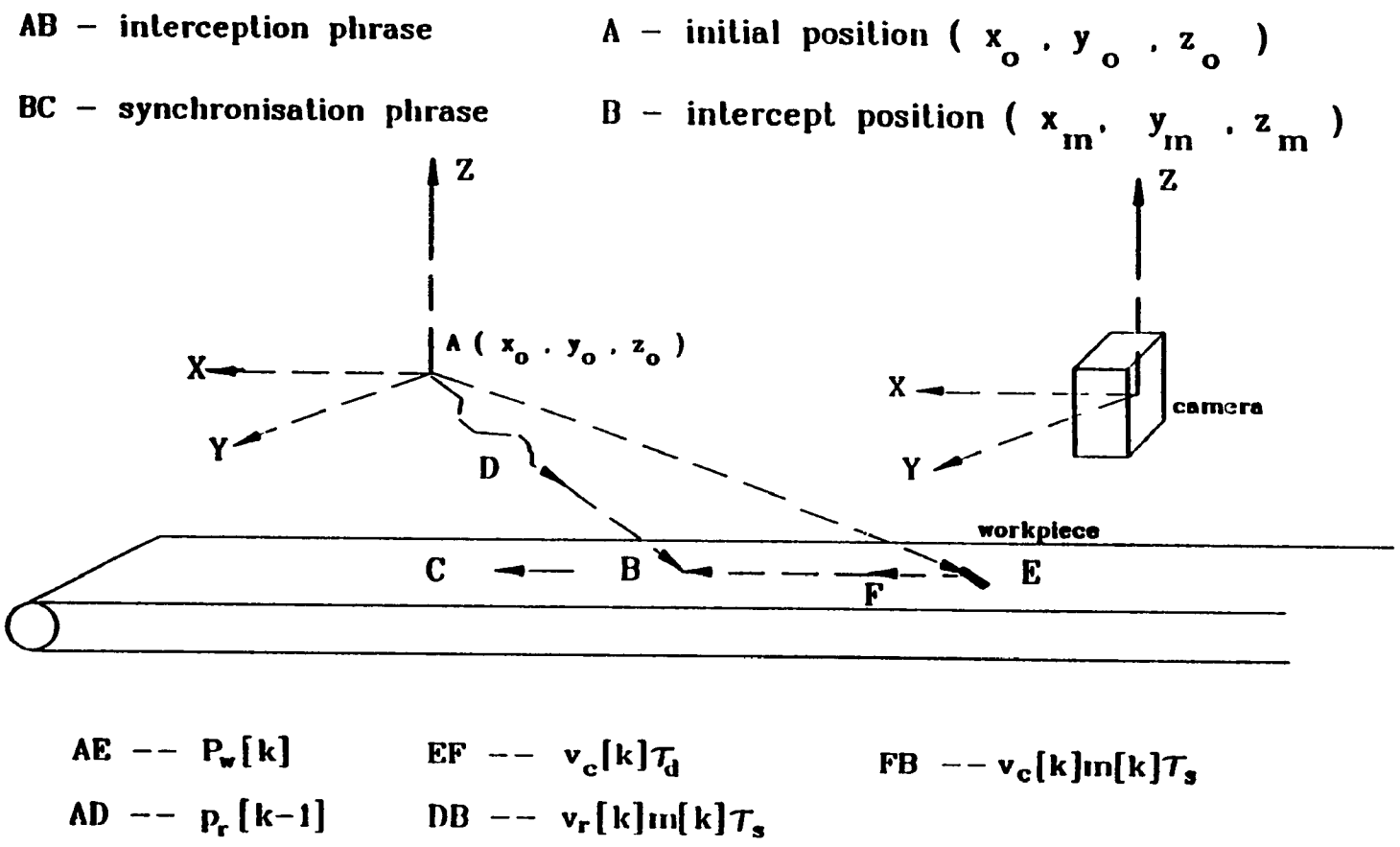


Fig.4.5 Relationship of Salient Position Vectors during the Intercept Phase (MT Model)

where the summation symbol Σ represents $\sum_{i=1}^{k-1}$ unless specified otherwise; i, j, k are unit vectors of displacement , and the components of $p_r[k-1]$ are

$$p_{rx}[k-1] = \Sigma r_x[i]$$

$$p_{ry}[k-1] = \Sigma r_y[i]$$

$$p_{rz}[k-1] = \Sigma r_z[i]$$

The position of the workpiece relative to the robot initial position and measured at time k is

$$P_w[k] = \{ \sum_{i=1}^{k} v_c[i]\tau_s - x_o \}i + (Y_c - y_o).j + (Z_d - z_o)k \qquad ...(4.3)$$

where

Y_c = lateral position of the centroid of the workpiece on the conveyor,

Z_d = vertical position of centroid of the workpiece modified suitably according to the design of end-effector

Substituting equations (4.2) and (4.3) into (4.1), resolving the robot velocity vector into components

$$v_r[k] = v_{rx}[k]i + v_{ry}[k]j + v_{rz}[k]k \qquad ...(4.4)$$

one gets

$$\{ \Sigma r_x[i]i + \Sigma r_y[i]j + \Sigma r_z[i]k \} + \{ v_{rx}[k]i + v_{ry}[k]j + v_{rz}[k]k \} m[k]\tau_s$$

$$= \{ \sum_{i=1}^{k} v_c[i]\tau_s - x_o \}i + (Y_c - y_o)j + (Z_d - z_o)k + v_c[k]\tau_d i + v_c[k]m[k]\tau_s i$$

$$...(4.5)$$

In practice, it simplifies matters considerably by prescribing a constant magnitude for the robot velocity:

$$|v_r[k]| = V_r \text{ for all } k.$$

(see Appendix D for the conditions in which V_r exists vis-a-vis a given conveyor speed v_c)

Equating components of i, j, k on both sides of equation (4.5)

$$r_x[k] = \left(\frac{C_1[k]}{m[k]} + v_c[k]\,\tau_s\right)$$

$$r_y[k] = \frac{C_2[k]}{m[k]}$$

$$r_z[k] = \frac{C_3[k]}{m[k]} \qquad \ldots(4.6)$$

where

$$C_1[k] = \sum_{i=1}^{k} v[i]\tau_s + \Sigma r_x[i] + v_c[k]\tau_d - x_o$$

$$C_2[k] = (Y_c - y_o) - \Sigma r_y[i]$$

$$C_3[k] = (Z_d - z_o) - \Sigma r_z[i] \qquad \ldots(4.7)$$

$$v_{rx}[k] = \frac{r_x[k]}{\tau_s} = \frac{C_1[k]}{m[k]\,\tau_s} + v_c[k]$$

$$v_{ry}[k] = \frac{r_y[k]}{\tau_s} = \frac{C_2[k]}{m[k]\,\tau_s}$$

$$v_{rz}[k] = \frac{r_z[k]}{\tau_s} = \frac{C_3[k]}{m[k]\,\tau_s}$$

$$\left[\frac{C_1[k]}{m[k]\,\tau_s} + v_c[k]\right]^2 + \left[\frac{C_2[k]}{m[k]\,\tau_s}\right]^2 + \left[\frac{C_3[k]}{m[k]\,\tau_s}\right]^2 = V_r^2$$

$$(V_r^2 - v_c^2[k])\, m^2[k]\tau_s^2 - 2\,C_1 v_c[k]\, m[k]\,\tau_s - (C_1^2 + C_2^2 + C_3^2) = 0$$

therefore, the value of $m[k]\tau_s$ is equal to the following,

$$m[k]\tau_s = \frac{C_1[k]v_c[k]}{V_r^2 - v_c^2[k]} + \left[\left\{ \frac{C_1[k]v[k]}{V_r^2 - v_c^2[k]} \right\}^2 + \frac{C_1^2[k]+C_2^2[k]+C_3^2[k]}{V_r^2 - v_c^2[k]} \right]^{1/2} \quad \ldots(4.8)$$

Equation set (4.6) constitutes the displacement commands at time k in cases where $m[k] \geq 1$.

When m[k] turns out to be less than 1, i.e. theoretically requiring less than one sample period to complete the interception, the robot is instructed to use the following adjusted increment:

$$r_x = p_w[k] + v_c[k]\tau_s + p_{rx}[k-1] - x_o$$

$$r_y = (Y_c - y_o) - p_{ry}[k-1]$$

$$r_z = (Z_d - z_o) - p_{rz}[k-1] \quad \ldots(4.9)$$

4.3.1.2 synchronization phase

Robotic operations at a workcell may require the end-effector to come into physical contact with the workpiece (as in the case of pick and place, machining, assembly etc.), or maintain a steady proximity relationship (as in the case of spraying, quality inspection via camera vision, etc.). Whatever the application, the end-effector is required to arrive at a situation of zero position error and zero velocity error reasonably quickly after the interception phase is completed :

$$P_r[k] = P_w[k]$$

and $$v_{rx}[k] = v_c[k] \qquad \text{for all k.}$$

(Note: the robot speed cannot assume a constant value, as in the interception phase)

In the context of the present scope where relative displacement between the workpiece and the conveyor is assumed to be zero or negligible, only position and velocity errors in the x direction need be considered. In other words, incremental positional commands in the lateral and vertical directions will be zero :

$$r_z[k] = r_y[k] = 0.$$

Recalling from Section 4.2, the robot arrives at a prescribed position with a delay of τ_a after the command is given. Note that τ_a is not necessarily an integral number of the sample period τ_s. In the case of PUMA 560, it is 3.4 approximately. Let λ be the nearest integer greater than τ_a/τ_s. (In the case of the PUMA 560 $\lambda = 4$). Path planning in the synchronization stage depends on the recognition of this delay, to provide the incremental commands accordingly.

With reference to fig.4.1, and at time $k+\lambda$, the robot position relative to its starting point is given as

$$p_{rx}[k+\lambda] = \sum_{i=1}^{k} r_x[i] + (\lambda\tau_s - \tau_a)v_c[k+\lambda] \qquad \ldots(4.10)$$

Similarly, the position of the workpiece at time $[k+\lambda]$ is

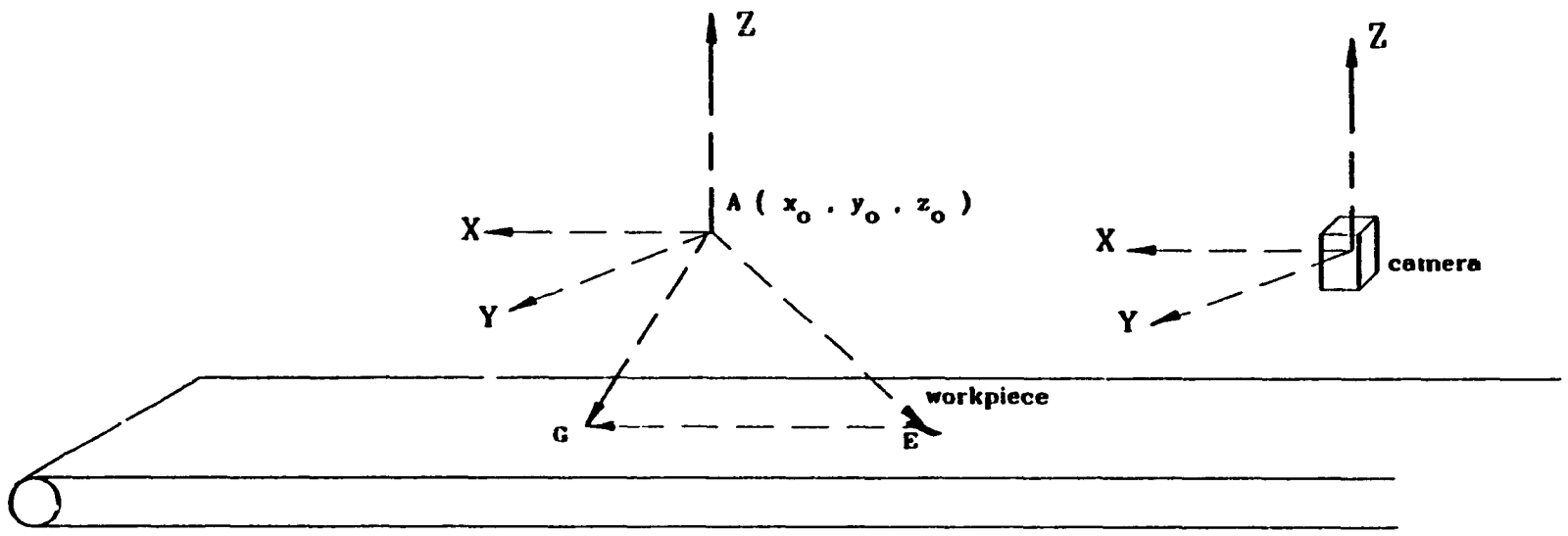
$$p_w[k+\lambda] = p_w[k] + \lambda\tau_s v_c[k] \qquad \ldots(4.11)$$

For perfect synchronization, these two positions are identical (see fig.4.6). Hence, the incremental command to give to the robot at time k is found to be

$$r_x[k] = p_w[k] + \tau_a v_c[k] - \Sigma r_x[i] \qquad \ldots(4.12)$$

by equating (4.10) and (4.11), and assuming constant conveyor speed with the λ periods starting from [k]:

$$v_c[k+\lambda] = v_c[k+\lambda-1] = \ldots = v_c[k]$$



AE -- $P_w[k]$ AG -- $p_r[k+\lambda]$ EG -- $\lambda \tau_s v_c[k]$

Fig.4.6 Relationship of Salient Position Vectors During the Synchronisation Phase (MT Model)

4.3.2 "Reduced Torque" Discrete Control Model

The linear path generates a large acceleration at the interception point B in fig.4.3. Large acceleration also introduces a large torque requirement due to the large derviative of the term dv_r/dx. In order to reduce the required torque , a *'reduced torque'* control model(fig.4.7) is proposed. The second control model consists of three path segments which are indicated by ADBCG. The first segment ADB is a quadrant of an ellipse. The second segment BC is a parabolic segment and CG is a linear segment. The first segment path is normal to the xy plane and the parabolic segment is on the yz plane. The path is designed such that the intersection of the elliptical segment and parabolic segment is right on the interception point B. The interception point B becomes the vertex of the first-segment ellipse and the second-segment parabola. The geometrical characteristic of the vertex gives a zero derivative, i.e. $dv_r/dx=0$. The zero acceleration reduces the joint torque requirement.

4.3.2.1 interception phase model

The first segment ADB of the path of RT model (fig.4.7) is the interception phase. This segment is described by a quadrant of an ellipse. An ellipse can be defined by three points - a centre and two vertices. In fig.4.7, the home position A is the minor vertex and the interception point B is the major vertex. Two normals from the points A and B intersects to form the centre of the ellipse I. The distance between the major vertex B and the centre of the ellipse I is defined to be the major axis 'a'. The distance between the home position A and the centre I is defined to be the minor axis 'b' of the ellipse. The path between the minor vertex and the major vertex is an quadrant of

an ellipse. Assuming a point D is a typical point on the path AB and the distance of ID is defined to be 'c'. The projection of the point D on the major axis is D'. The angle ∠DIB is φ. An elliptical path can be constructed under the following conditions :

$$\overline{ID'} = \overline{ID} \sin\varphi \qquad \ldots(4.13)$$

$$\overline{DD'} = \overline{ID} \cos\varphi \qquad \ldots(4.14)$$

Geometrically,

$$\overline{ID'} = \left[(x_d - x_o)^2 + (y_d - y_o)^2 \right]^{1/2}$$

$$\overline{DD'} = (z_d - z_m)$$

$$\overline{ID} = c$$

Equations (4.13) and (4.14) become,

$$\left[(x_d - x_o)^2 + (y_d - y_o)^2 \right] = c \sin\varphi$$

$$(z_d - z_m) = c \cos\varphi$$

Geometrically, the interception point B has zero acceleration, i.e. $dv_r/dx = 0$. When the robot end-effector is at the home position A, the elliptical path can be determined if the interception point B (x_m, y_m, z_m) is known. The value y_m is obtained from the image processing and it is invariant to the conveyor position. The z_m is designed by the user w.r.t. specific industrial applications. Therefore, the remaining unknown x_m has to be solved to determine the elliptical path.

Theoretically, an elliptical path can be constructed such that its vertex is right on the interception point B, however, the second parabolic segment becomes practically impossible. It is because the robot end-effector is forced to synchronise with the moving workpiece at point B immediately. Its velocity profile is totally governed by the conveyor velocity. Therefore, although the elliptical interception phase provides a zero acceleration at point B, the sudden change of the robot velocity from zero to full conveyor velocity also generates a large torque requirement. Thus, the control model is modified as shown in fig.4.8. A virtual interception point B' is introduced to provide the solution. The virtual interception B' is ahead of the actual interception point B by a distance 's'. The position of the point B' is defined to be $(x_m + s, y_m, z_m)$ and the term $(x_m + s)$ is defined to be x_v. When the robot arrives at the virtual interception point B', the workpiece is at the point B'' which is right underneath the point B. The robot velocity is zero at the point B' and it starts to accelerate until it synchronise with the workpiece at the point C. This stage (from B' to C) is called the transition phase.

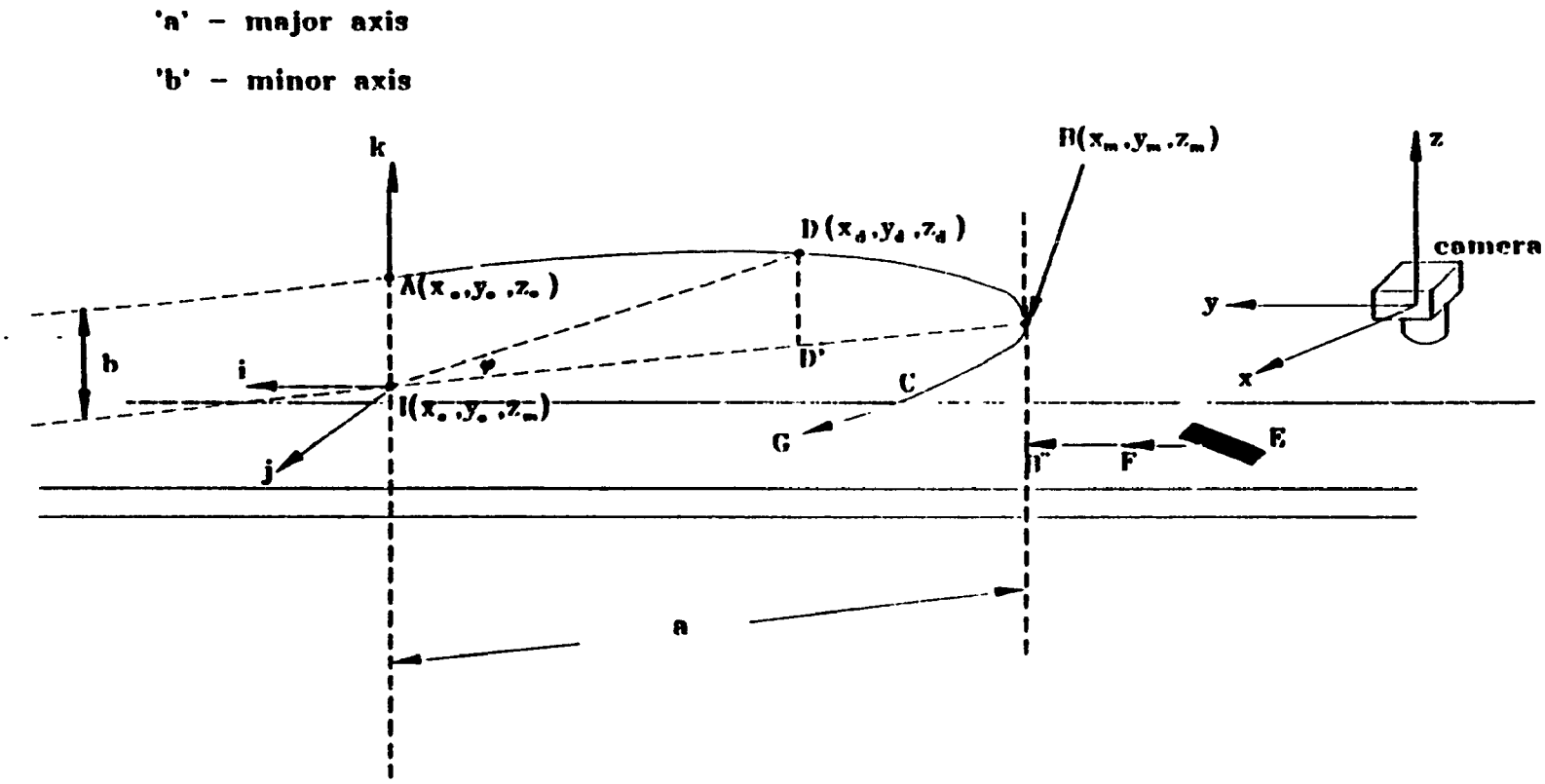


Fig.4.7 Centre and Vertices of the Elliptical Segment

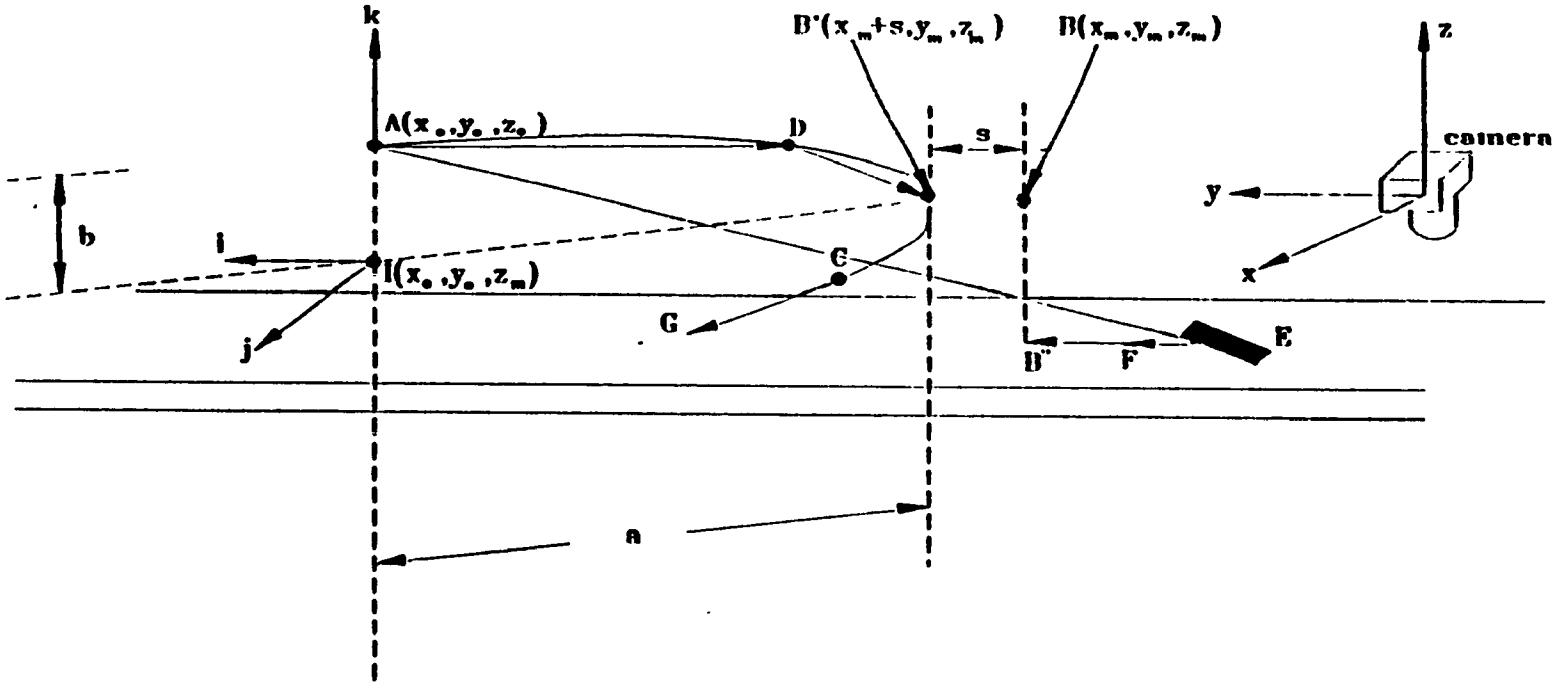


Fig.4.8 Path of the Reduced Torque Model with Virtual Interception Point

Similar to the MT model, the displacement is defined w.r.t. the starting position : $p_r[k]$, the accumulative positional command at time k. Assuming the point D is an intermediate position of the end-effector on the elliptical path at time k, while the workpiece is given by the measured position $P_w[k]$ plus a distance transversed by the conveyor at the measured speed $v_c[k]$ for the duration τ_d (assuming $v_c[k]$ being constant during τ_d). An appropriate robot velocity $v_r[k]$ is maintained constant for the next m[k] periods of time, it is conceivable that the robot will intercept the workpiece at the virtual intercept position B'. Let S[k] to be the length of the remaining elliptical path from the point D to the virtual interception point B'. The xyz components of the S[k] are described by $s_x[k]$, $s_y[k]$ and $s_z[k]$. Let n be the total number of iterations taken for the robot to move from the home position to the virtual interception point B'. Thus,

$$n = m[k] + k$$

Referring to fig.4.8, vectorially speaking,

$$\mathbf{AD} + \mathbf{DB'} = \mathbf{AE} + \mathbf{EF} + \mathbf{FB''} + \mathbf{si}$$

$$\text{i.e. } p_r[k-1] + \sum_{i=k}^{n} [s_x[i]\mathbf{i} + s_y[i]\mathbf{j} + s_z[i]\mathbf{k}$$

$$= P_w[k] + v_c[k]\,\tau_d\mathbf{i} + v_c[k]\,m[k]\,\tau_s\mathbf{i} + \mathbf{si} \qquad \ldots(4.15)$$

Referring to equations (4.2) and (4.3), equation (4.15) becomes,

$$\left(\Sigma\, r_x[i]\mathbf{i} + \Sigma r_y[i]\mathbf{j} + \Sigma r_z[i]\mathbf{k} \right) + \left(\sum_{i=k}^{n} s_x[i]\mathbf{i} + \sum_{i=k}^{n} s_y[i]\mathbf{j} + \sum_{i=k}^{n} s_z[i]\mathbf{k} \right)$$

$$= \left(\sum_{i=1}^{k} v_c[i]\tau_s - x_o \right)\mathbf{i} + \left(Y_c - y_o \right)\mathbf{j} + \left(Z_d - z_o \right)\mathbf{k} + v_c[k]\tau_d\mathbf{i} + v_c[k]m[k]\tau_s\mathbf{i}$$

$$+ \mathbf{si} \qquad \ldots(4.16)$$

By equating the coefficient of the vectors i, j and k in (4.16),

$$\Sigma r_x[i] + \sum_{i=k}^{n} s_x[i] = \left(\sum_{i=k}^{n} v_c[i]\tau_s - x_o\right) + v_c[k]\tau_d + v_c[k]m[k]\tau_s + s \quad \ldots(4.17)$$

$$\Sigma r_y[i] + \sum_{i=k}^{n} s_y[i] = Y_c - y_o \quad \ldots(4.18)$$

$$\Sigma r_z[i] + \sum_{i=k}^{n} s_z[i] = Z_d - z_o \quad \ldots(4.19)$$

Using $C_1[k]$, $C_2[k]$ and $C_3[k]$ to replace the terms in equations (4.17), (4.18) and (4.19),

$$C_1[k] = \sum_{i=1}^{k} v_c[k]\tau_s - x_o + v_c[k]\tau_d + s \quad \ldots(4.20)$$

$$C_2[k] = (Y_c - y_o) - \Sigma r_y \quad \ldots(4.21)$$

$$C_3[k] = (Z_d - z_o) - \Sigma r_z \quad \ldots(4.22)$$

Equations (4.17), (4.18) and (4.19) become,

$$\Sigma r_x[i] + \sum_{i=k}^{n} s_x[i] = v_c[k]m[k]\tau_s + C_1[k] \quad \ldots(4.23)$$

$$\sum_{i=k}^{n} s_y[i] = C_2[k] \quad \ldots(4.24)$$

$$\sum_{i=k}^{n} s_x[i] = C_3[k] \quad \ldots(4.25)$$

Solving $m[k]\tau_s$ from equation (4.23),

$$m[k]\tau_s = \frac{\Sigma r_x[i] + \sum_{i=k}^{n} s_x[i] - C_1[k]}{v_c[k]} \quad \ldots(4.26)$$

Assuming that the robot velocity is maintained to be constant, say V_r, from the point D to the virtual interception point B'. The length of the remaining elliptial path can be evaluated by the following equation,

$$\sum_{i=k}^{n}[s_x[i] + s_y[i] + s_z[i]\ ] \cong m[k]\ \tau_s V_r \qquad \ldots(4.27)$$

The s_x and m[k] are two unknowns in equations (4.26) and (4.27). The objective of the following formulation is to solve the value of m[k] and $s_x[k]$. If m[k] is greater than 1, incremental positional commands are then calculated to direct the robot end-effector to move in an elliptical path.

Combining equations (4.26) and (4.27) gives,

$$\sum_{i=k}^{n}[s_x[i]+s_y[i]+s_z[i]] = \frac{V_r}{V_c[k]}\ [\Sigma\ r_x[i] + \sum_{i=k}^{n} s_x[i]-C_1[k]] \qquad \ldots(4.28)$$

The left side of the equation (4.28) represents the remaining path of an elliptical path which has to be solved by evaluating the elliptical integral shown in Appendix E.

Let the length of the elliptical path S[k] travels from kth iteration to nth iteration and the angle $\angle DIB'$ be φ. The distances between the points $A(x_o, y_o, z_o)$ and $B'(x_v, y_m, z_m)$ be x_{vo}, y_{mo} and z_{mo} and they are evaluated by the following equations,

$$x_{vo} = x_v - x_o = \Sigma\ r_x[i] + \sum_{i=k}^{n} s_x[i]$$

$$y_{mo} = y_m - y_o = Y_c - y_o$$

$$z_{mo} = z_m - z_o = Z_d - z_o$$

Referring to reference [12], the length of an elliptical segment can be expressed by the following equation,

$$S[k] = (x_{vo}^2 + y_{mo}^2)^{1/2}\ E\left[\left[\frac{x_{vo}^2 + y_{mo}^2 - z_{mo}^2}{x_{vo}^2 + y_{mo}^2}\right]^{1/2}, \varphi\right]$$

Thus, equation (4.28) becomes,

$$(x_{vo}^2 + y_{mo}^2)^{1/2} E \left[\left[\frac{x_{vo}^2 + y_{mo}^2 - z_{mo}^2}{x_{vo}^2 + y_{mo}^2} \right]^{1/2}, \varphi \right] = \frac{V_r}{V_c[k]} \left[x_{vo} - C_1[k] \right] \quad ...(4.29)$$

Let

$$\alpha = \left[\frac{x_{vo}^2 + y_{mo}^2 - z_{mo}^2}{x_{vo}^2 + y_{mo}^2} \right]^{1/2}$$

Thus,

$$E(\alpha, \varphi) = E \left[\left[\frac{x_{vo}^2 + y_{mo}^2 - z_{mo}^2}{x_{vo}^2 + y_{mo}^2} \right]^{1/2}, \varphi \right] \quad ...(4.30)$$

$E(\alpha, \varphi)$ is a non-linear function. A closed form solution cannot be obtained easily. A linearisation technique will be formulated to linearise the $E(\alpha, \varphi)$ by several segments of linear functions. Finally, the equation (4.29) can be expressed approximately in a form of a second order quadratic equation.

4.3.2.1.1 elliptical integral linearisation

The elliptical integral function $E(\alpha, \varphi)$ is shown in the Appendix E fig.E.2 and fig.E.3. The function $E(\alpha, \varphi)$ is a non-linear function which must be solved by numercial techniques [12]. Therefore, it is less likely to be applicable in real-time path planning. However, the numerical solution of the elliptical integral does not have to be calculated from scratch. The numerical solution shown in the Table of Functions by Jahnke and Emde[43] (Table E.1 in Appendix E) can be interpolated to approximate the path data in real-time. Referring to fig.E.2 in Appendix E, the values of $E(\alpha, \varphi)$ are very linear if the angle φ is less than 60°. It starts to be nonlinear at

the angle greater than 60°. The methodology to solve the solution in equation (4.29) is to linearise the non-linear curves by several segments of linear functions, e.g. straight lines as shown in fig.4.9. Let the number of segments be n_s. During the process of real-time spatial path generation, each segment gives a solution of $s_x[k]$. Therefore, there are totally n_s solutions. The value which has minimum error will be chosen as the solution. The detailed formulation will be described latter in this section. On the other hand, one can also use other functions, such as polynomials, spline, etc. to approximate the function $E(\alpha,\varphi)$. However, if a high order function, is used, iterations are required in solving the equation (4.29) to obtain $s_x[k]$. Therefore, order higher than one is not recommended for on-line application. If a straight line function is used, the equation (4.29) can be solved to obtain a closed form solution. Therefore, several segments of best-fit lines will be used to approximate the elliptical integral function in the following formulation. To investigate the error of the approximation, several trials have been done to approximate the curve at angle φ=90°, i.e. the most non-linear curve, with 1 to 4 segments (fig.4.9). The accuracy of the approximation is based on the root-mean-square errors which are calculated by the following equation,

$$e^2 = \sum_{i=1}^{n} (f(\theta_i) - g(\theta_i))^2$$

where

$f(\theta)$ be the elliptical integral computed by the numerical method

$g(\theta)$ be the elliptical integral approximated by the best-fit lines

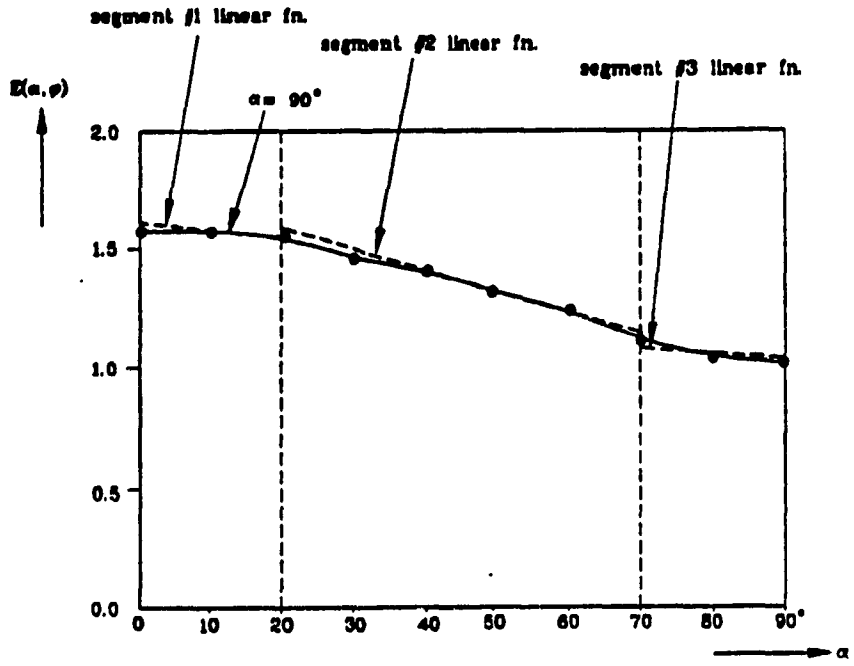


Fig. 4.9 Linearisation of the Elliptical Integral

TABLE 4.1 ERROR OF THE LINEARISATION OF THE ELLIPTICAL INTEGRAL

# of segment	$\sqrt{(e^2/n)}$			
1	$0 \le \theta \le 90^\circ$			
% error	7.7%-12.1%			
2	$0 \le \theta \le 50^\circ$	$55^\circ \le \theta \le 90^\circ$		
% error	1.38%-1.66%	1.1%-1.47%		
3	$0 \le \theta \le 20^\circ$	$25^\circ \le \theta \le 70^\circ$	$75^\circ \le \theta \le 90^\circ$	
% error	0.3%-0.32%	1.18%-1.8%	0.73% 0.82%	
4	$0^\circ \le \theta \le 20^\circ$	$25^\circ \le \theta \le 50^\circ$	$55^\circ \le \theta \le 80^\circ$	$85^\circ \le \theta \le 90^\circ$
% error	0.3%-0.32%	0.45%-0.52%	0.34%-0.43%	0.33%-0.34%
4	$0^\circ \le \theta \le 20^\circ$	$25^\circ \le \theta \le 55^\circ$	$60^\circ \le \theta \le 80^\circ$	$85^\circ \le \theta \le 90^\circ$
% error	0.3%-0.32%	0.54%-0.66%	0.29%-0.36%	0.33%-0.34%

There is a large percentage error in the one segment group. The more reasonable accuracy may be the groups having three or four segments. One can try with higher number of segments to improve the accuracy. However, larger number of segments requires greater number of iterations in real-time computation. One can use the following formula as reference in selecting the number of segments.

Let n_s be the maximium number of segments

Let t_{co} be the time to compare the errors and decide which segments should be used

Let t_{io} be the time for the I/O handshake with the robot controller

Let t_{seg} be the time to calculate the path data of one segment

Let t_{tr} be the communication overhead for one transputer talking with another transputer

The total time spending on the path calculation and communication overhead should be less than or equal to the sampling time. Therefore,

$$n_s t_{seg} + t_{io} + t_{tr} + t_{co} = \tau_s$$

and the allowable maximum number of segments is

$$n_s = \frac{\tau_s - t_{io} - t_{tr} - t_{co}}{t_{seg}}$$

The following steps are the linearisation technique to solve the elliptical path in real-time and it is assumed that three segments are used :

<u>step 1</u> : Divides the curves in Appendix E into three linearised parts (0° to θ_1, θ_1 to θ_2, θ_2 to 90°) as shown in fig.4.9. The function $E(\alpha, \varphi)$ is represented by three linear best-fit equations as follows :

$$E_1(\alpha,\ \varphi) = m_1(\varphi)\alpha + c_1(\varphi) \qquad \text{where } 0^\circ < \alpha \le \theta_1$$

$$E_2(\alpha,\ \varphi) = m_2(\varphi)\alpha + c_2(\varphi) \qquad \text{where } \theta_1 < \alpha \le \theta_2$$

$$E_3(\alpha,\ \varphi) = m_3(\varphi)\alpha + c_3(\varphi) \qquad \text{where } \theta_2 < \alpha \le 90^\circ$$

...(4.31)

where $$\alpha = \left[\frac{x_{vo}^2 + y_{mo}^2 - z_{mo}^2}{x_{vo}^2 + y_{mo}^2} \right]^{1/2}$$

The values of $m_1(\varphi)$, $m_2(\varphi)$ and $m_3(\varphi)$ are the gradients of each linear segment and the $c_1(\varphi)$, $c_2(\varphi)$ and $c_3(\varphi)$ are the linear constants. They are represented by m_1, m_2, m_3, c_1, c_2 and c_3 in the following formulations. By assigning the point B′ in each three segments E_1, E_2 and E_3. The values of x_{vo1}, x_{vo2} and x_{vo3} can be obtained. The one which has the minimum error is considered to be the solution .

<u>step 2</u> : Substitute the equation (4.31) into the non-linear equation (4.29) to generate the following equation (4.32),

$$\left(x_{voi}^2 + y_{mo}^2 \right)^{1/2} \left[m_i \left[\frac{x_{voi}^2 + y_{mo}^2 - z_{mo}^2}{x_{voi}^2 + y_{mo}^2} \right]^{1/2} + c_i \right] \cong \frac{V_r}{v_c[k]} \left[x_{voi} - C_i[k] \right]$$

...(4.32)

where i = 1, 2 or 3

<u>step 3</u> : Simplify the order of the equation (4.32) into second order by choosing certain design conditions.

One can found that equation (4.32) cannot be solved easily because of the square and square-root terms. However, by choosing certain design conditions, the high order terms can be eliminated and

the equation (4.32) can be simplified in the form of a second-order quadratic equiation as the following form,

$$x_{voi}^2 + BB_i\, x_{voi} + CC_i = 0$$

By expanding equation (4.32), it becomes,

$$m_i\left(x_{voi}^2 + y_{mo}^2 - z_{mo}^2 \right)^{1/2} + c_i \left(x_{voi}^2 + y_{mo}^2 \right)^{1/2} - \frac{V_r}{v_c[k]}\, x_{voi} = - \frac{V_r}{v_c[k]} C_i[k]$$

$$m_i x_{voi}\left[1 + \frac{y_{mo}^2 - z_{mo}^2}{x_{voi}^2} \right]^{1/2} + c_i x_{voi}\left[1 + \frac{y_{mo}^2}{x_{voi}^2} \right]^{1/2} - \frac{V_r}{v_c[k]}\, x_{voi} = - \frac{V_r}{v_c[k]}\, C_i[k]$$

...(4.33)

To simplify the equation (4.33), y_{mo} can be designed to be approximately equal to z_{mo} such that the term of

$$\left[\frac{y_{mo}^2 - z_{mo}^2}{x_{voi}^2} \right]^{1/2} = 0$$

Another term $\left[1 + \frac{y_{mo}^2}{x_{voi}^2}\right]^{1/2}$ can be approximated by a low order Taylor series

Expanding the term $\left[1 + \frac{y_{mo}^2}{x_{voi}^2}\right]^{1/2}$ into series which gives

$$\left[1 + \frac{y_{mo}^2}{x_{voi}^2} \right]^{1/2} = 1 + \frac{1}{2}\left[\frac{y_{mo}^2}{x_{voi}^2} \right] + \frac{1/2 * -1/2}{2!}\left[\frac{y_{mo}^2}{x_{voi}^2} \right]^2 +$$

$$+ \frac{1/2 * -1/2 * -3/2}{3!}\left[\frac{y_{mo}^2}{x_{voi}^2} \right]^3 + \ldots.$$

$$= 1 + \frac{1}{2}\left[\frac{y_{mo}}{x_{vo1}}\right]^2 - 0.125\left[\frac{y_{mo}}{x_{vo1}}\right]^4 + 0.0625\left[\frac{y_{mo}}{x_{vo1}}\right]^6 + \ldots. \qquad \ldots(4.34)$$

The virtual interception point B'(x_y, y_m, z_m) is designed to be closer to the upstream such that the ratio y_{mo} / x_{vo} is less than 1, equation (4.34) can be approximated to be

$$\left[1 + \frac{y_{mo}^2}{2}\right]^{1/2} \cong 1 + \frac{1}{2}\left[\frac{y_{mo}}{x_{vo1}}\right]^2 \qquad \ldots(4.35)$$

step 4 : Substitute the simplified expression (4.35) into (4.33) to approximate (4.33) into a second-order quadratic equation,

$$m_i x_{vo1} + c_i x_{vo1}\left[1 + \frac{1}{2}\frac{y_{mo}^2}{x_{vo1}^2}\right] - \frac{V_r}{v_c[k]} x_{vo1} = -\frac{V_r}{v_c[k]} C_i[k] \qquad \ldots(4.36)$$

$$\left(m_i + c_i - \frac{V_r}{v_c[k]}\right) x_{vo1} + \frac{c_i y_{mo}^2}{2 x_{vo1}} = -\frac{V_r}{v_c[k]} C_i[k]$$

$$\left(m_i + c_i - \frac{V_r}{v_c[k]}\right) x_{vo1}^2 + \frac{V_r}{v_c[k]} C_i[k]\, x_{vo1} + \frac{c_i y_{mo}^2}{2} = 0$$

$$x_{mo1}^2 + \frac{V_r\ C_i[k]}{m_i v_c[k] + c_i v_c[k] - V_r} x_{vo1} + \frac{c_i y_{mo}^2}{2}\left[\frac{v_c[k]}{m_i v_c[k] + c_i v_c[k] + V_r}\right] = 0$$

The above equation is a second order quadratic equation and the coefficients for each segment i are defined as the follows,

$$BB_i = \frac{V_r\ C_i[k]}{m_i v_c[k] + c_i v_c[k] - V_r}$$

$$CC_i = \frac{c_i\, y_{mo}^2}{2}\left[\frac{v_c[k]}{m_i v_c[k]+c_i v_c[k]+V_r}\right]$$

The solution of the virtual interception point position is given by,

$$x_{voi} = \frac{-BB_i + (BB_i^2 - 4\,CC_i)^{1/2}}{2} \qquad \ldots(4.37)$$

By substituting the corresponding values of m_1, m_2, m_3 and c_1, c_2, c_3, x_{vo1}, x_{vo2} and x_{vo3} can be obtained.

<u>step 5</u> : Since there are three segment (i=1,2,3) , thus there are three possible values of $x_{vo}(x_{vo1},x_{vo2},x_{vo3})$. Recalling the equation (4.32),

$$(x_{voi}^2 + y_{mo}^2)^{1/2}\left[m_i\left[\frac{x_{voi}^2 + y_{mo}^2 - z_{mo}^2}{x_{voi}^2 + y_{mo}^2}\right]^{1/2} + c_i \right] - \frac{V_r}{v_c[k]}\left[x_{voi} + C_i[k]\right] = 0$$

The values of x_{vo1}, x_{vo2} and x_{vo3} are substituted in equation (4.32). The one which is closest to zero will be considered as the solution.

4.3.2.1.2 elliptical path position commands

The previous section has evaluated the position of the virtual interception point $B'(x_{vo}, y_m, z_m)$. At time k, if the robot end-effector is at the point D, it can be determined how many iterations still have to go to arrive the virtual interception point B'. The number of iterations required to arrive the point B' is by evaluating the m[k] in equation (4.26). Recalling equation (4.26),

$$m[k]\tau_s = \frac{\Sigma\, r_x[i] + \sum_{i=k}^{n} s_x[i] - C_i[k]}{v_c[k]}$$

and $$x_{vo} = \Sigma\, r_x[i] + \sum_{i=k}^{n} s_x[i]$$

$$m[k]\tau_s = \frac{x_{vo} - C_i[k]}{v_c[k]}$$

If m[k] is greater than 1, current position should be updated and furthur position commands should also be calculated to direct the robot end-effector. This section will also describe the formulation to calculate the position commands in the x, y and z directions with a fixed robot velocity V_r such that the robot end-effector can be directed in a prescribed elliptical path segment.

Let the robot coordinate system to be represented by i, j and k. Assuming the elliptical path shown in fig.4.10 is along a plane inclined an angle ϕ w.r.t. the xz plane. The inclined plane has the coordinate system i_ϕ and k_ϕ which related to the robot cartesian coordinate i, k as the followings,

$$i_\phi = \cos\phi\, i + \sin\phi\, j$$
$$k_\phi = k \qquad \ldots(4.38)$$

Referring to fig.4.8, it can be defined that

$$\phi = \tan^{-1}\left[\frac{y_{mo}}{x_{vo}}\right]$$

The centre of the ellipse is $I(x_o, y_o, z_m)$ and two vertices are $A(x_o, y_o, z_o)$ and $B'(x_v, y_m, z_m)$. Assuming the robot end-effector is at the point D. Let $q_r[k]$, i.e. ID is the current position vector of the robot effector w.r.t. the centre. The $\Delta q_r[k]$ is the incremental position vector such that

$$q_r[k+1] = q_r[k] + \Delta q_r[k] \qquad \ldots(4.39)$$

At kth iteration, let the angle ∠AID be $\beta[k]$. As the robot starts at the home position A travelling to the point B', β varies from 90 degree (k=0) to 0 degree (k=n). The angle between the vectors $q_r[k]$ and $q_r[k+1]$ is $\Delta\beta[k]$. The major axis of the elliptical path be 'a'(IB') and the minor axis of the elliptical path be 'b' (IA). As shown in fig.4.10,

$$a = \left[x_{vo}^2 + y_{mo}^2 \right]^{1/2}$$

$$b = z_{mo}$$

Geometrically, an ellipse can be described in the vector form as the following equation,

$$q_r[k] = a \cos \beta[k]\ i_\phi + b \sin \beta[k]\ k_\phi \qquad \ldots(4.40)$$

Differentiating equation (4.40) gives the equation (4.41),

$$\Delta q_r[k] = -a \sin \beta[k]\ \Delta\beta[k]\ i_\phi + b \cos \beta[k]\ \Delta\beta[k]\ k_\phi \qquad \ldots(4.41)$$

Transforming the coordinate in equation (4.41) into robot coordinate by substituting equation (4.38) into(4.41),

$$\Delta q_r[k] = -a \sin \beta[k] \cos \phi\ \Delta\beta[k]\ i - a \sin \beta[k] \sin \phi\ \Delta\beta[k]\ j + b \cos \beta[k]\ \Delta\beta[k]\ k \qquad \ldots(4.42)$$

The incremental position command of the elliptical path can be actually splitted into three components,

$$\Delta q_r[k] = s_x[k]\ i + s_y[k]\ j + s_z[k]\ k \qquad \ldots(4.43)$$

Therefore, by equating the coefficients in equations (4.42) and (4.43), the incremental position commands of the elliptical path are

$$s_x[k] = - a \sin \beta[k] \cos \phi\ \Delta\beta[k] \qquad \ldots(4.44)$$

$$s_y[k] = - a \sin \beta[k] \sin \phi \, \Delta\beta[k] \qquad \ldots(4.45)$$

$$s_z[k] = b \cos \beta[k] \, \Delta\beta[k] \qquad \ldots(4.46)$$

The only unknown in the equations (4.44), (4.45) and (4.46) is the $\Delta\beta[k]$. The following formulation is to evaluate the $\Delta\beta[k]$ by equating the components of the prescribed robot velocity V_r. Differentiating the incremental position commands gives,

$$\dot{\Delta q_r}[k] = \dot{s}_x[k] \, i + \dot{s}_y[k] \, j + \dot{s}_z[k] \, k \qquad \ldots(4.47)$$

The velocity of the incremental position commands can be evaluated through dividing the incremental displacement by the sampling period τ_s. Therefore,

$$|\dot{\Delta q_r}[k]| = \left| \frac{\Delta q_r[k]}{\tau_s} \right|$$

$$\dot{s}_x[k] = \frac{s_x[k]}{\tau_s}$$

$$\dot{s}_y[k] = \frac{s_y[k]}{\tau_s}$$

$$\dot{s}_z[k] = \frac{s_z[k]}{\tau_s}$$

The robot end-effector velocity V_r should be equal to the total end-effector velocity $\dot{\Delta q_r}[k]$, therefore

$$V_r = \frac{s_x[k]}{\tau_s} + \frac{s_y[k]}{\tau_s} + \frac{s_z[k]}{\tau_s} \qquad \ldots(4.48)$$

Substitute equations (4.44), (4.45) and (4.46) into equation (4.48), equation (4.48) becomes,

$$\frac{a^2\sin^2\beta[k]\cos^2\phi\,\Delta\beta^2[k]}{\tau_s^2} + \frac{a^2\sin^2\beta[k]\cos^2\phi\,\Delta\beta^2[k]}{\tau_s^2} + \frac{b^2\cos^2\beta[k]\,\Delta\beta^2[k]}{\tau_s^2} = V_r^2 \qquad \ldots(4.49)$$

$$\Delta\beta[k] = \frac{V_r\,\tau_s}{\{a^2\sin^2\beta[k]\cos^2\phi + a^2\sin^2\beta[k]\sin^2\phi + b^2\cos^2\beta[k]\}} \qquad \ldots(4.50)$$

By subsituting the $\Delta\beta[k]$ in equation (4.50) into equations (4.44), (4.45) and (4.46), the incremental position commands of the elliptical path are expressed as follows,

$$s_x[k] = \frac{-a\sin\beta[k]\cos\phi\,V_r\,\tau_s}{\{a^2\sin^2\beta[k]\cos^2\phi + a^2\sin^2\beta[k]\sin^2\phi + b^2\cos^2\beta[k]\}}$$

$$s_y[k] = \frac{-a\sin\beta[k]\sin\phi\,V_r\,\tau_s}{\{a^2\sin^2\beta[k]\cos^2\phi + a^2\sin^2\beta[k]\sin^2\phi + b^2\cos^2\beta[k]\}}$$

$$s_z[k] = \frac{b\cos\beta[k]\,V_r\,\tau_s}{\{a^2\sin^2\beta[k]\cos^2\phi + a^2\sin^2\beta[k]\sin^2\phi + b^2\cos^2\beta[k]\}}$$

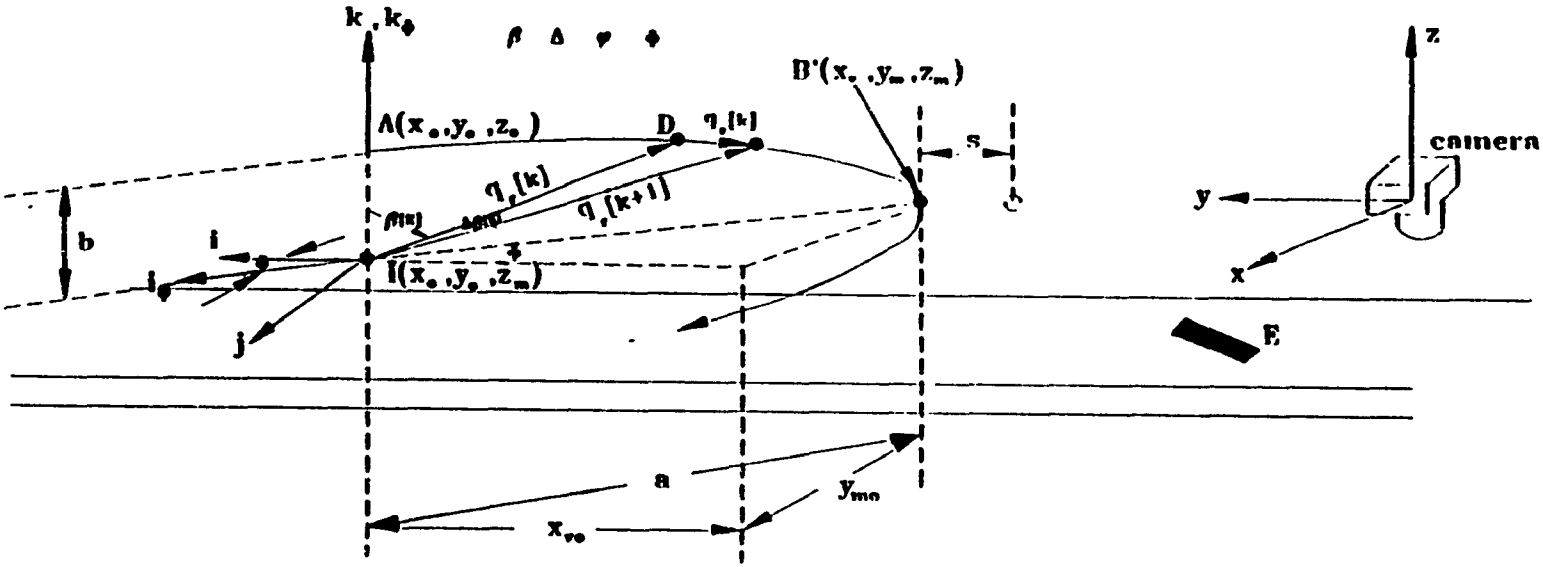


Fig.4.10 Position Vectors and Commands of the RT Model

4.3.2.2 transition and synchronisation phase model

It has been discussed in section 4.3.2.1 that an elliptical path can be constructed such that its vertex is right on the interception point B. However, the zero velocity at B becomes practically impossible because the robot end-effector is forced to synchronise with the moving workpiece at point B immediately. Therefore, although the elliptical interception phase provides a zero velocity at the point B, the sudden change of the robot velocity from zero to full conveyor velocity also generates a large torque requirement. Thus, a virtual interception point B′ is introduced and it is designed to be ahead of the interception point B by a distance 's'. In fig.4.11, when the robot end-effector is at the point B′, the workpiece will be at the point B′′ which is right underneath the point B. Starting with a zero velocity at point B′, the robot will accelerate and eventually synchronise with the moving workpiece at the point C , while the workpiece is at point C′. The period for the robot to travel from point B′ to C is called the transition phase. Starting from the interception point B′, only the x-position of the end-effector is taken into consideration since the velocity in y-direction is zero and velocity in z-direction is set to be constant. The velocity profile in the transition phase can be linear, parabolic, spline, etc. For the sake of the simplicity, a linear velocity profile shown in fig.4.12 will be formulated and implemented.

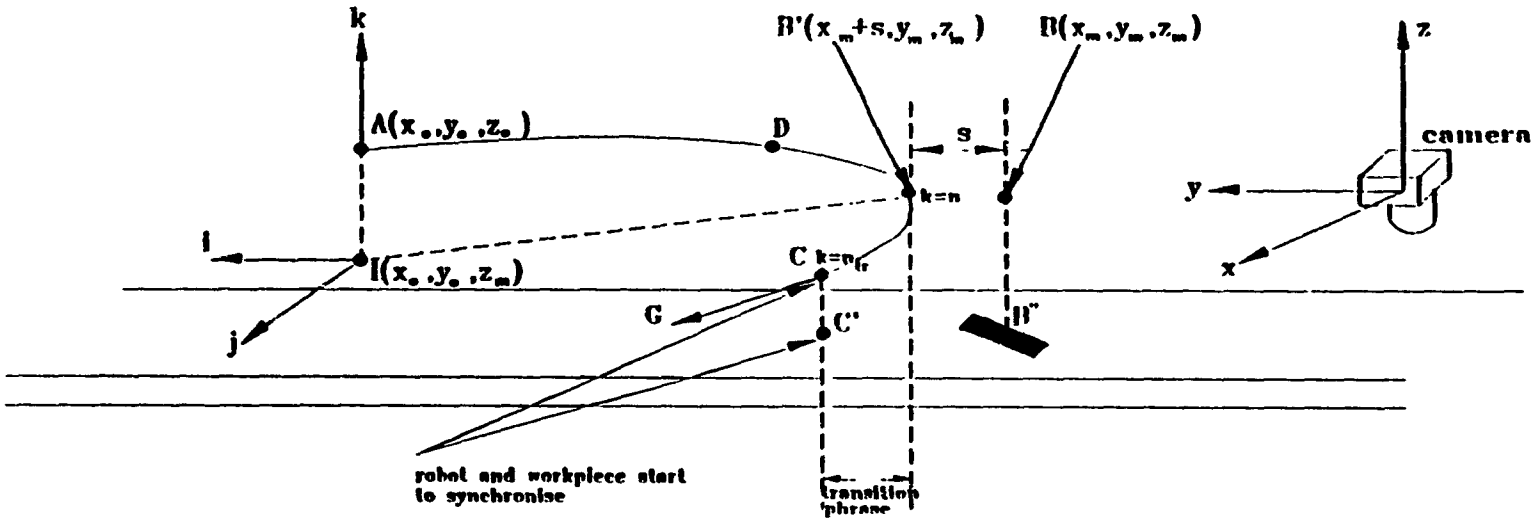


Fig.4.11 TRansition Phase of the RT Model

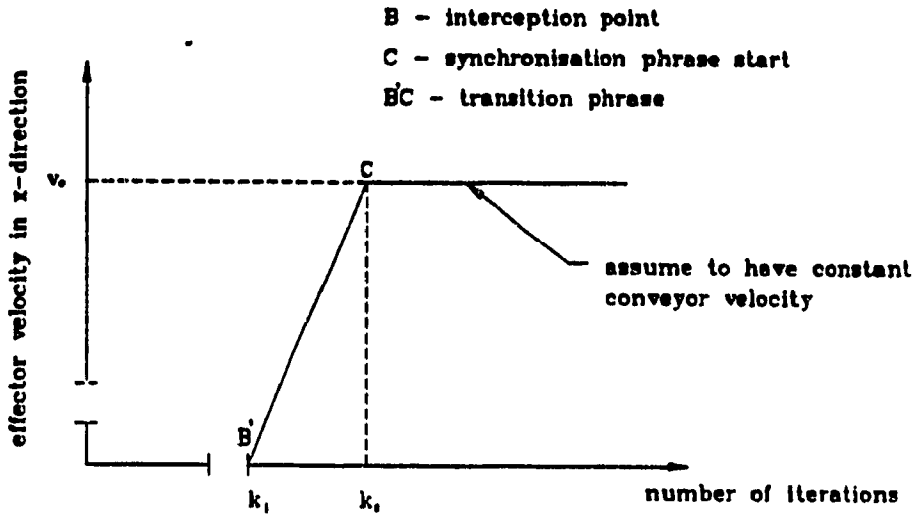


Fig.4.12 Linear Velocity Profile of the Transition Phase

Let n be the number of iterations required for the robot travelling from the home position A to the virtual inteception point B'

Let n_{tr} be the number of iterations at the end of the transition phase

Let $v_{rx}[k]$ be the x-component of the robot velocity at kth iteration

Let $a_{rx}[k]$ be the x-component of the robot acceleration at kth iteration

The displacement of the workpiece from B″ to C′ should be equivalent to the x-component of the robot end-effector displacement from B′ to C plus the distance 's'. Supposing a linear velocity profile shown in fig.4.12 is assumed in the transition phase. By equating the displacements, equation (4.51) is obtained.

$$\sum_{i=n}^{n_{tr}} v_{rx}[i]\ \tau_s + s = \sum_{i=n}^{n_{tr}} v_c[i]\ \tau_s \qquad \ldots(4.51)$$

In addition, under the assumption that the velocity of the workpiece does not change from nth to n_{tr}th iterations, the velocity $v_c[k]$ is symbolised to be a constant V_c. Therefore, the end-efffector velocity can be expressed as in equation (4.52).

$$\sum_{i=n}^{n_{tr}} v_{rx}[i]\ \tau_s = \frac{V_c\ (n_{tr} - n\ )\ \tau_s}{2} \qquad \ldots(4.52)$$

The workpiece velocity can be expressed as,

$$\sum_{i=n}^{n_{tr}} v_c[i]\ \tau_s = V_c\ (n_{tr} - n\ )\ \tau_s \qquad \ldots(4.53)$$

Substitute equation (4.52) and (4.53) into (4.51),

$$\frac{V_c\ (n_{tr} - n\ )\ \tau_s}{2} + s = V_c\ (\ n_{tr} - n\ )\ \tau_s \qquad \ldots(4.54)$$

$$\therefore \qquad (\ n_{tr} - n\ )\ \tau_s = \frac{2 * s}{V_c} \qquad \ldots(4.55)$$

For a linear velocity profile,

$$V_c = a_{rx}\ [n_{tr} - n]\ \tau_s$$

Therefore,

$$a_{rx} = \frac{V_c}{[n_{tr} - n]\ \tau_s}$$

The acceleration of the robot end-effector can be expressed by,

$$a_{rx} = \frac{V_c^{\,2}}{2\,s} \qquad \ldots(4.56)$$

The robot velocity at kth iteration is,

$$\therefore \quad v_{rx}[k] = \frac{V_c^{\,2}}{2\,s}\,(k - n)\,\tau_s \qquad \ldots(4.57)$$

The velocity relationship in equation (4.57) directs the robot to move a parabolic segment. For k which is greater than n_{tr}, the model is similiar to what has been discussed in section 4.3.1.2.

4.4 PERFORMANCE

Three performance parameters of the two control models will be compared. These parameters are :

. productive space (section 4.4.1)

. torque transient (section 4.4.2)

. synchronisation error (section 4.4.3)

The shaft encoders of all six joints are sampled and decoded by a joint data acquisition circuitry in the hierarchical controller. The six input joint angles are then transformed into the cartesian position by a PUMA forward kinematic algorithm shown in Appendix H to reproduce the experimental trajectory of the robot.

4.4.1 Productive Space Comparsion

The two limits of the robot working space may be called the "upstream limit" and the "downstream limit" shown in fig.4.13. The upstream limit has a distance of L_1 and the downstream limit has a distance of $(L_1 + L_e)$ measuring from the camera station. The workspace is then equal to be the distance of L_e. For the case of the MT model, let d_{m1} be the distance of the interception point measuring from the camera station and d_{s1} be the maximum allowable productive space.

$$d_{s1} = L_1 + L_e - d_{m1} \qquad \ldots(4.58)$$

For the case of the RT model, let d_{m2} be the distance of the interception point measuring from the camera station. Let d_t be the travelling distance for the transition phrase and d_{s2} be the maximum allowable productive space.

$$d_{s2} = L_1 + L_e - d_{m2} - d_t \qquad \ldots(4.59)$$

AB – interception phrase

BC – synchronisation phrase (approaching)

CD – synchronisation phrase (productive)

A – home position

B – intercept point

C – arrival position

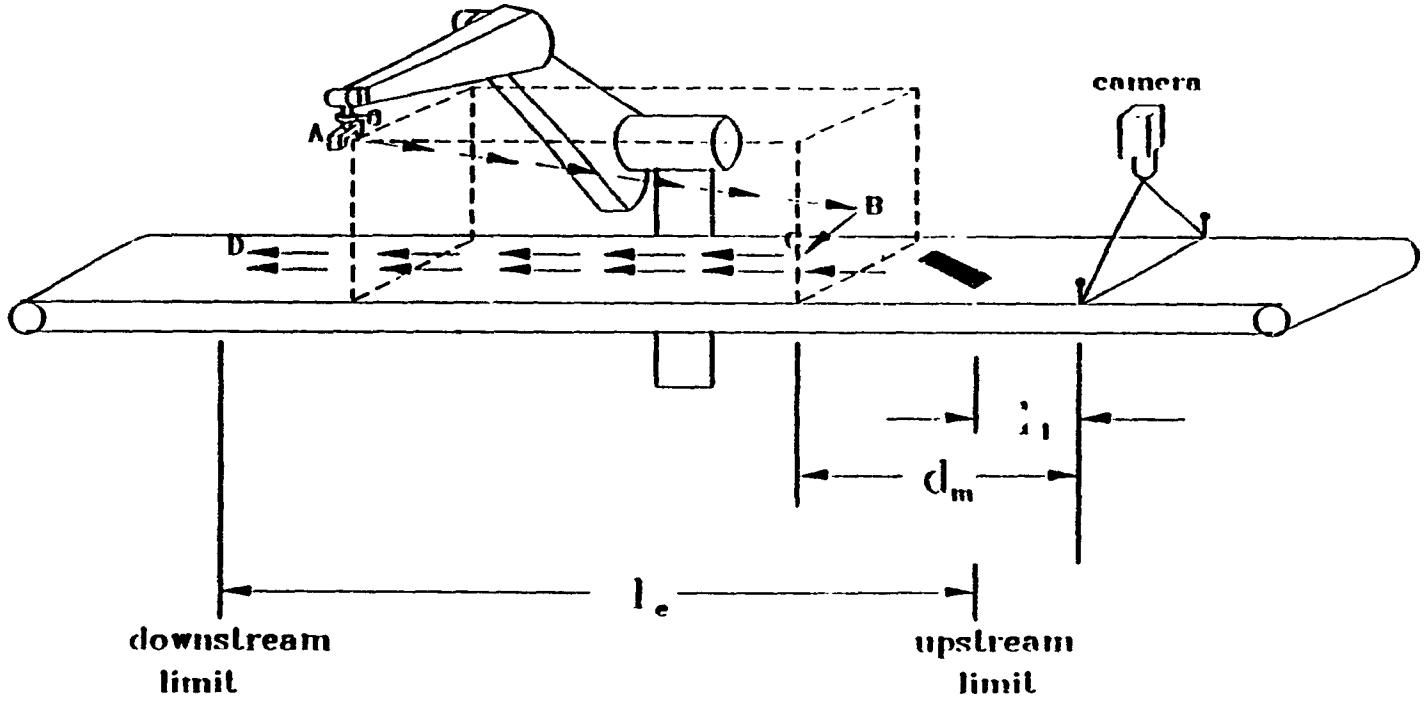


Fig.4.13 Upstream and Downstream Limit of the Workspace

The productive space of the two control models can be compared by substracting the equations (4.58) by (4.59) to obtain the following equation (4.60),

$$d_{s1} - d_{s2} = - d_{m1} + d_{m2} + d_t \qquad \ldots(4.60)$$

Let Δd_s be the difference in the productive workspace, therefore equation (4.60) becomes

$$\Delta d_s = d_{s1} - d_{s2} = d_{m2} + d_t - d_{m1} \qquad \ldots(4.61)$$

If Δd_s is greater than zero, the MT model has more productive space. If it is less than zero, the RT model has more productive space. By examining equation (4.61), it is found that the value Δd_s is greater than zero because of the following reasons :

1. The elliptical path is longer than the linear path. Thus, if the robot velocity is kept constant in both models during the interception phrase, the time required to achieve the RT model interception is more than that required in the MT model. In other words, the workpiece will travel more downstream in the RT model. Therefore, it can be concluded that d_{m2} greater than d_{m1}.
2. The distance d_t is positive.

The comparison has been verified by a series of experiements which are shown in fig.4.14 and fig.4.15. The graph (a) is the MT model and graphs (b), (c), (d) are the RT model with different values of 's'. The robot is running at 200 mm/sec in all cases. The x-axis of the graphs is the time in seconds. The y-axis is the robot end-effector position in millimeters. At home position A, the robot indicates itself is at x= 210mm. With respect to the robot home position A, the camera station is at the x-position -790mm. The

upstream limit position is -620mm and the downstream limit position is 610mm. These positions have been indicated on the y-axis of the graphs in fig.4.14 and fig.4.15. For the MT model, the graph has two straight lines - MBG and ABG. The straight line MBG represents the moving workpiece position. At position M ,approximately -650mm in robot coordinate, the image processing has finished and the robot starts to activate. Another straight line ABG comes from the top, approximately 210mm, represents the path of the robot end-effector. The robot starts at the point A, i.e. home position. The section AB is the interception phase. The robot intercepts the moving workpiece at the point B. The section BG onwards is the synchronisation (productive) phase. The point G is the downstream limit. For the RT model, the section AB' is the elliptical interception phrase the section B'C is the transition phrase. The section CG is the synchronisation(productive) phase. Table 4.2 is the summary of the test runs shown in fig.4.14 and fig.4.15. Table 4.2 shows that the MT model provides a larger productive space.

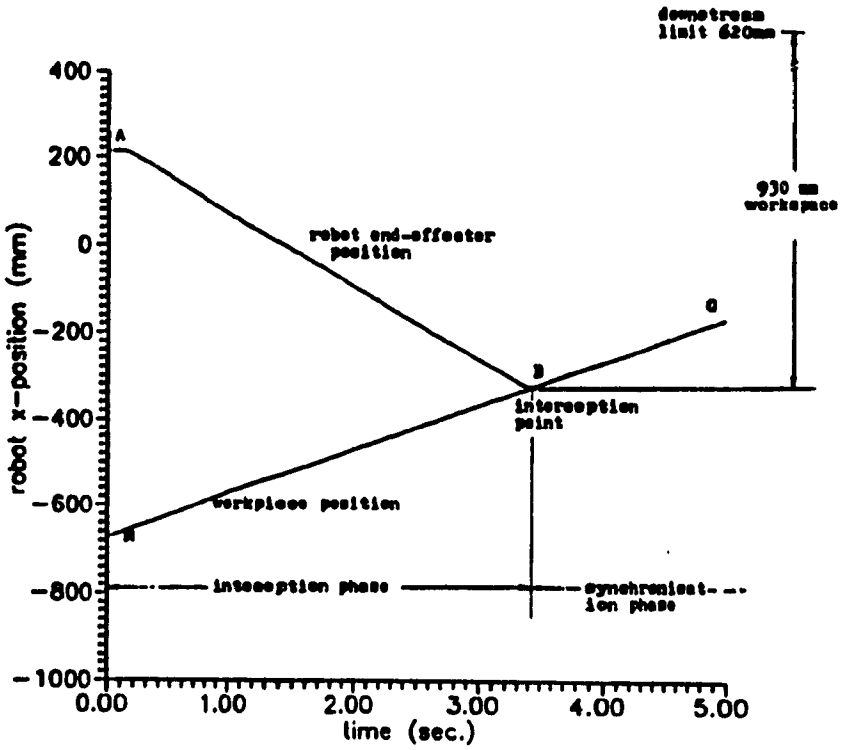


Fig. 4.14a Productive Space of MT Model (v_c=100mm/sec v_r=200mm/sec)

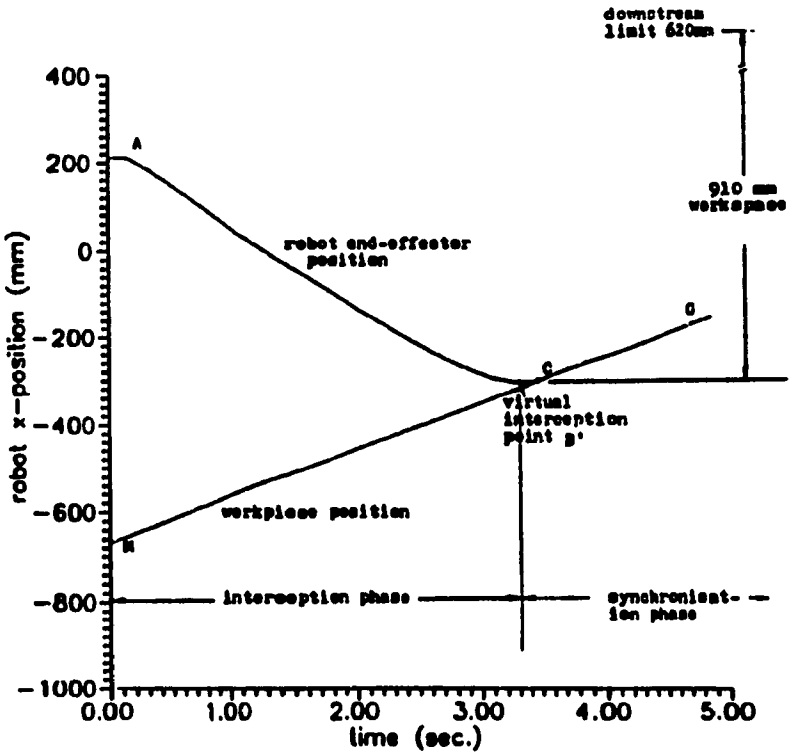


Fig. 4.14b Productive Space of RT Model (v_c=100mm/sec v_r=200 mm/sec s=10mm)

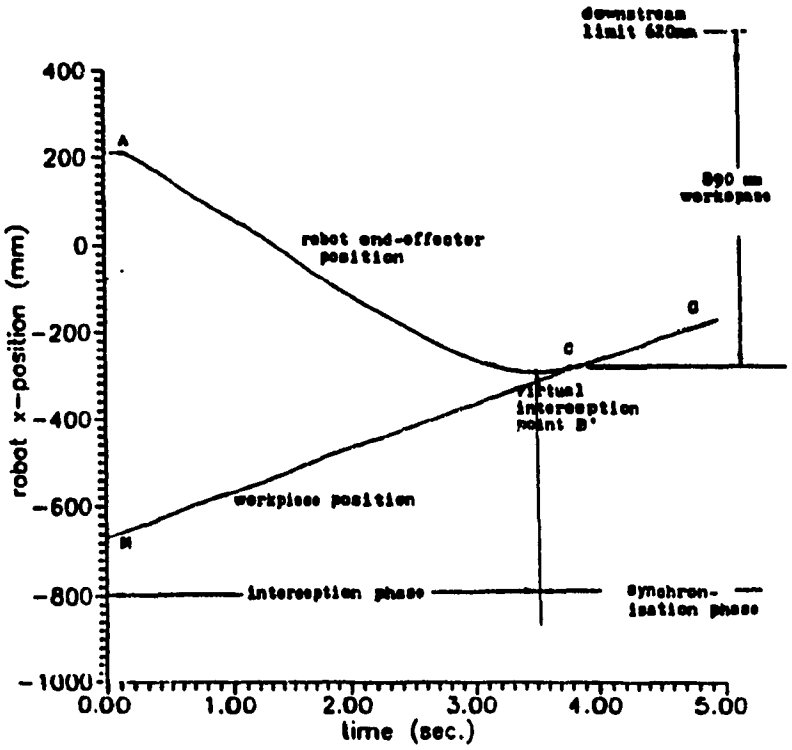


Fig. 4.14c Productive Space of RT Model (v_c=100mm/sec v_r=200 mm/sec s=25mm)

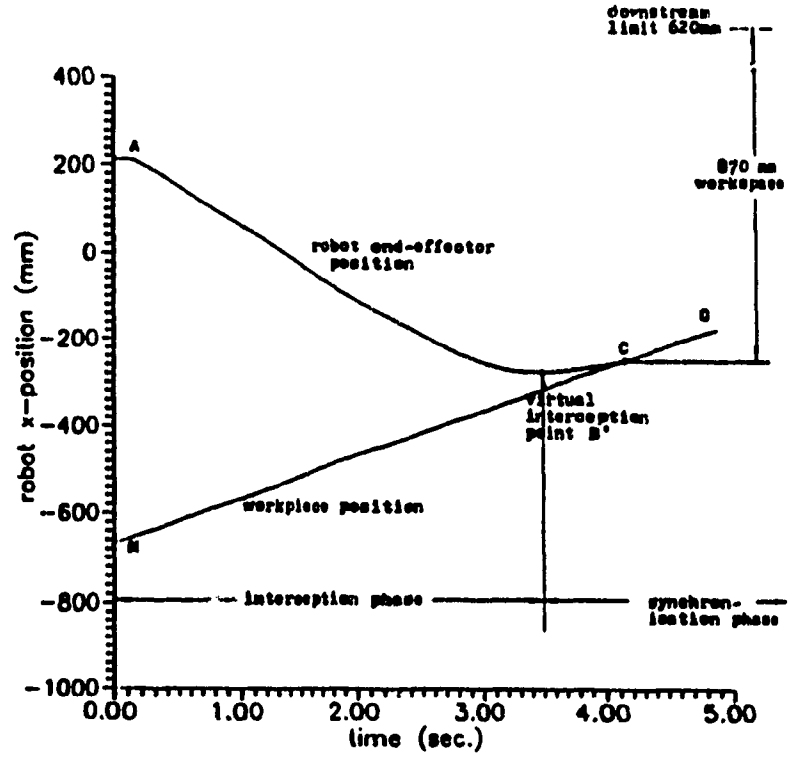


Fig. 4.14d Productive Space of RT Model (v_c=100mm/sec v_r=200 mm/sec s=50mm)

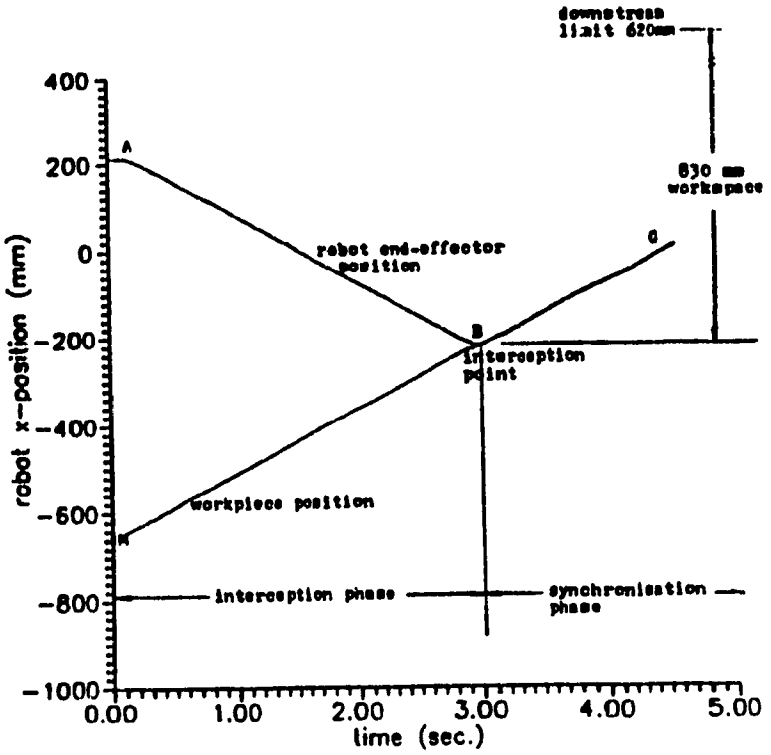


Fig.4.15a Productive Space of MT Model (v_c=150mm/sec v_r=200mm/sec)

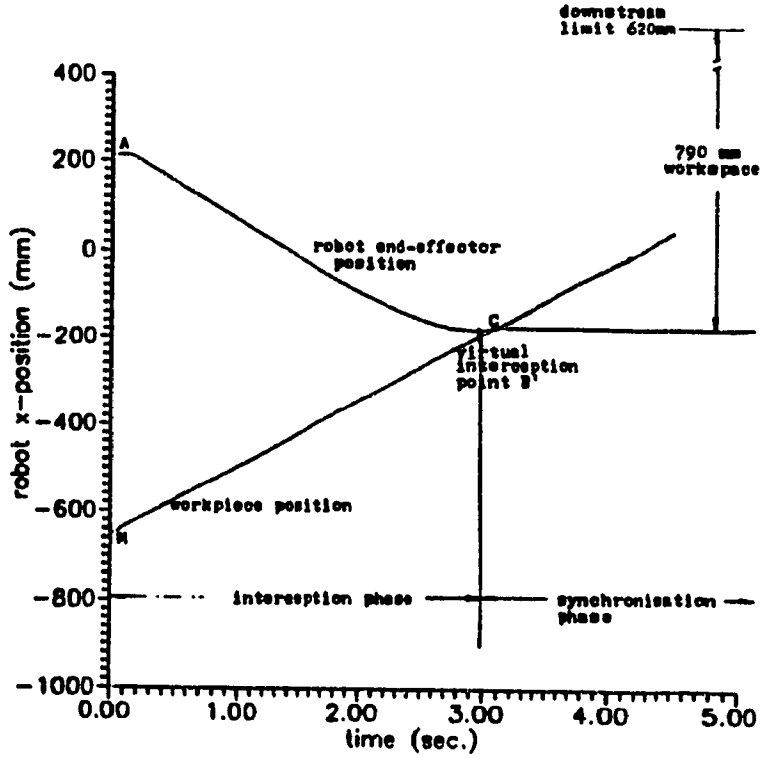


Fig.4.15b Productive Space of RT Model (v_c=150mm/sec v_r=200 mm/sec s=10mm)

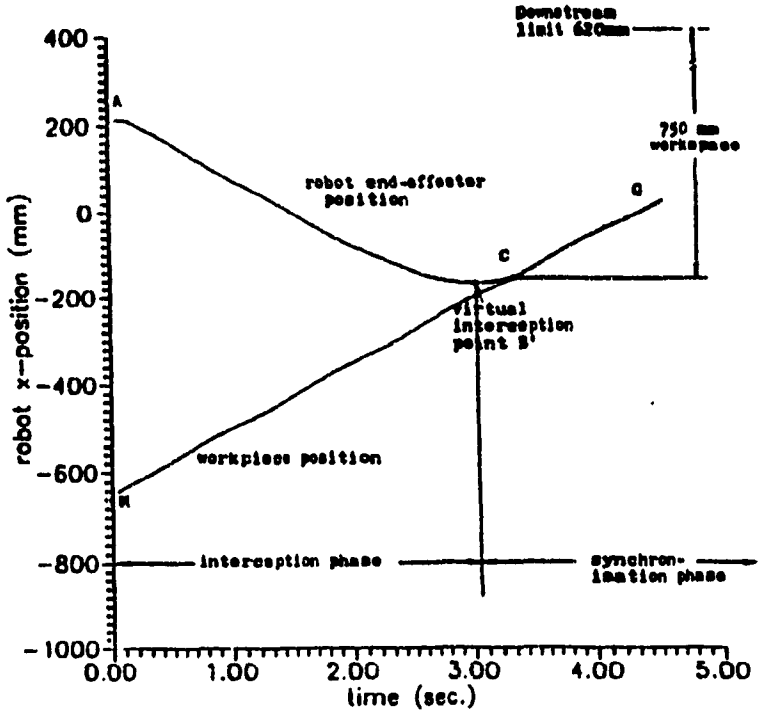


Fig.4.15c Productive Space of RT Model (v_c=150mm/sec v_r=200 mm/sec s=25mm)

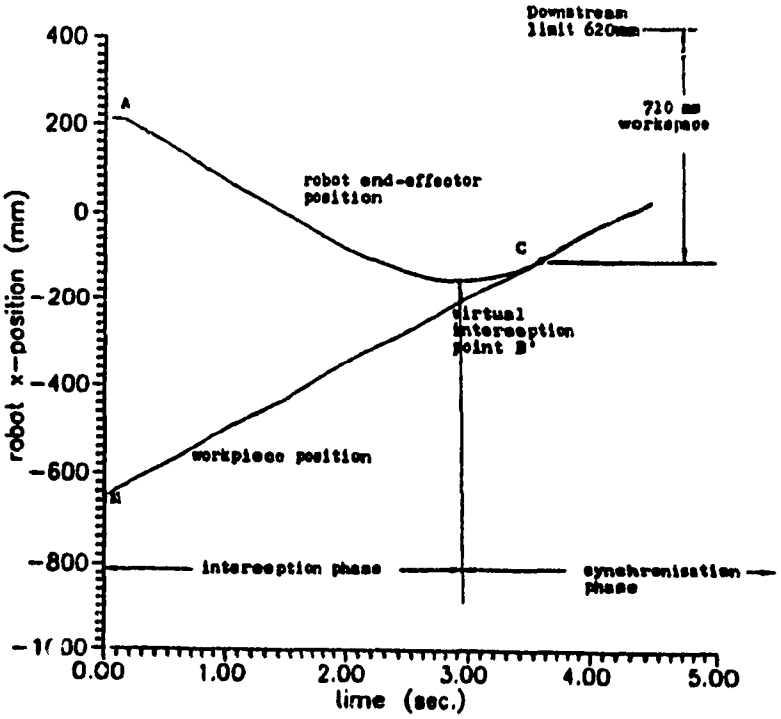


Fig.4.15d Productive Space of RT Model (v_c=150mm/sec v_r=200 mm/sec s=50mm)

Table 4.2 : Productive Space Comparison of MT and RT Models

Graph	Model	s (mm)	productive space (mm)
14a	Minimum Time	/	930
14b	Reduced Torque	10	910
14c	Reduced Torque	25	890
14d	Reduced Torque	50	870
15a	Minimum Time	/	830
15b	Reduced Torque	10	790
15c	Reduced Torque	25	750
15d	Reduced Torque	50	710

4.4.2 Torque Comparsion

The drastic change of the end-effector in x-direction at the interception point causes a large torque transient at joint #1 motor. The objective of this section is to concentrate on comparing the joint #1 torque transient required in both models. Although six joint torques have been computed, only joint #1 is shown for the discussion. The joint torques are computed by inputing the joint positions in real-time. The joint positions are then differentiated numercially(Appendix F) offline to obtain the joint velocities and accelerations. With the joint positions, velocities, accelerations and the inertia parameters, the joint torques at each iteration can be calculated through the Newton-Euler inverse dynamics model. For detailed analysis on these dynamical equations, the reader is advised to refer reference[11]. Detailed modelling is beyond the scope of this thesis and the final form of the equations can be referred to the Appendix G.

The major drawback of the MT model is to have a torque spike at the interception point which causes jerky joint motion. Two velocity profiles are shown in fig.4.16. Fig.4.16a shows the joints #1 to #3 velocity trend of the MT model and fig.4.16b shows the joints #1 to #3 velocity trend of the RT model. It can be observed that the velocity profile of the MT model is more jerky than that of the RT model. The velocity profiles are just to give a general overview of the trends. The actual numeric comparison are the torque differences which will be shown in the fig.4.17. Fig.4.17a to fig.4.17g shows the experimental results of the torque comparison. The curves represent the computed torque of joint #1. It is observed that there is a spike in the

transition between the interception and synchronisation phase. The spike is due to the drastic change of the end-effector direction at the interception point. Fig. 17a shows the torque of joint #1 of the MT model in one operation cycle. Fig. 17b to fig. 17g show that torque of joint #1 of the RT model with various transition distance 's'. It can be found that the MT model has a maximum jerky torque transient up to -260 Nm. It is shown that the torque transient can be reduced by using the RT model. The experimental results are summarised in the following Table 4.3 :

Table 4.3 Torque Transient Comparison

Graph	Model	s (mm)	Maximum Torque Transient	%
17a	Minimum Time	/	-260 Nm	100
17b	Reduced Torque	0	-120 Nm	46
17c	Reduced Torque	5	- 95 Nm	37
17d	Reduced Torque	10	- 70 Nm	26
17e	Reduced Torque	15	- 65 Nm	25
17f	Reduced Torque	25	- 65 Nm	25
17g	Reduced Torque	50	- 65 Nm	25

By using the RT model , Table 4.3 shows that the joint #1 torque can be reduced. Further reduction can be achieved by increasing the transition distance 's'. As shown in Table 4.3, the torque can be furthur reduced to 25% of the MT model. The torque cannot be furthur reduced because certain torque is still required to hold the arm and to maintain the smooth motion.

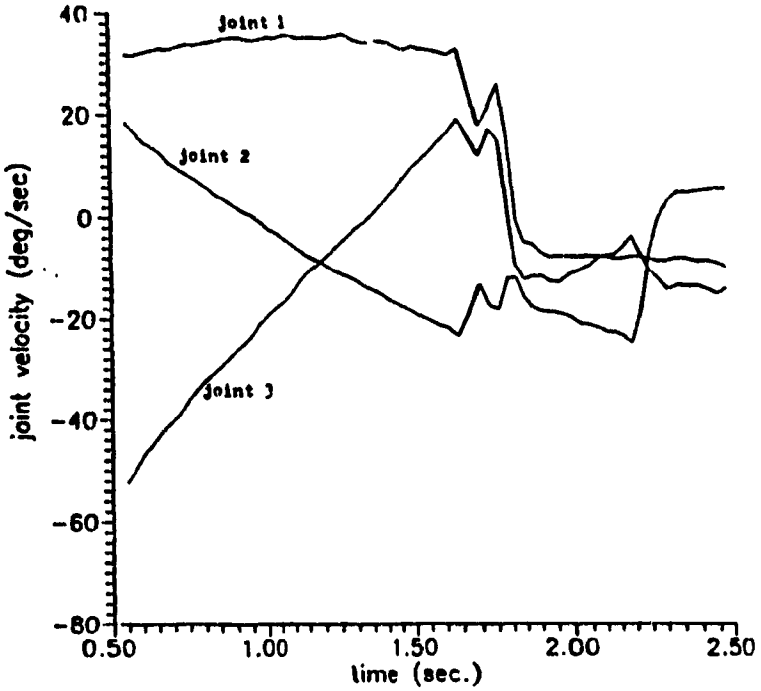


Fig. 4. 16a Velocity Profile (MT Model)

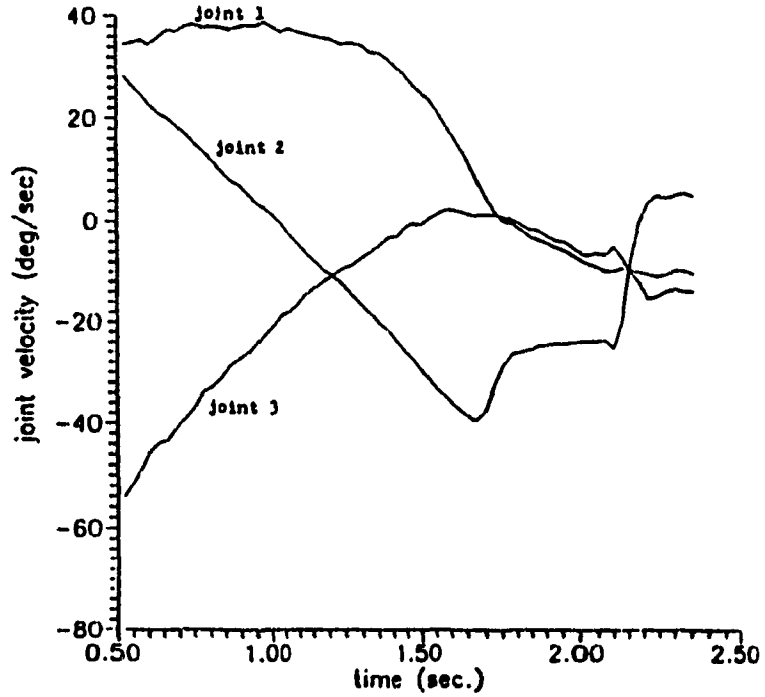


Fig. 4. 16b Velocity Profile (RT Model)

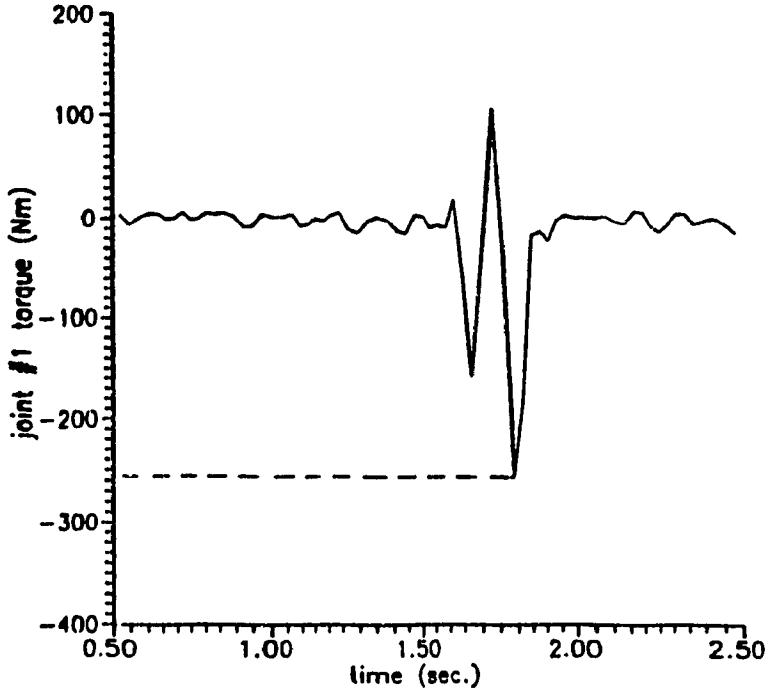


Fig.4.17a Joint #1 Computed Torque Profile (MT Model)

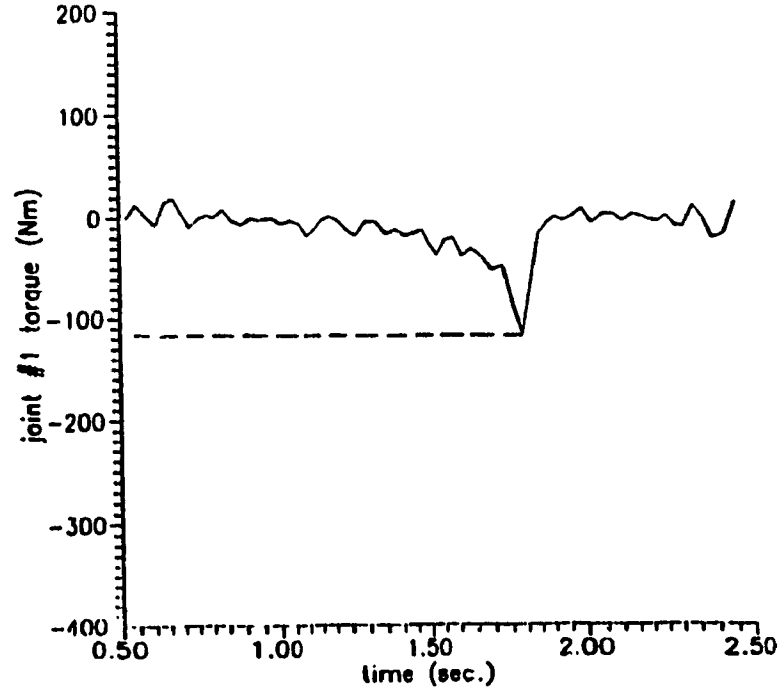


Fig.4.17b Joint #1 Computed Torque Profile (RT Model s = 0mm)

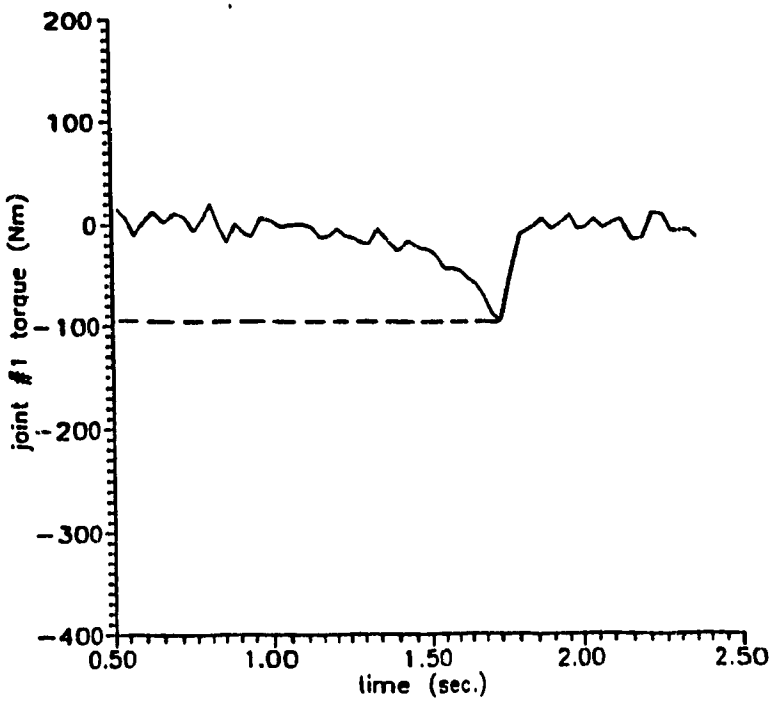


Fig.4.17c Joint #1 Computed Torque Profile (RT Model s = 5mm)

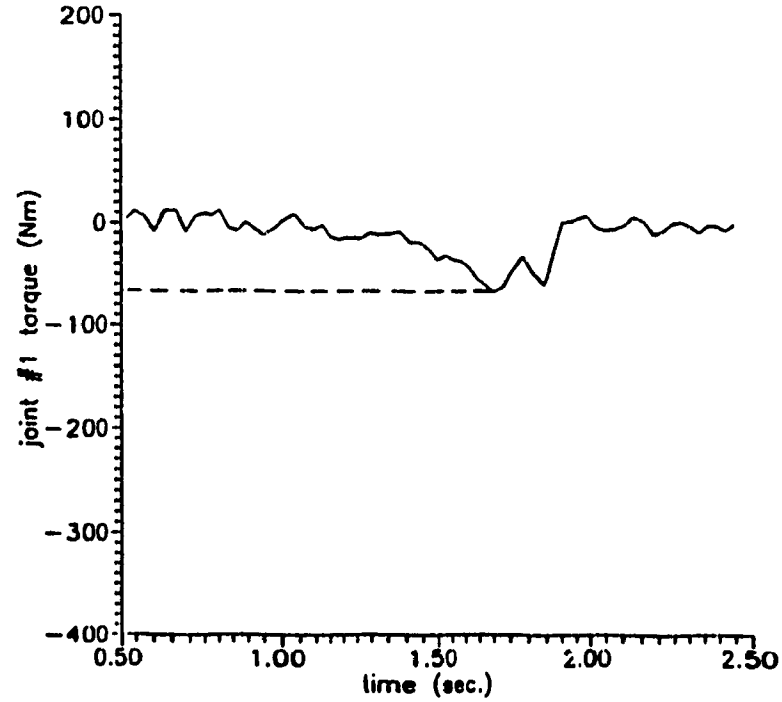


Fig.4.17d Joint #1 Computed Torque Profile (RT Model s =10mm)

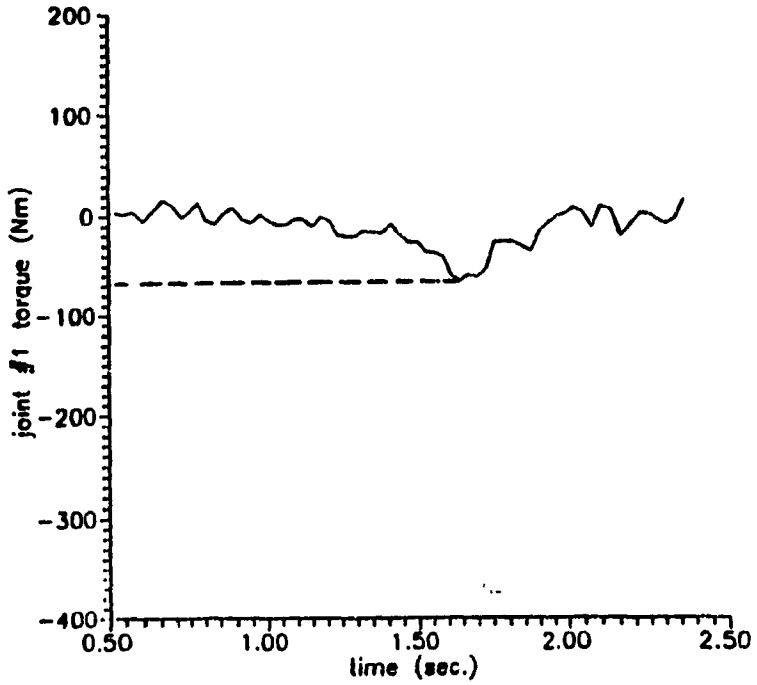


Fig.4.17e Joint #1 Computed Torque Profile (RT Model s =15mm)

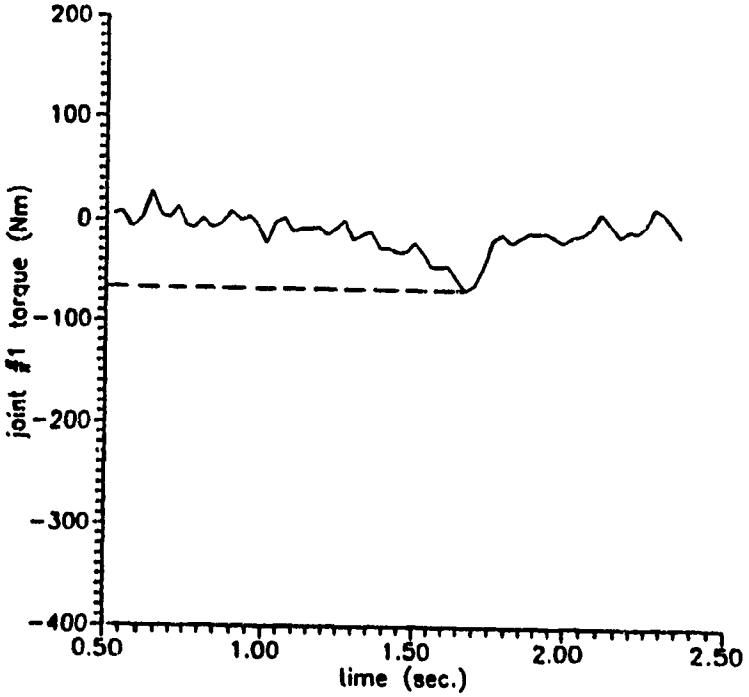


Fig.4.17f Joint #1 Computed Torque Profile (RT Model s =25mm)

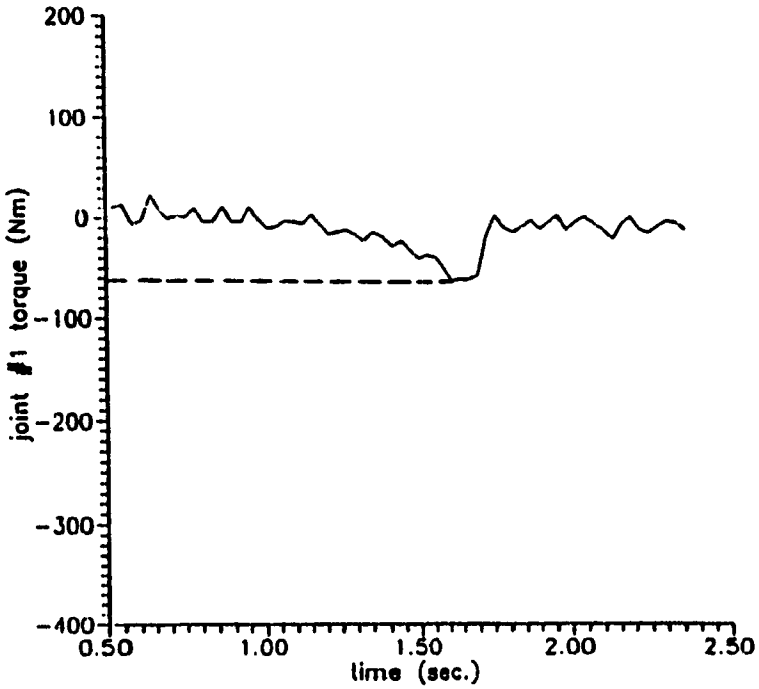


Fig. 4.17g Joint #1 Computed Torque Profile (RT Model s =50mm)

4.4.3 Synchronisation Performance

The synchronisation error is the position error between the end-effector and the moving workpiece on the conveyor. In particular, it is important in the on-line assembly operations. The objective here is to evaluate the error offset between the robot end-effector and the moving workpiece during the synchronisation phrase by taking the time delays in section 4.2 into consideration. Fig.4.18 shows the behaviour of the robot when controlled by a control scheme in which all time delays(τ_s, τ_d, τ_a) are set to zero. The robot trajectory is represented by the three separate curves (x, y, z). Let us concentrate on comparing between the x-position curve and the workpiece position. In both simulation and experimentation, it is observed that after interception, the robot tends to overshoot, and then lags behind the workpiece during the synchronization stage. While the velocity error is relatively small, the position error is typically approximately equal to 4 times (τ_a/τ_s) the product of the sample period (τ_s) and the speed of the conveyor (v_c). The implementation of this control scheme is relatively simple, but the performance is clearly not suitable for applications which demand positional accuracy. By incorporating the provision of the various important time lags into the control scheme through the use of equations (4.1) to (4.12), dramatic improvement in accuracy is observed.

Some results arising from the simulation studies and from experimental runs of the workcell are presented in the following sections. Section 4.4.3.1 discusses the simulation technique. The sections 4.4.3.2 and 4.4.3.3 compare the simulation and the experimental synchronisation performances of the two models. In all subsequent sections, emphasis is only on the x-position only. The y,z

errors are not being a function of conveyor speed (i.e. quasi-static process) and therefore they are neglected in the following discussions.

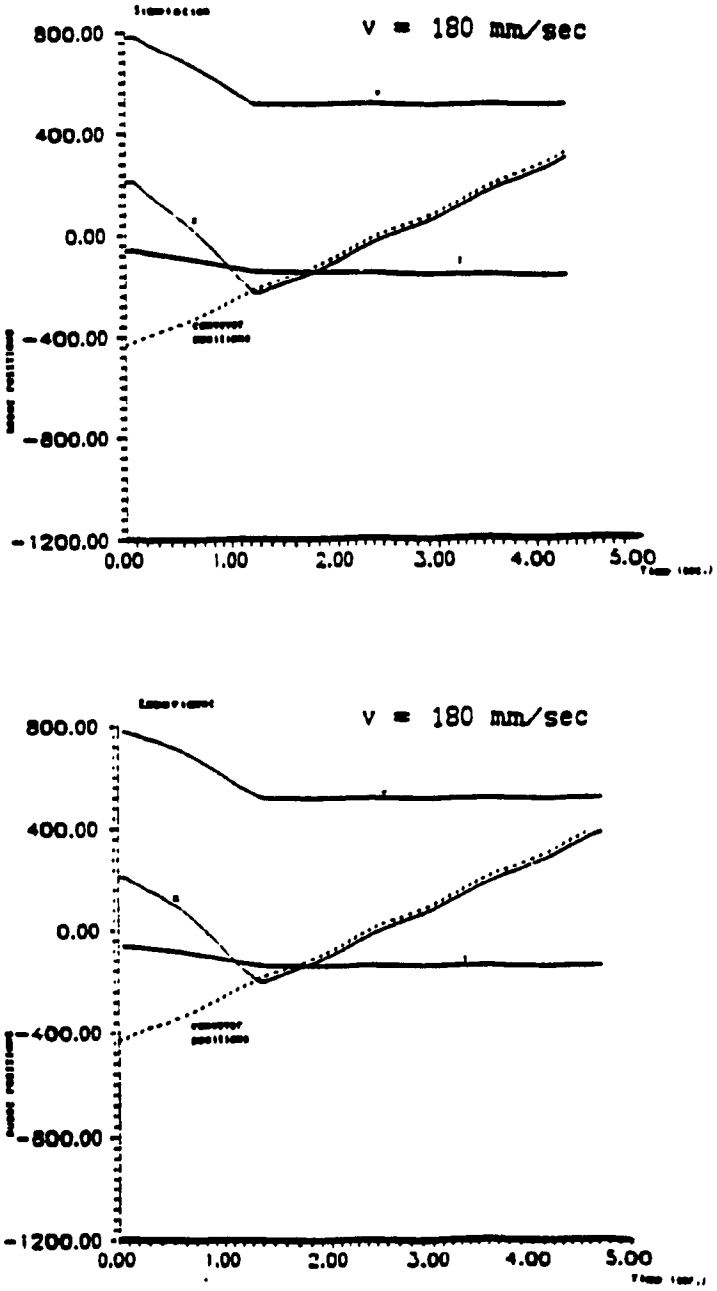


Fig. 4.18 MT Synchonisation Performance (Delays Ignored)

4.4.3.1 simulation technique

The model of the path simulation is shown in fig.4.19. The simulation model consists of three blocks - Initialisation Block, Interception Block and Synchronisation Block. The inputs of the Initialisation Block are the conveyor velocity v_c, number of the image frame of the workpiece n_f and the image processing time per frame t_{pf}. It calculates the initial workpiece position. One of the models - MT or RT can be used in the Interception and Synchronisation Blocks and they require the following inputs :

- workpiece position and orientation (X_c, Y_c in chapter 2)
- robot home position (point A in fig.4.3)
- robot velocity (v_r)
- controller delay time (τ_s, τ_d, τ_a in section 4.2)
- conveyor velocity (v_c)
- workpiece x-position (p_{wx})
- robot end-effector descending velocity (v_z)

The movement of the workpiece at each iteration can be updated by the distance of the conveyor travelling at the speed v_c during the period of the sampling τ_s.

i.e. $$p_{wx}[k] = p_{wx}[k-1] + v_c\ \tau_s$$

With respect to the updated workpiece position at each iteration, accumulative commands are then calculated by the path model(MT or RT). The robot positions are updated by the summation of the accumulative commands.

$$p_{rx}[k-1] = \Sigma r_x[i]$$

$$p_{ry}[k-1] = \Sigma r_y[i]$$

$$p_{rz}[k-1] = \Sigma r_z[i]$$

The robot and workpiece positions are then stored in a file corresponding to the specfic time $t = k\tau_s$. The synchronisation performance can be indicated by comparing the offset between the workpiece position $p_{rx}[k]$ and the end-effector position $p_{wx}[k]$.

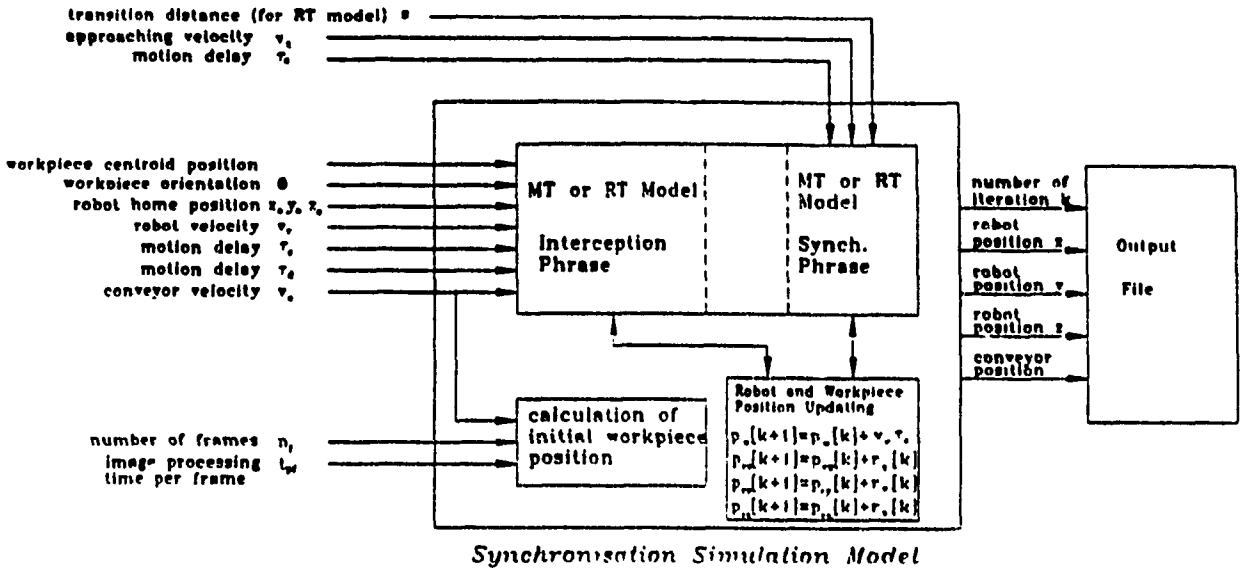


Fig.4.19 Path Simulation Model

4.4.3.2 MT model synchronisation performance

Fig.4.20 shows the improved synchronisation performance of the robotic workcell for three different conveyor speeds. While the simulations (left column of fig.4.20) show perfect synchronization between the robot end-effector and the moving workpiece, the experiments still show minor errors that vary from 1.0 to 2.5 mm for a conveyor speed range of 100 mm/sec to 150 mm/sec. The overshoot in each case is correspondingly reduced. By manipulating the x-incremental commands as given by equation (4.9), one can obtain higher degrees of success in reducing or eliminating this overshoot. However this fine-tuning does not seem to have appreciable effect on the synchronization errors.

To investigate the integrity of this control model, severe step disturbances in the conveyor speed are imposed, and the results are shown in fig.4.21. During each of the three runs, the conveyor is brought to an abrupt stop at time t_a and then restored to its original speed moments later at time t_b. In all cases, the system is observed to be well-behaved, with no appreciable increase in positional errors when compared to fig.4.20.

The insignificant error can be attributed to the following factors:

(a) there is an observed discrepancy (0.2% average) between the robot positional reading as provided by the VAL II software of the PUMA controller and that monitored by the data acquisition system developed for this purpose.

(b) the forward kinematic model that is used to transform the acquired joint data into world-coordinates (x,y,z etc) probably

contributed an error of ± 0.5 mm in the x-direction;

(c) the conveyor speed controller is capable of regulating to no better than 1% .

4.4.3.3 RT model synchronisation performance

A series of tests have been experimented by varying the robot and conveyor velocities. Fig.4.22 shows the synchronisation performance of the RT model. The simulation results (right column of fig.4.22) and experimental results (left column of fig.4.22) show that the synchronisation performance of the RT model is very good. However, the accuracy depends on the range and ratio of the robot velocity and the conveyor velocity. The relationship of the synchonrous error, conveyor velocity and the effector velocity is summarised in the fig.4.23. By using proper combination of robot velocity and conveyor velocity, one can limit the synhronisation error to within 1.0- 2.0 mm.

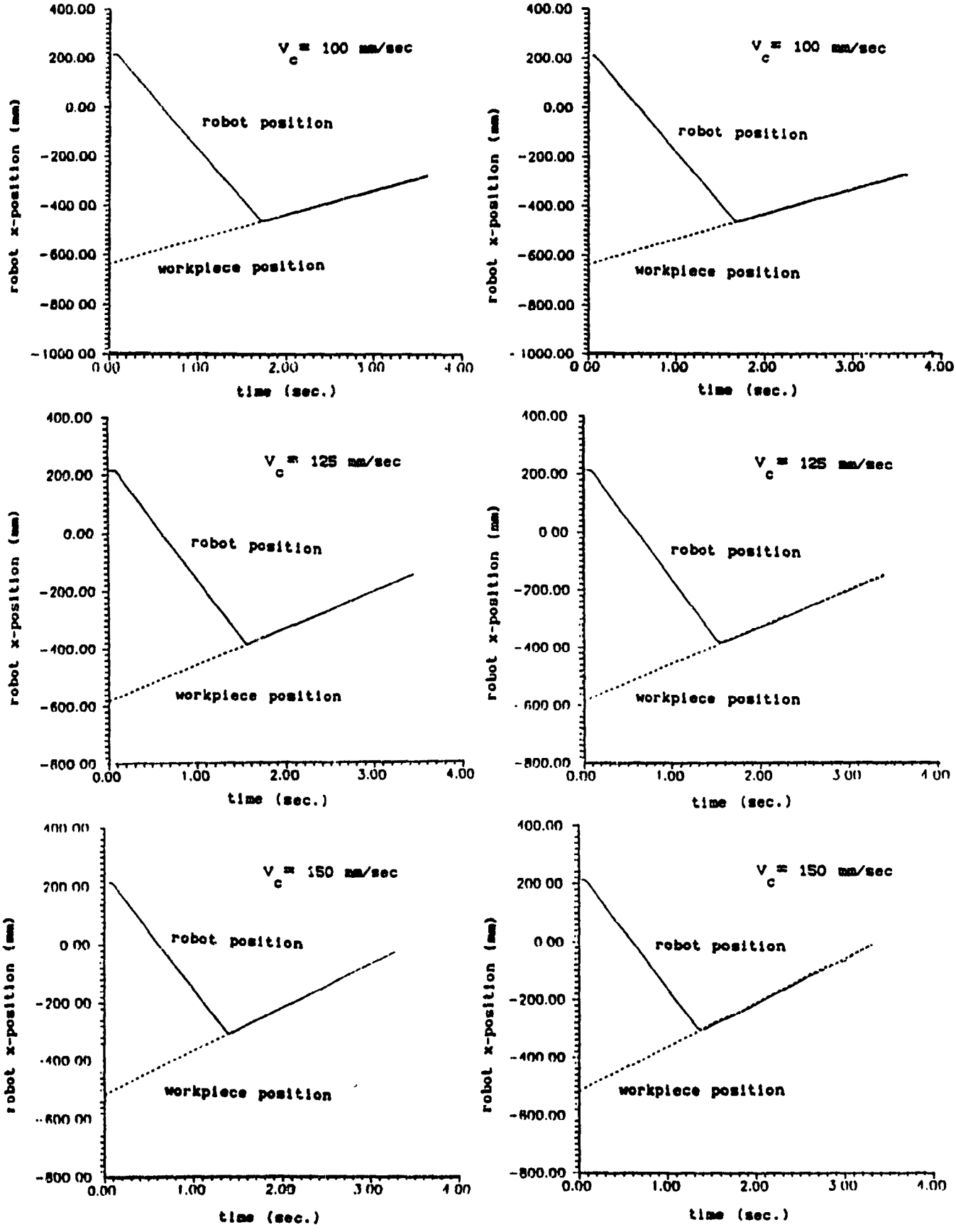


Fig. 4.20 Simulation and Experiment of the Synchronisation Performance of the MT Model

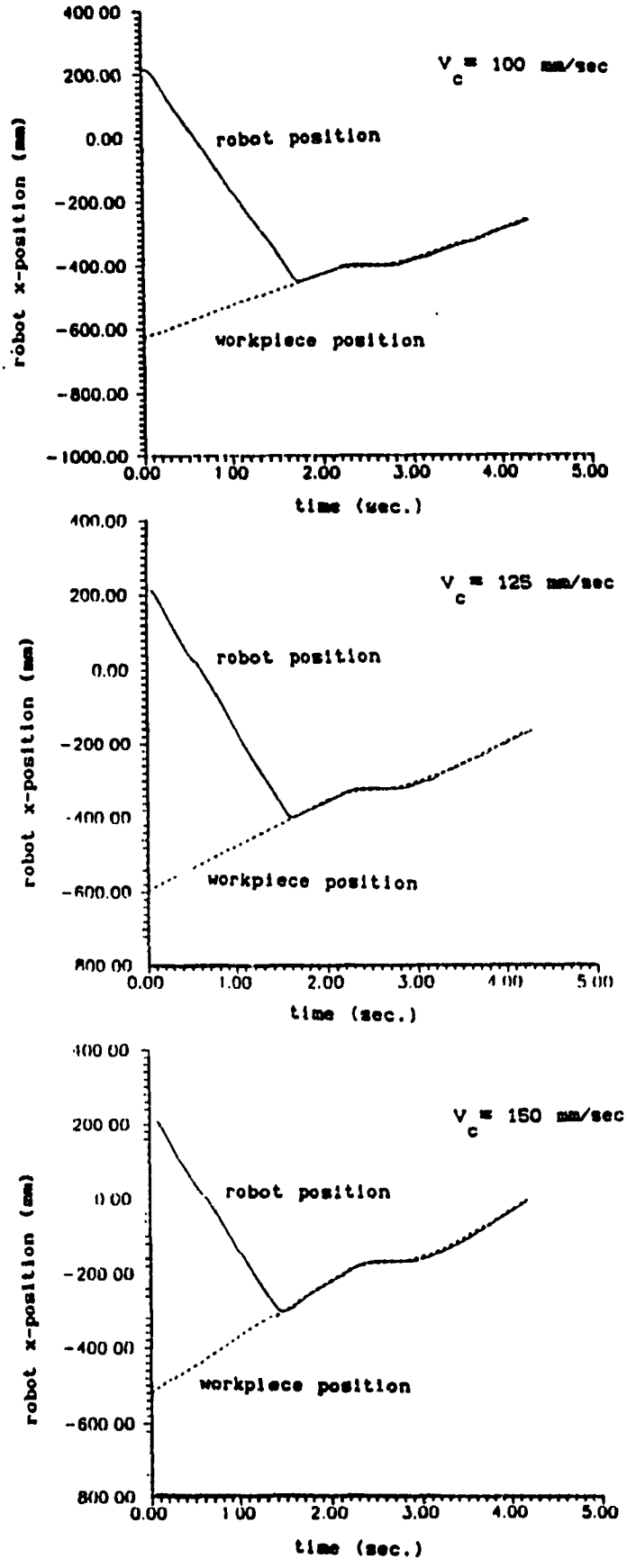


Fig. 4.21 Experiments with Disturbances on Conveyor Speed

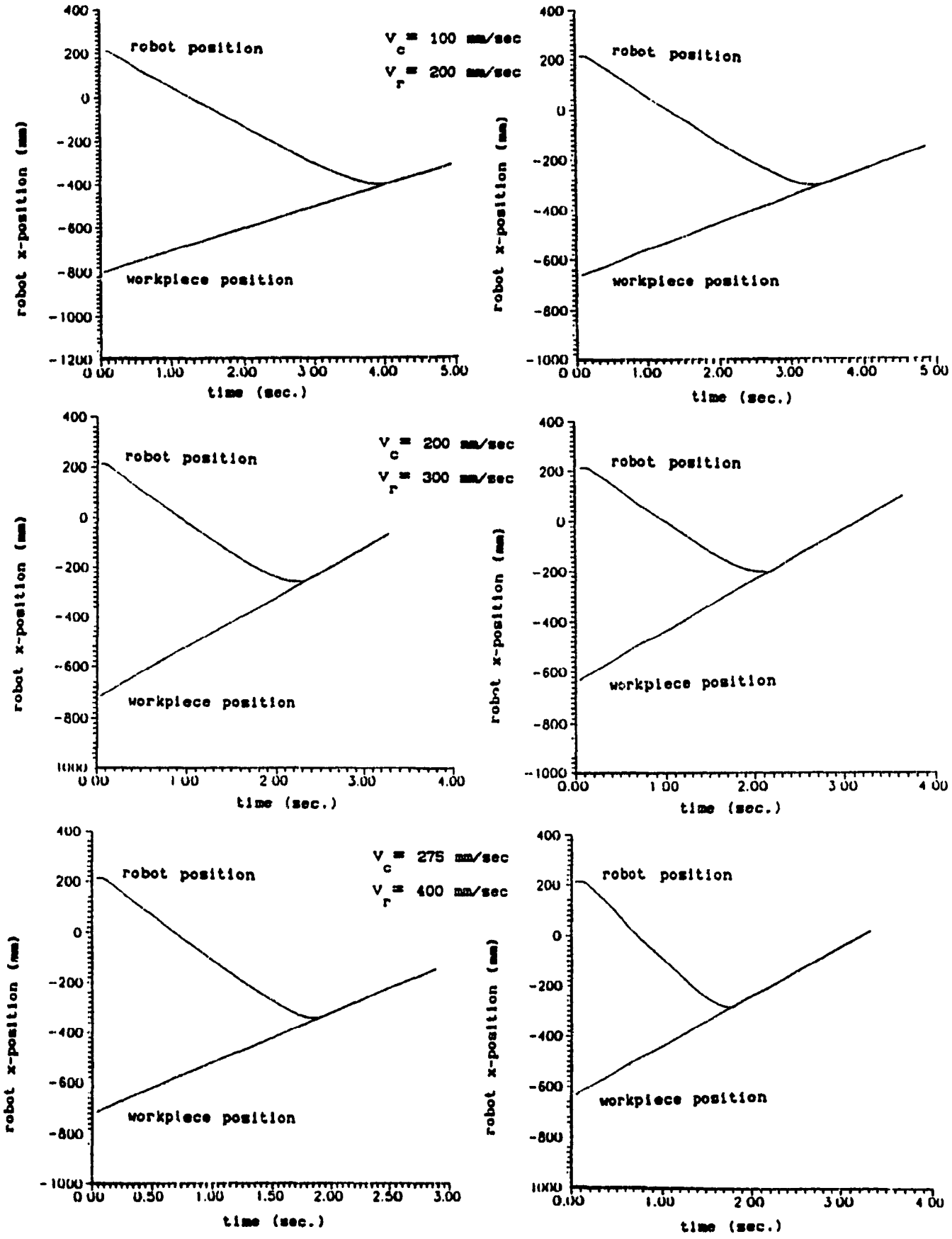


Fig. 4.22 Simulation and Experiment of the Synchronisation Performance of the RT Model

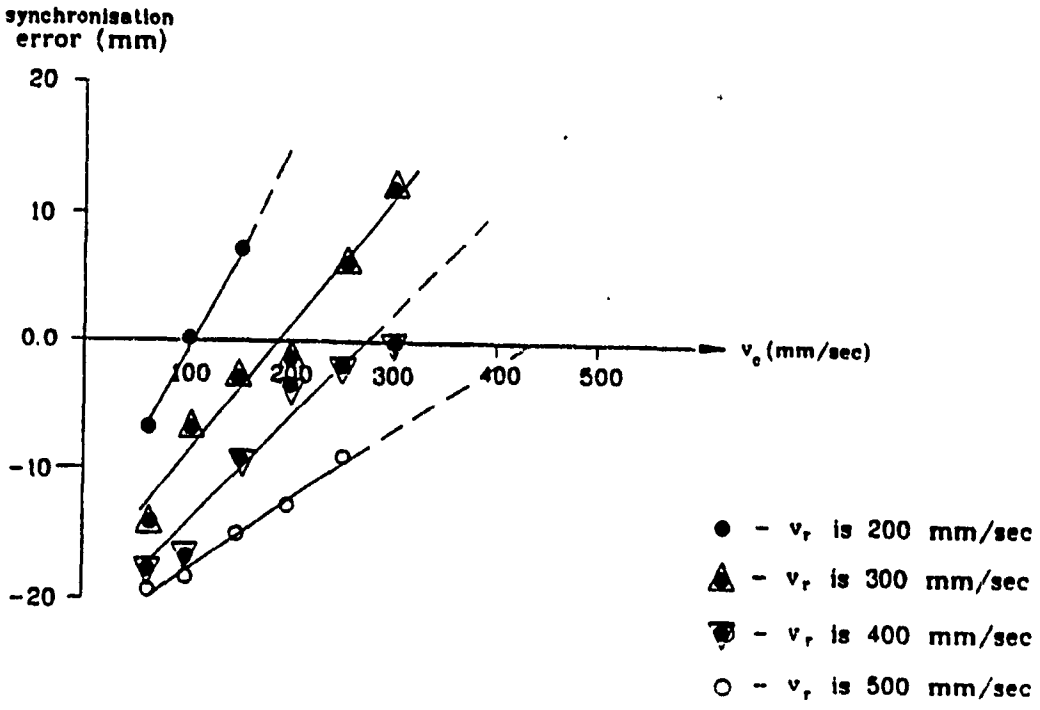


Fig.4.23 RT Model Synchronisation Error

4.5 SUMMARY

This chapter shows the importance of taking the time delays into the control models. By experimentally determining these characteristic delays τ_s, τ_d, τ_a and λ, one can adopt the described discrete control models to design a synchronization control scheme for a robotic workcell. These models have been demonstrated to be very well behaved as regards position errors and sensitivity towards conveyor speed disturbances.

The MT model is recommended to be used in the small on-line assembly appplications. The reasons are

1) it provides larger productive space;

2) the synchronisation performance is more accurate under any combinations of robot and conveyor velocities.

In case of handling large industrial parts, the RT model is

recommended to be used because it requires less torque.

One should recall that the models presented here is by and large an open loop control scheme, the joint feedback data being used only for monitoring purposes and not incorporated into the overall control loop. An overall feedback feature will be described in next chapter. It becomes possible to achieve a greater degree of adaptive containment of the system errors.

Chapter 5

OVERALL FEEDBACK TASK

5.1 INTRODUCTION

The synchronisation performance (MT model) shown in fig.4.20 is more or less limited to initial transient after the interception point. However, as time goes on, the position error tends to accumulate and grow typically as in fig.5.1a. Fig.5.1b shows magnified scale to indicate the trends of the synchronisation offset. If the conveyor and robot velocities are not carefully chosen for the RT model, fig.4.23 shows that the worst case would have the synchronisation error up to -15mm to +15mm.

In order to compensate such errors, two possible strategies can be used :

The synchronisation offset can be pre-calibrated under different operating conveyor velocities, robot velocities and synchronisation period. At the time of on-line operation, the offset is then incorporated inside the control algorithm. This method may not be efficient nor accurate because of the unpredictable changes of environment , e.g. disturbance of the conveyor velocity.

Another strategy is to provide an overall feedback of the actual end-effector cartesian position. It provides higher adaptive capabilty to compensate for the offset at any operating situation.

This chapter discusses the overall system feedback control loop. Section 5.2 formulates the feedback control model in discrete form. Finally, the experimental results of the MT and RT models would be discussed in section 5.3.

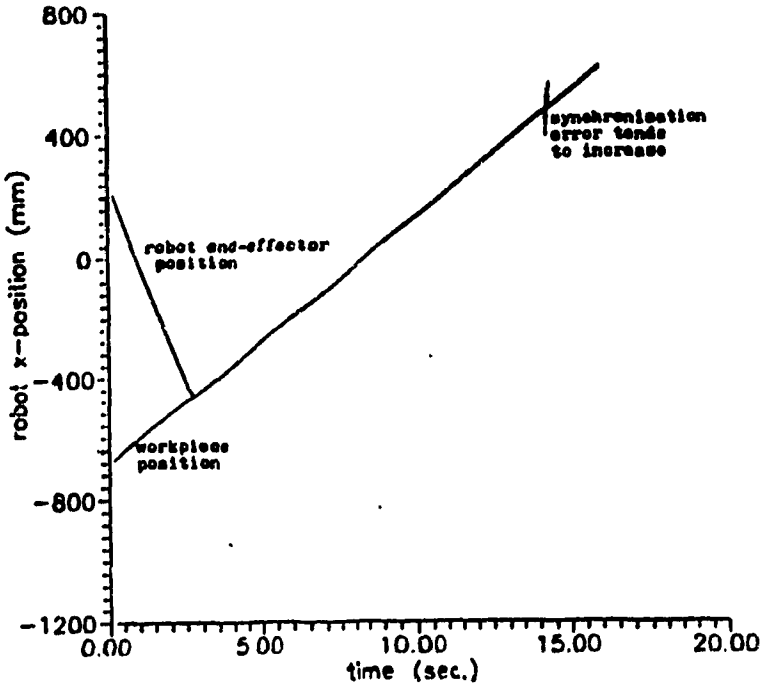


Fig. 5.1a Synchronisation Performance without Overall Feedback

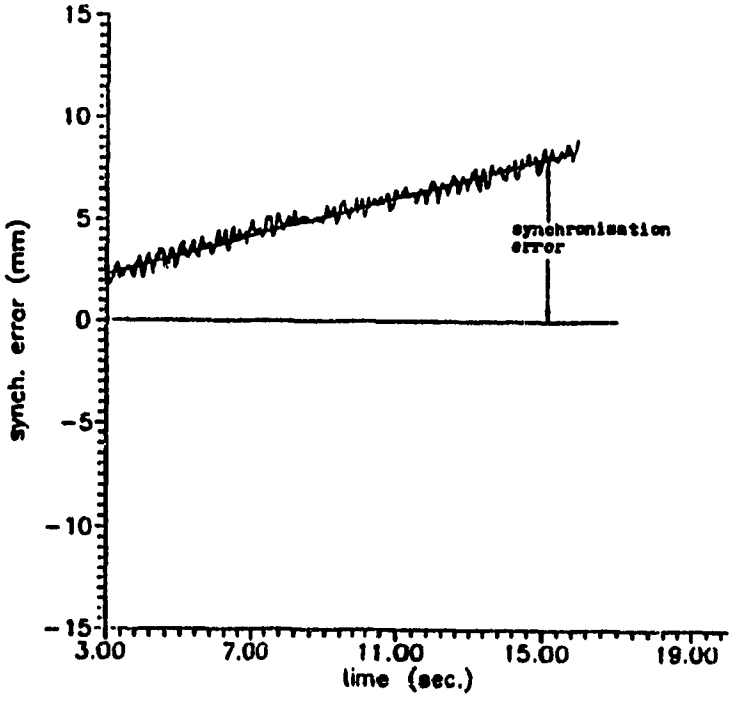


Fig. 5.1b Synchronisation Error

5.2 CONTROL MODEL

5.2.1 Overall Feedback Control Loop

The control models described in chapter 4 are implemented in the hierarchical controller. The incremental position commands are inputed to the motion controller through the ALTER port. The robot control is still dedicated to the motion controller. However, the motion controller is limited to be accessed or modified by the users, e.g. position feedback. In order to improve the synchronisation performance, a joint angle data acquisition circuitry (Appendix B) is used to obtain the instantaneous joint positions to calculate the actual robot end-effector position as the overall position feedback.

Fig.5.2 shows the schematic of the overall feedback controller. The path generator has the inputs of the workpiece geometrical position and orientation (X_c, Y_c, θ) from the image processing task. For every 28 msec, the path generator samples the updated workpiece position p_w and velocity v_c from the tracking circuitry. The path generation model calculates the incremental motion commands (r_x, r_y, r_z). The commands are then fed to the robot controller to move the robot. The path generator keeps updating the movements of the robot by adding the incremental position commands. The robot motion controller consists of the inverse kinematic model to transform the cartesian motion commands into the joint motion commands. The individual joint controls are carried out by six joint servo which are situated inside the motion controller. The overall feedback controller consists of a forward kinematic model(appendix H) and a PID controller. The joint angles are fed into a forward kinematic model to compute the actual cartesian end-effector position at kth iteration $x_{act}[k]$. The value $x_{act}[k]$ is then compared with the current workpiece position $p_w[k]$.

The error is fed into the PID controller to calculate the output e_{xo}. The position error e_{xo} is then superimposed on the incremental command r_x to compensate the synchronisation offset.

5.2.2 Overall Feedback Controller Algorithm

The overall feedback control model uses a proportional-integral-derivative algorithm inside the controller. The three controller parameters are the proportional gain K_c, the integral time T_i and the derivative time T_d. The following is the formulation of the overall feedback control algorithm.

$$\text{Let } e_{xo} = K_c \left\{ e_x + \frac{1}{T_i} \int e_x \, dt + T_d \frac{d\, e_x}{dt} \right\} \qquad \ldots(5.1)$$

The corresponding digital form is derived as follows :

At kth iteration, the equation (5.1) can be expressed into the following equation,

$$\int_0^{k\tau_s} e_x[k] \, dt \cong \sum_{i=0}^{k} e_x[i] \, \tau_s = \tau_s \sum_{i=0}^{k} e_x[i] \qquad \ldots(5.2)$$

$$\frac{d\, e_x[k]}{dt} \cong \frac{e_x[k] - e_x[k-1]}{\tau_s} \qquad \ldots(5.3)$$

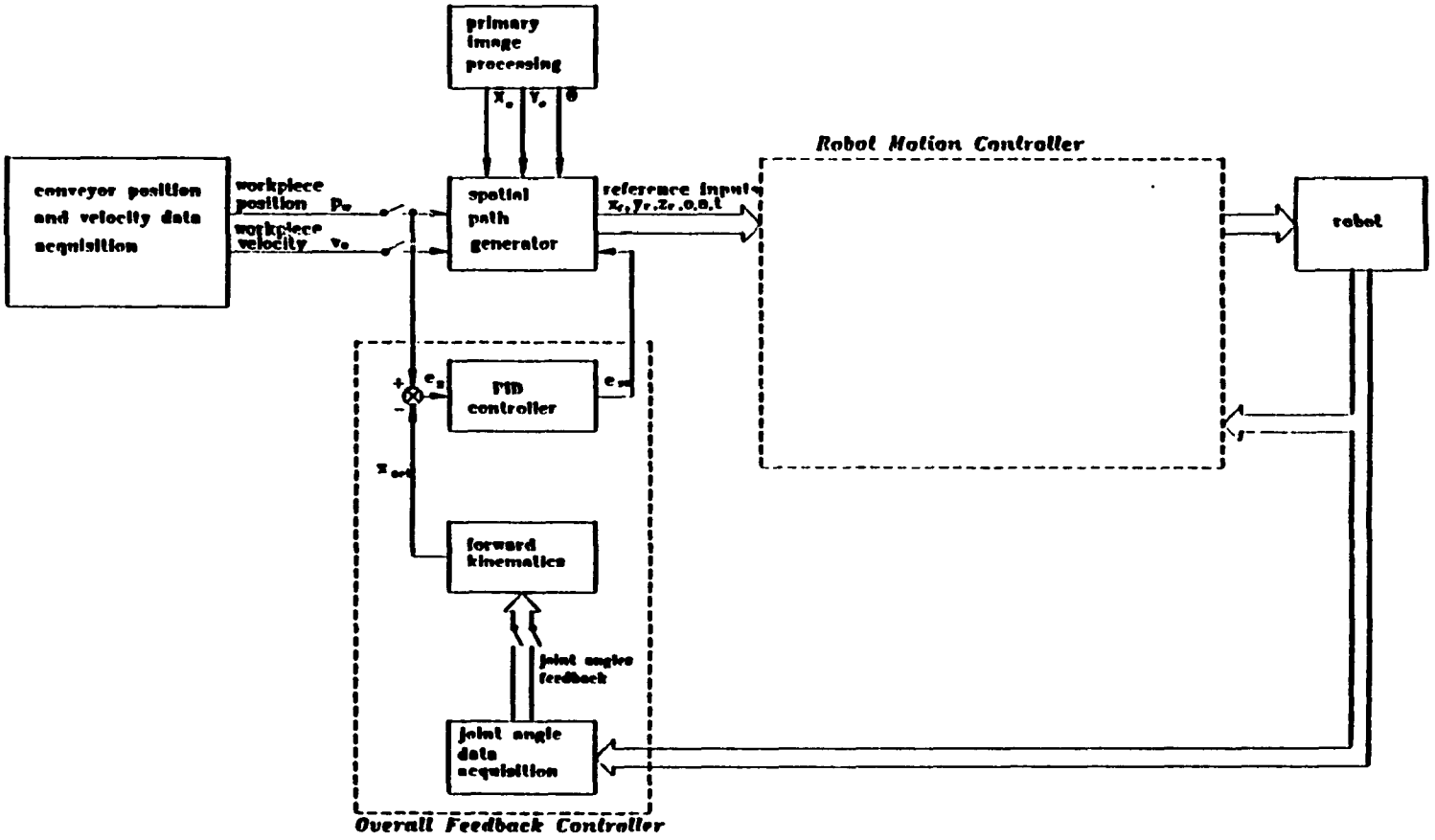


Fig.5.2 Schematic of the Overall Feedback Controller

Substituting equation (5.2) and (5.3) into (5.1),

$$e_{xo}[k] = K_c \left\{ e_x[k] + \frac{\tau_s}{T_i} \sum_{i=0}^{k} e_x[i] + \frac{T_d}{\tau_s} [\, e_x[k] - e_x[k-1] \,] \right\} \qquad \ldots(5.4)$$

Similarly,

$$e_{xo}[k-1] = K_c \left\{ e_x[k-1] + \frac{\tau_s}{T_i} \sum_{i=0}^{k-1} e_x[i] + \frac{T_d}{\tau_s} [\, e_x[k-1] - e_x[k-2] \,] \right\} \qquad \ldots(5.5)$$

Subtract equation (5.4) from (5.5),

$$e_{xo}[k] - e_{xo}[k-1] = K_c \left\{ [\, e_x[k] - e_x[k-1] + \frac{\tau_s}{T_i} e_x[k] + \frac{T_d}{\tau_s} [\, e_x[k] - 2e_x[k-1] + e_x[k-2] \,] \right\} \qquad \ldots(5.6)$$

Therefore, $e_{xo}[k]$ can be expressed by

$$e_{xo}[k] = e_{xo}[k-1] + K_1 e_x[k] + K_2 e_x[k-1] + K_3 e_x[k-2] \qquad \ldots(5.7)$$

where $$K_1 = K_c \left\{ 1 + \frac{\tau_s}{T_d} + \frac{T_d}{\tau_s} \right\}$$

$$K_2 = - K_c \left\{ 1 + \frac{2\ T_d}{\tau_s} \right\}$$

$$K_3 = \frac{K_c\ T_d}{\tau_s}$$

Let $r_x[k]$ be the incremental position command calculated by the spatial path generator

Let $r'_x[k]$ be the incremental position command to actually move the robot

The synchronisation offset can be compensated by adding the incremental position command $r_x[k]$ with the error output $e_{xo}[k]$. Thus,

$$r'_x[k] = r_x[k] + e_{xo}[k] \qquad \ldots(5.8)$$

5.3 EXPERIMENTAL RESULTS

5.3.1 Controller Tuning

There are infinite number of combinations of the three parameters, K_c, T_i, T_d and different sets of parameters may have different responses. Tuning is a process to find out the optimum setting so that the response can meet the required specifications. The criterion for this overall feedback controller is to maintain the synchronisation error within 1.0mm.

Generally speaking, there are two types of tuning techniques -- open-loop and closed-loop. For the open-loop tuning, there are Reaction Curve Tuning[87] and Analytical Tuning[96] methods. However, these open-loop tuning techniques have their limitations and therefore sometimes a true closed-loop response cannot be obtained. For example, the required system time-delay in the Reaction Curve Tuning model may not be very accurate to evaluate the controller parameters. And the Analytical Tuning method only restricted to the pure second order system which may not be adequate to model a 6-dof robot. In order to observe a true closed-loop system response, the closed-loop tuning methods are preferable. There are three common closed-loop tuning

methods.

. trial and error

. minimum error criterion

. continuous cycles method

The first method is by trial and error[58]. First of all, the integral action and derivative action are removed by setting the T_i to be very large and T_d to be zero. The proportional gain K_c is set to approximately 0.5. Then, the value of K_c is doubled for every test until the system becomes underdamped and oscillatory. At this time, the gain is called the ultimate gain. The ultimate gain is then decreased by a factor of 2.0 to become the proportional gain of the system. The same technique is applied to the the integral actions by decreasing the T_i a factor of 2.0 until the system starts to be underdamped. After the values of K_c and T_i are determined, the value of the T_d is doubled for every trials until the noise become significant appeared in the system response. This method is simple. However, it does not necessarily come to the optimal settings.

The second closed-loop tuning method is called error integral criterion. It finds the optimum settings of a controller by means of evaluating the minimum integral error from a closed-loop transfer function[74]. Considering a time response of a system is characteristized by a relation :

$$\Phi = \int_0^{\infty} F \left[e[t],\ t \right] dt$$

where F is a function of the system error and the time

It is known that the error can never be zero in a realistic system. The only way to find the optimum settings of an controller is to evaluate the minimum values of the error. If Φ is a function of the K_c, T_i and T_d, then the condition will be optimum when

$$\frac{\partial \Phi}{\partial K_c} = 0$$

$$\frac{\partial \Phi}{\partial T_i} = 0$$

$$\frac{\partial \Phi}{\partial T_d} = 0$$

With proportional control, the error will not be converged because of the steady state error. For integral action, the integral error can be converged, i.e. the ultimate error is zero. The tuning procedure of this method is more complicated since the closed-loop theoretical models of the controller and the robot is not easily to be formulated for optimisation.

The Continuous Cycle Method[97] is a classical method developed by Zielger and Nichols. It is also an efficient and practical method to tune a controller. First of all, the integral and derivative actions are taken away. The proportional gain K_c is increased gradually until the system starts to oscillate. At this time, the value of K_c is the ultimate gain K_u and the period of oscillation is P_u. According to Zielger's suggestion, the optimum settings for the PI control is the following,

$$K_c = 0.45\ K_u$$

$$T_i = \frac{P_u}{1.2} \qquad \ldots(5.9)$$

The optimum settings for the PID controller are,

$$K_c = 0.6\ K_u$$

$$T_i = \frac{P_u}{2.0}$$

$$T_d = \frac{P_u}{8.0} \qquad \ldots(5.10)$$

This method eliminates the tedious procedures of trial and error. Possibility of inaccurate model is also eliminated. It is therefore preferable to use this technique to tune the overall feedback controller.

5.3.2 Performance of the MT Model

Fig.5.3 shows the effect of synchronisation performance by increasing the value of K_c (0.5, 1.0, 1.25) without the integral and derivative actions. In the case of K_c to be 1.0 (fig.5.3a) , the synchronisation error starts from 1 mm. However, as time goes on, the error keeps on accumulating up to about 5.5mm at 15 seconds. By doubling the proportional gain, i.e. K_c = 1.0, there are greater restoring force to reduce the maximum error to 3.5mm (fig.5.3b) By increasing the K_c to 1.25, the system oscillates with increasing amplitude at a period of 0.4 second. Therefore, the values of K_u and P_u are set to be 1.25 and 0.4 sec. For an optimal PI controller by referring the equation (5.9), the values of K_c and T_i are,

$$K_c = 0.45 * 1.25 = 0.5625$$

$$T_i = 0.4 / 1.2 = 0.3333$$

For an optimal PID controller as stated in equation (5.10), the values of K_c, T_i and T_d are,

$$K_c = 0.6 * 1.25 = 0.75$$

$$T_i = 0.4 / 2.0 = 0.2$$

$$T_d = 0.4 / 8.0 = 0.05$$

Fig.5.4 shows that the PI control scheme can reduce the error but it is not completely eliminated. The graph shows that the synchronisation error fluctuates around ±1.0mm with an average steady state offset +0.5mm. The time required to arrive the steady state takes about 1.5 seconds. By using the PID setting in equation (5.10) ,fig.5.5 shows that the average 0 mm offset can almost be obtained and the settling time only takes about 0.75 seconds. However, the response has more oscillated ripples than the PI control scheme. Thus, eliminating the derivat'ye action would achieve better synchronisation performance. The reason of the ripples may be due to the inaccurate system delays of the overall control loop. To fine tune the system by modifying the optimal settings of K_c to 0.6 and T_i to 0.4 (fig.5.6), the noise ripples is slightly reduced. It fluctuates around ±0.5mm and the steady state offset is about 0.2mm.

5.3.3 Performance of the RT Model

By applying the same technique to the RT model, the system response with K_c=0.6 and T_i=0.4 is shown in the fig.5.7. The initial 10mm offset can be restored back to 1mm steady offset error within 6 seconds. Therefore, even though the improper combination of the

conveyor and robot velocities are selected, the large error due to the approximation of the elliptical integral can be easily compensated on-line by the overall feedback. Thus, a smooth trajectory and sufficient accuracy can be achieved by using the RT model by incorporating the overall feedback.

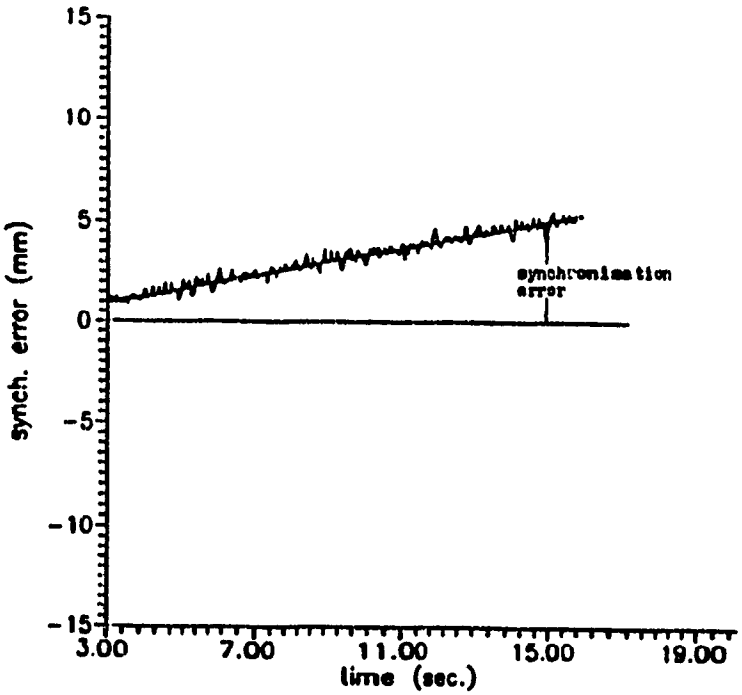


Fig.5.3a Synchronisation Response of MT Model
$K_c = 0.5$ $T_i = 100000$ $T_d = 0.0$

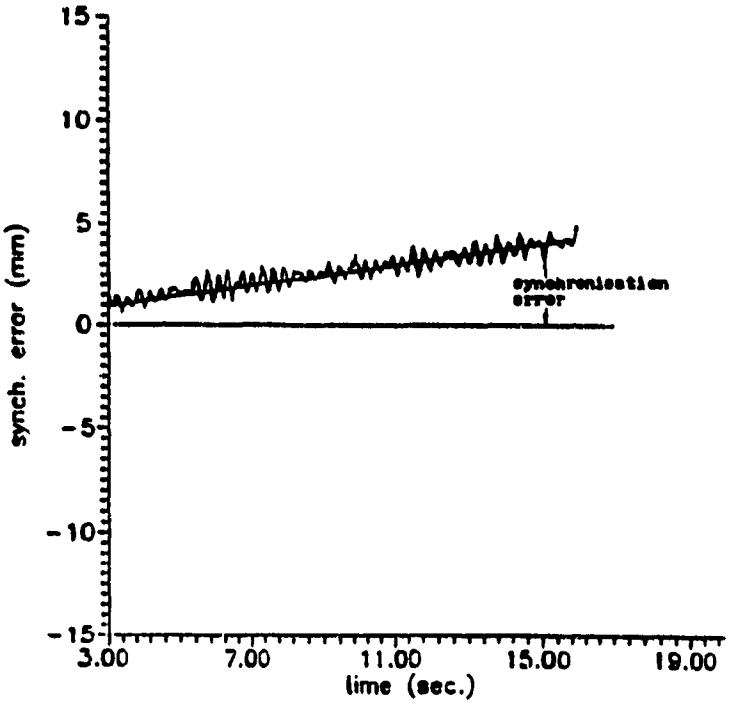


Fig.5.3b Synchronisation Response of MT Model
$K_c = 1.0 \quad T_i = 100000 \quad T_d = 0.0$

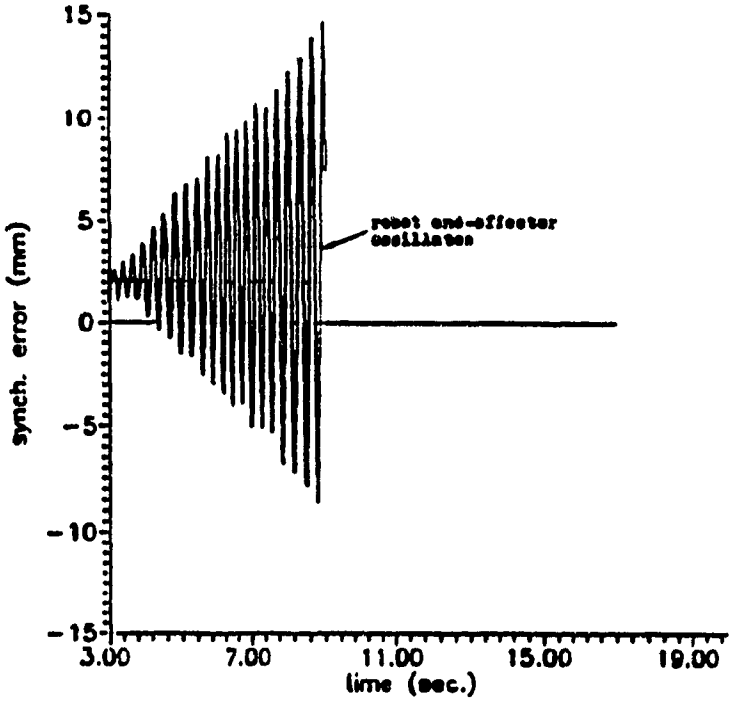


Fig.5.3c Synchronisation Response of MT Model
$K_c = 1.25 \quad T_i = 100000 \quad T_d = 0.0$

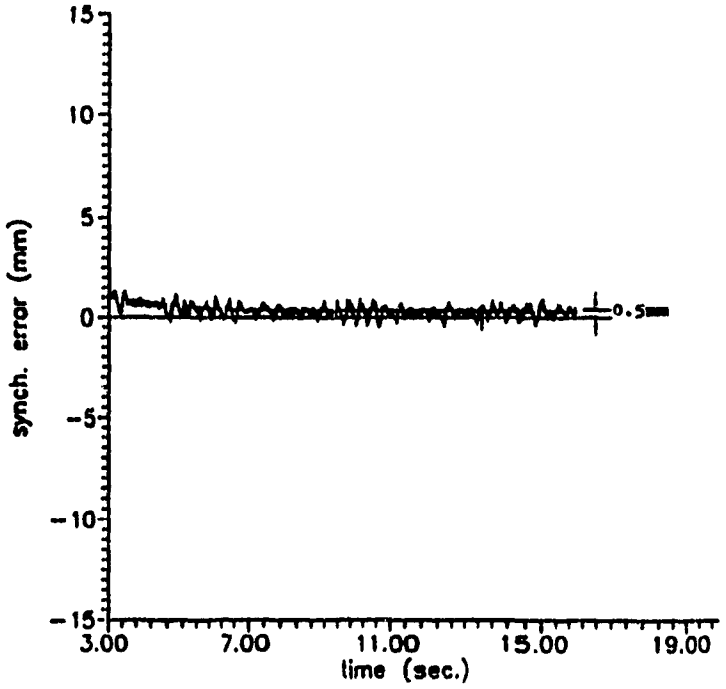


Fig.5.4 Synchronisation Response of MT Model
K_c = 0.5625 T_i = 0.3333 T_d = 0.0

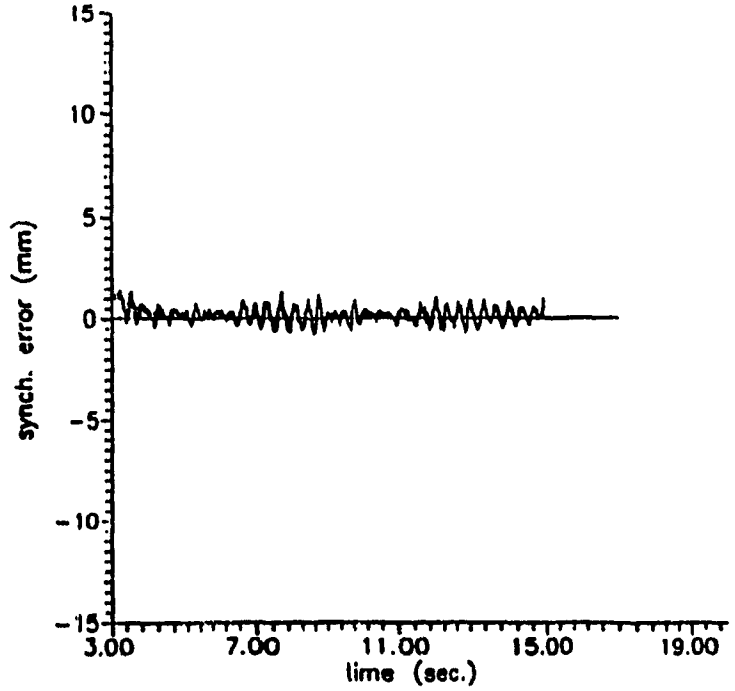


Fig.5.5 Synchronisation Response of MT Model
K_c = 0.75 T_i = 0.2 T_d = 0.05

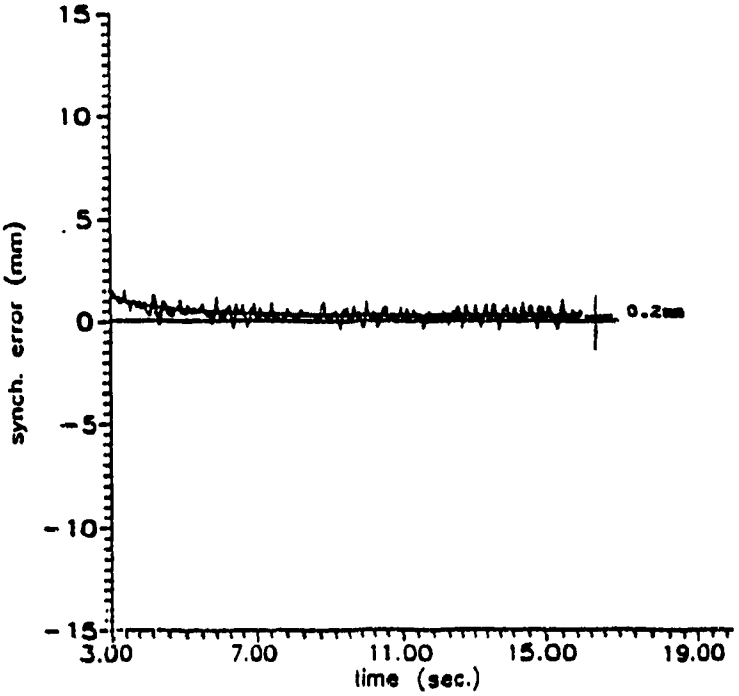


Fig.5.6 Synchronisation Response of MT Model
$K_c = 0.6 \quad T_i = 0.4 \quad T_d = 0.00$

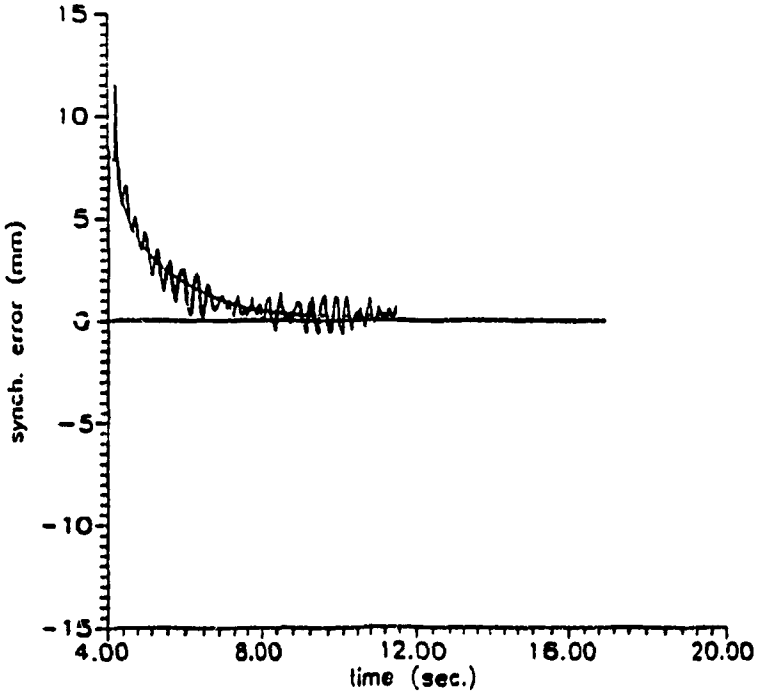


Fig.5.7 Synchronisation Response of RT Model
$K_c = 0.6 \quad T_i = 0.4 \quad T_d = 0.0$

5.4 SUMMARY

This chapter has discussed the strategy of the overall control loop. The overall feedback control model has been formulated to compensate the accumulated synchronisation errors. The overall feedback controller has also been implemented by using the P, PI and PID control schemes. It has been shown that the P-scheme is not able to eliminate the offset error. The PID-scheme can eliminate the offset error to ±0.5mm, however, it turns out to have large noise ripples. By using the PI-scheme, the synchronisation error can be controlled within +0.2 mm with ±0.5mm noise ripples. The noise ripple may be due to the inaccurate system delays. By incorporating this feature in the hierarchical controller would have greater degree of adaptive containment to perform on-line assembly task with higher accuracy.

Chapter 6

CONCLUSION AND SUGGESTIONS FOR FUTURE WORK

This thesis has addressed the development of an multiprocessor based intelligent robotic workcell for on-line industrial material handling and task assembly application involving a conveyor line system. The transputer-based workcell hierarchical controller is able to handle multiple tasks in real-time operations as follows :

. It is able to detect the workpiece position and orientation in a simple and efficient way.

. It generates the spatial paths to intercept and ultimately synchronise with the moving workpiece. Two control models have been formulated and compared. The *'Minimum Time'* model takes a shortest path to achieve the rendezvous with the moving workpiece and then synchronise with it to perform productive action. It provides larger productive space. However, experiment shows that large torque is required in joint #1 to reverse the direction of the end-effector at the interception point. Thus, it is suitable to be used in small on-line assembly applications. For handling large industrial parts, the *'Reduced Torque'* model is recommended because its smooth spatial path only takes 23% of the joint #1 torque which is required in the MT model. In other words, the robot can sustain higher payload (≅ 500% more).

. The spatial path generator alone tends to accumulate the

synchronisation error as time goes on. In order to provide an accurate synchronisation, overall feedback is used. It is found that the PI scheme is the most suitable control law of the overall feedback controller. By following the Zielger and Nichols continuous method to tune the controller parameters, the synchronisat n performance is ultimately reduced to fluctuate within 1.0 mm error. The small synchronisation ripples are probably due to the inaccurate matching of the control timing delays of the motion controller and the hierarchical controller.

At the present time, the primary image data acquisition and processing technique is only limited to recognise the 2D workpiece. An additional camera can be incorporated to provide a stereo vision system to recognise the 3D workpiece so that the workcell can be applicable in wider areas of operations.

The synchronisation performance can be improved by two options. First, the control elements, including the hierarchical controller, robot controller and the robot, can be dynamically modelled by using the z-transform. The discrete control model may be more accurate to match the timing problem. By re-adjusting the spatial path control algorithm, hopefully the ripples can be minimised. However, the discrete model sometimes may not be easily formulated because of the limited accessibility of the motion controller. The second approach is to replace the motion controller by a customised unit, e.g. adding more transputers to carry out the motion control task. All the control algorithms are customised developed and tested. Under such situation, the system timings and the robot behaviour can be accurately predicted

and modelled. The ripples may probably be eliminated.

In the system has been presented, the workpiece is initially located with the stationary camera and subsequently tracked by the belt encoder. However, if the conveyor is subjected to vibrations which are often happened in industrial environments, the workpiece may have drifts relative to the conveyor. Therefore, additional adaptive features can be incorporated to detect such unpredictable position and orientation drifts. To provide a solution, a technique which is called the *incremental image processing* has been developed. The technique and its formulation are described in appendix I in details. The concept of the technique is to use a small auxiliary camera mounted on the robot wrist to detect certain invariant features of the moving workpiece. The features are then compared with those features in the primary image. The position differences of the features are then transformed into the position and orientation drifts. The preliminary implementation result can detect within 0.45° orientation drift and 1mm in the xy-drift under the situation that the velocity of the workpiece and the effector are zero. The complete implementation for detecting different cases of critical features is left for the future development.

For a long term development, the architecture of the hierarchical controller can be expanded by adding more transputers to form a multi-device and multi-robot workcell controller.

REFERENCES

[1] 3L Ltd., Parallel C User Guide, pp.197-198, 1988.

[2] 3L Ltd., Parallel C User Guide, pp.39-62, 1988.

[3] 3L Ltd., Parallel C User Guide, pp.177-181, 1988.

[4] Agin, G.J., "An Experimental Vision System for Industrial Application", *Proc. 5th ISIR*, 1975.

[5] Aken, L.V., and Brussel, H.V., "On-Line Robot Trajectory Control in Joint Coordinates by Means of Imposed Acceleration Profiles", *Proc. 15th ISIR* , vol. 2, pp.1003-1010, 1985.

[6] Amin-Javheri, M., and Orin, D.E., "A Systolic Architecture for Computation of the Manipulator Inertia Matrix", *Proc. IEEE Intl. Conf. Robotics and Automation*, vol.2, pp.647 - 653, 1987.

[7] Andersson, R.L., "Computer Architectures for Robot Control : A Comparsion and A New Processor Delivering 20 Real Mflops", *Proc. IEEE Intl. Conf. Robotics and Automation*, vol.2, pp.1162-1167, 1989.

[8] Armbruster, K., et. al., "A Very Fast Vision System for Recognition Parts and their Location and Orientation", *Proc. 9th ISIR*, pp.265-280, 1979.

[9] Bernardon, E., and Kondoleon, T.S., "Real Time Robotic Control for Apparel Manufacturing", *Proc. Robots 9 Advancing Applications*, vol.1, pp.4-46 - 4-66, 1985.

[10] Brady, M., et. al., Robot Motion : Planning and Control, Cambridge : MIT Press, 1982, p.27.

[11] Brady, M., et. al., Robot Motion : Planning and Control, Cambridge : MIT Press, 1982, pp.64-66.

[12] Brand, L., Advanced Calculus : An Introduction to Classical Analysis, NY : Wiley, 1962, pp.286-288.

[13] Bribiesca, E., and Guzman, A., "How to Describe Pure Form and How to Measure Differences in Shapes using Shape Numbers", *Pattern Recognition*, vol.12, pp.101-112, 1980.

[14] Burden, R.L., Faires, J.D., and Reynolds, A.C., Numerical Analysis, Boston : Prindle, Weber & Schmidt, 2nd ed., 1981, pp.125-133.

[15] Carlisle, B., "The PUMA/VS-100 Robot Vision System", *Proc. 1st Intl. Conf. on Robot Vision and Sensory Controls*, pp.149-160, 1981.

[16] Castain, R.H., and Paul, R.P., "An On-line Dynamic Trajectory Generator", *Int. J. Robotic Res.*, vol. 3, no. 1, pp.68-72, 1984-86.

[17] Chand, S., and Doty, K.L., "On-Line Polynomial Trajectories for Robot Manipulators," *Int. J. Robotic Res.*, vol. 4, no.2, pp.38-48, 1985.

[18] Chen, J.B., et. al., "NYMPH : A Multiprocessor for Manipulation Applications", *Proc. IEEE Intl. Conf. Robotics and Automation*, vol.3, pp.1731-1736, 1986.

[19] Cheng, R.M.H., and Fahim, A.E., "On-line System for Identifying the Angular Orientation of a Class of Engineering Components", *Industrial Elect. & Control Systems*, vol. IECI-17, #3, IEEE Transactions, 1980.

[20] Cheng, R.M.H., Lequoc, S., and Athanasoulias, D., "An On-line System for Locating the Position of the Centroid of a Flat Object", *Proc. of the 9th Canadian Congress of Applied Mechanics (Univ. of Sas., Saskatoon)*, 1983.

[21] Cheng, R.M.H., and Montor, T., "Synchronisation Control of an Industrial Robotic Manipulator Using Camera Vision", *Proc. Intl.*

Conf. Intelligent Autonomous Systems, pp.157-161, 1986.

[22] Cheng, R.M.H., and Montor, T., "Application of a Low Cost Vision System to Automatic Assembly", *Proc. IFAC Low Cost Automation*, 1986.

[23] Cheng, R.M.H., and Poon, S.C.L., "Architecture of an Intelligent Robotic Workcell for Synchronisation Control", *Proc. IASTED*, pp.155-160, 1988.

[24] Cheng, R.M.H., Montor, T., and Poon, S.C.L., "Adaptive Synchronisation Control of a Robotic Manipulator Operating in an Intelligent Workcell", *accepted by IEEE Transaction of Industrial Electronic on 25 Sept.89.*

[25] Choi, Y.K., Bien, Z., and Youn, M.J., "A Path Planning Algorithm for Positioning Manipulator End-Effector in Cartesian Space Using Circular Interpolation", *Proc. 15th ISIR* , vol. 1, pp.1031-1040, 1985.

[26] Cook, C.C., and Ho, C.Y., "The Application of Spline Functions to Trajectory Generation for Computer-Controlled Manipulators", Computing Techniques for Robots, Igor Aleksander (Ed), pp, 102-110, 1985.

[27] Corripio, A.B., "Controller Tuning from Simple Process Models", *Instrumentation Technology*, Dec. 1975.

[28] Craig, J.J., Introduction to Robotics : Mechanics and Controls, Reading : Addison-Wesley Publishing Co., 1986.

[29] Driels, M., et. al., "The Use of Visual Feedback for the Acquisition of Pseudorandomly Oriented Parts", *J. Robotic Systems*, pp.195-204, 1984.

[30] Esfahani, M., and Radharamanan, R., "A Fast Recursive Algorithm for Contour Extraction of Binary Images in Robotics Application",

Proc. IASTED, pp.12-16, 1988.

[31] Freeman, H. "Shape Description Via the Use of Critical Points", *Pattern Recognition*, vol.10, pp.159-166, 1978.

[32] Froberg, C.E., Introduction to Numerical Analysis, Reading : Addison- Wesley Publishing Co.,2nd ed., 1969, pp.335-357.

[33] Gershon, D., Porat, I., "Vision Servo Control of a Robotic Sewing System", *IEEE Intl. Conf. Robotics and Automation*, vol.3, pp.1830-1835, 1988.

[34] Hashizume, A., Yeh, P.S., and Rosenfeld, A., "A Method of Detecting the Orientation of Aligned Components", *Pattern Recognition Letters*, vol.4, no.2, Apr. 1986.

[35] Hayward, V, and Paul, R.P., "Robot Manipulator Control Under UNIX", *Proc. 13th ISIR*, vol.2, pp.20-32 - 20-44, 1983.

[36] Holland, S.W. et al, CONSIGHT-I : A Vision-Controlled Robot System for Transferring Parts from Belt Conveyors, Warren, Michigan, internal report no. : GMR-2790, 1978.

[37] Homewood, M., et al., "The IMS T800 Transputer", *IEEE MICRO*, Oct., 1987.

[38] Horaud, R., Olympieff, S., and Charras, J.P., "Shape and Position Recognition of Mechanical Parts from their Outlines", *Proc. 1st Intl. Conf. Robot Vision and Sensory Controls*, pp. 125-134, 1981.

[39] Inmos Corp., Preliminary Data IMS T800 transputer, Mar., 1988.

[40] Inmos Corp., Engineering Data IMS C012 Link Adaptor, Aug. 1987.

[41] Inmos Ltd., OCCAM 2 Reference Manual, Cambridge : Prentice Hall Int., 1988.

[42] Ish-Shalom, J., and Kazanzides, P., "SPARTA : Multiple Processors for High-Performance Robotic Control", *Proc. IEEE Intl. Conf. Robotics and Automation*, vol.1, pp.284-290, 1988.

[43] Jahnke, E., and Emde, F., Table of Functions with Formulae and Curves, Leipzig : B.G. Teubner, 1933, pp.140-144.

[44] Kammenos, P., "Performances of Polar Coding for Visual Localisation of Planar Objects", *Proc. 8th ISIR*, 1978.

[45] Kazanzides, P., Wasti, H., and Wolovivh, W.A., "A Multiprocessor System for Real-Time Robotic Control : Design and Applications", *Proc. IEEE Intl. Conf. Robotics and Automation*, vol.3, pp.1903-1908, 1987.

[46] Khalil, W., "Trajectories Calculations in the Joint Space of Robots", Advanced Software in Robotics, Danthine/Geradin : Elsevier Science Publishers B.V., 1984.

[47] King, F.G., Puskorius, G.V., and Yuan, F., "Vision Guided Robots for Automated Assembly", *Proc IEEE Intl. Conf. Robotics and Automation*, vol.3, pp.1611-1616, 1988.

[48] Kossman, D., and Malowany, A., A Multiprocessor Robot Control System, McGill research Centre for Intelligent Machines, McGill University, internal report, Sept., 1986.

[49] Kyriakopoulos, K.J., and Saridis, G.N., "Minimum Jerk Path Generation",*Proc. IEEE Intl. Conf. Robotics and Automation*, vol.1, pp.364-369, 1988.

[50] Leahy, M.B. Jr., and Saridis, G.N., "The RAL Hierarchical Control System", *Proc. IEEE Conf. Robotics and Automation*, pp.407-411, Apr. 1986.

[51] Lenarcic, J., Oblak, P., and Stanic, V., "Approximate Calaulation of Robot Joint Trajectories for Control Along Cartesian Paths", *Proc. 15th ISIR*, vol.1, pp.1021-1030, 1985..

[52] Liebes, S., Gong, W., and Gray, A., "FLAIR-Robotic Printed Circuit Board Assembly Workcell", *Proc IEEE Intl. Conf. Robotics*

and Automation, vol.3, pp.1594-1599, 1988.

[53] Lin, F.S., Chang, P.R., and Luh, J.Y.S. ,"Formulation and Optimization of Cubic Polynomial Joint Trajectories for Industrial Robots", *IEEE Trans. Automat. Contr.*, vol. AC-28, no.12, pp.1066-1073, 1983.

[54] Lin, C.S., and Chang, P.R., "Joint Trajectories of Mechanical Manipulators for Cartesian Path Approximation", *IEEETrans. Systems, Man, and Cybernetics*, vol.SMC-13, no.6, pp.1094-1102, 1983.

[55] Ling, Y.L.C., et. al., "A VLSI Robotics Vector Processor for Real-Time Control", *Proc. IEEE Intl. Conf. Robotics and Automation*, vol.1, pp.303-308, 1988.

[56] Luh, J.Y.S., and Lin, C.S., "Optimium Path Planning mechanical Manipulators", *J. Dynamics Sys., Measurment, and Control*, vol.102, pp.142-151, 1981.

[57] Luh, J.Y.S., "Approximate Joint Trajectories for Control Industrial Robots Along Cartesian Paths", *IEEE Trans. Systems, Man, and Cybernetics*, vol. SMC-14, no. 3, pp. 444-450, 1984.

[58] Luyben, W.L., Process Modelling, Simulation Control for Chemical Enginneering, N.Y. : McGraw-Hill Inc., 1973 , p.320.

[59] Makhlin, A.G., "Grey Scale Robot Vision for Realtime Inspection and Assembly", *Proc. 13th ISIR*, vol.2, pp. 17.36 - 17.46, 1983.

[60] Mangaser, A.A., Wang, Y., and Butner, S.E., "Concurrent Programming Support for a Multi-Manipulator Experiment on RIPS", *Proc. IEEE Intl. Conf. Robotics and Automation*, vol.2, pp.1162-1167, 1989.

[61] May, D., Communicating Processes and Occam, *Inmos Transputer Technical Note #20*, 1988.

[62] Mckee, J.W., and Aggarwal, J.K., "Computer Recognition of Partial Views of Curved Objects", *IEEE Trans. Computers*, vol. C-22, Sept. 1977.

[63] Micron Technology Inc., Idetix Digital Vision System Operator's Manual, 1986.

[64] Micron Technology Inc., MicronEye Operator's Manual, 1983.

[65] Mo, X.J., Liu, L.Z., "A Robot System with Vision, Touch and Slide Senses for the Grip onto a Moving Conveyor Belt", *Proc. 15th ISIR*, pp.129-136, 1985.

[66] Mochizuki, J., Takahashi, M., and Hata, S., "Unpositioned Workpieces Handling Robot with Visual and Force Sensors", *IEEE Tran. Industrial Electronics*, vol. IE-34, no.1, pp.1-4, Feb. 1987.

[67] Montor, T., The Development and Analysis of an Intelligent Robotic Workcell, M.Eng. Thesis, Department of Mechanical Engineering, Concordia University*(Montreal, Canada)*, 1988, pp.9-10.

[68] Montor, T., The Development and Analysis of an Intelligent Robotic Workcell, M.Eng. Thesis, Department of Mechanical Engineering, Concordia University*(Montreal, Canada)*, 1988, p.14.

[69] Montor, T., The Development and Analysis of an Intelligent Robotic Workcell, M.Eng. Thesis, Department of Mechanical Engineering, Concordia University*(Montreal, Canada)*, 1988, p.56.

[70] Montor, T., The Development and Analysis of an Intelligent Robotic Workcell, M.Eng. Thesis, Department of Mechanical Engineering, Concordia University*(Montreal, Canada)*, pp.84, 1988.

[71] Montor, T., The Development and Analysis of an Intelligent Robotic Workcell, M.Eng. Thesis, Department of Mechanical

Engineering, Concordia University(*Montreal, Canada*), 1988, pp. 102-103.

[72] Montor, T., The Development and Analysis of an Intelligent Robotic Workcell, M.Eng. Thesis, Department of Mechanical Engineering, Concordia University(*Montreal, Canada*), 1988, p. 117.

[73] Montor, T., The Development and Analysis of an Intelligent Robotic Workcell, M.Eng. Thesis, Department of Mechanical Engineering, Concordia University(*Montreal, Canada*), 1988, p. 147-159.

[74] Murrill, P.W., "Tuning Controllers with Error-Integral Criteria", *Instrumentation Technology*, Nov., 1967.

[75] Nagao, M., "Shape Recognition by Human-Like Trail and Error Random Processes", Robotic Research 2nd International Symposium, MIT Press, 1985.

[76] Nagy, P., and Lim, K.B., An Optical Encoder Conditioning/Counting Circuit, Department of Mechanical and Production Engineering, National University of Singapore, Technical Report TR-ME-004-CON-86, 1986.

[77] Nagy, P.V., Tracking Control of an Industrial Robot, M.Eng. Thesis, Department of Mechanical and Production Engineering, National University of Singapore, 1987.

[78] Nayak, N., et. al., "Conceptual Development of an Adaptive Real-Time Seam Tracker for Welding Automation", *IEEE Intl. Conf. Robotics and Automation*, vol.2, pp. 1019-1024, 1987.

[79] O'Hara, D.H., Elgazzar, S., and The, G., ALTER-Harmony : Control of aPUMA Robot from the Chrous Multiprocessor, Division of Electrical Engineering, National Research Council Canada, internal report NRCC no. 28492.

[80] Ozguner, F., and Kao, M.L., "A Reconfigurable Multiprocessor Architecture for Reliable Control of Robotic Systems", *Proc. IEEE Intl. Conf. Robotics and Automation*, pp.802-806, 1985.

[81] Paramax Corp. LC-1 RS-232 to Transputer Link Converter Module User Manual, 1988.

[82] Paul, R., "Cartesian Coordinate Control of Robots in Joint Coordinates", *Proc. 3rd Intl. Symposium on Theory and Practice of Robots and Manipulators*, pp.228-250, 1978.

[83] Paul, R., "Manipulator Cartesian Path Control", *IEEE Trans. Systems, Man and Cybernetics*, vol. SMC-9, no. 11, pp.702-711, 1979.

[84] Paul, R.P. and Zhang, H., "Robot Motion Trajectory Specification and Generation", Robotics Research 2, Hanafuse and Inous (eds), MIT Press, Massachusetts, pp.187-193, 1985.

[85] Perkins, W.A., "A Model-Based Vision System for Industrial Parts", *IEEE Trans. Computers*, vol.C-27, no.2, pp.126-143, 1978.

[86] Pfeiffer, F., and Johanni, R., "A Concept for Maniputlator Trajectory Planning", *IEEE J. Robotics and Automation*, vol.RA-3, no.2, pp.115-123, Apr.1987.

[87] Pollard A., Process Control, N.Y. : American Elsevier Publishing Co.Inc., 1971 , p.179-181.

[88] Rajagopalan, R., Parallel Computation of Kinematics and Dynamics of Robot Arms and Mobile Robots Employing Transputers, Dept. of Mechanical Engineering, Concordia Univeristy,Montreal, Canada, internal report #CIC - 0023.

[89] Rajagopalan, R., A Computer-Aided Methodolgy for the Calibration of Industrial Robots Employing Optical Metrology, M.Eng. Thesis, Department of Mechanical Engineering, Concordia University, 1986.

[90] Rao, K., et. al., "Robot Hand-Eye Coordination : Shape Description and Grasping", *Proc. IEEE Intl. Conf. Robotics and Automation*, vol.1, pp.407-411, 1988.

[91] Redfern, S., Implementing Data Structures and Recursion in Occam, *Inmos Transputer Technical Note #38*, 1988.

[92] Sahar, G., and Hollerbach, J.M., "Planning of Minimum-Time Trajectories for Robot Arms", *Intl. J. Robotics Res.*, vol.5, no.3, pp.90-100, 1986.

[93] Shahinpoor, M., and Abdel-Rahman, S.Z., "Generalized Algorithm for Computing the 4-3-4 N-Axis Robotic Trajectory", *J. Robotic Syst.*, vol. 1, no. 4, pp.395-420, 1984.

[94] Shin, K.G., and McKay, N.D., "Minimum-Time Trajectory Planning for Industrial Robots with General Torque Constraints", *Proc. IEEE Intl. Conf. Robotics and Automation*, vol.1, pp.412-417, 1986.

[95] Shin, K.G., and McKay, N.D., "Selection of Near-Minimum Time Geometric Paths for Robotic Manipulators", *IEEE Trans. Automatic Control*, vol.AC-31, no.6, June 1986.

[96] Smith, C.L., and Murrill, P.W., "Analytical Tuning of Underdamped Systems", *ISA Journal*, Sept. 1966.

[97] Smith, C.L., "Controller Tuning that Works", *Instruments and Control Systems*, Sept. 1976.

[98] Takanashi, N., Ikeda, T., and Tagawa, N., "A High-Sample-Rate Robot Control System Using A DSP Based Numerical Calculation Engine", *Proc. IEEE Intl. Conf. Robotics and Automation*, vol.2, pp.1168-1173, 1989.

[99] Taylor, G.E., et. al., "Vision applied to the orientation of Embroidered Motifs in the Textile Industry", *Proc. Intl. Conf.*

Robot Vision and Sensory Controls, pp.39-46, 1982.

[100] Taylor, R.H., "Planning and Execution of Straight Line Manipulator Trajectories", *IBM J. Research and Development*, pp. 424-436, 1979.

[101] Torgerson, E., and Paul, F.W., "Vision-Guided Robotic Fabric Manipulation for Apparel MAnufacturing", *IEEE Control System*, pp.14-20, Feb. 1988.

[102] Ueda, M., Matsuda F., Takahara, Y., "A Real Time Shape Recognition of Moving Objects", *Proc. 11th ISIR*, pp. 129-136, 1981.

[103] Ward, M., et. al., "Production Plant Consight Installations", General Motors Research Publications, report no. GMR-4156, Aug. 1982.

[104] Watanabe, T., et. al., "Robot Manipulator Control by Multi-Microprocessors", *Proc. Japan-U.S.A. Sym. on Flexible Automation*, pp.189-194, 1986.

[105] Yen, P.S., "A Vision System for Safe Robot Operation", *Proc. IEEE Intl. Conf. Robotics and Automation*, vol.3, pp.1461-1465, 1988.

[106] Zakaria, M.F., et. al., "Fast Algorithm for the Computation of Moment Invariants", *Pattern Recognition*, vol.20, pp.639-643, 1987.

[107] Zheng, Y.F., and Chen, B.R., "A Multiprocessor for Dynamic Control of Multilink Systems", *Proc. IEEE Intl. Conf. Robotics and Automation*, pp.295-300, 1985.

TABLE OF APPENDICES

Appendix	Title	Related Section /Appendix
A	Register Address and Flowchart of the Idetix Camera Software Driver	2.1.1.2
B	Joint Feedback Data Acquisition Circuitry	2.4.2
C	Comparison of Common Multiprocessor Systems	3.4
D	Limiting Speed Ratios between Robot and Workpiece of MT Model	4.3.1.1
E	Numerical Solution of the Elliptical Integral	4.3.2
F	Numerical Differentiation to Evaluate Joint Velocities and Accelerations	4.4.2
G	Newton-Euler Inverse Dynamics Formulations	4.4.2
H	Forward Kinematics of the PUMA 560 Robot	5.2.1
I	Incremental Image Processing	chapter 6
J	Architecture of an Intelligent Robotic Workcell for Synchronisation Control	I.1
K	Auxiliary Camera Transeceiver/Strobe Light Circuit	I.3
L	Classes of Industrial Workpieces	I.4

APPENDIX A

Register Address And Flowchart Of The Idetix Camaera Software Driver

Fig.A.1 shows a software driver to input an image from the idetix camera. Fig.A.2 shows a software driver to reset the camera controller. Fig.A.3 shows a software driver how the camera controller parameters are actually downloaded. It has to be sured that the camera is not in transferring process before any downloading. There are two kinds of downloading process. One is to download the command register to take single frame, continuous frame or to stop the camera. Another one is to download a complete set of parameters (18 bytes) to the controller. A service bit is used to indicate that the controller is free to be downloaded. The default settings of the parameters are listed in Table A.1. The bit settings of the command and status register are listed in Tables A.2 and A.3. The camera controller uses the Direct Memory Access(DMA) method to get the image into the memory. However, the DMA method does not allow the data buffer across the boundary of two 64k segments. Therefore, fig.A.4 shows a technique to allocate two buffers. In case of one buffer across the boundary, another buffer will be assigned. Fig.A.5 shows the procedures to initialise the DMA registers and DMA channels.

TABLE A.1 IDETIX CAMERA CONTROLLER PARAMTERS

BYTE #	DEFINITION	DEFAULT SETTINGS
1	Command Register	0x12
2	Refresh Start Address	0x0000
3	LSByte Refresh Last Address	0xfe
4	MSByte Refresh Last Address	0x00
5	LSByte Row Start Address	0x81
6	MSByte Row Start Address	0x00
7	Row Address Increment	0x02
8	LSByte Row Last Address	0xff
9	MSByte Row Last Address	0x00
10	LSByte Column Start Address	0x00
11	MSByte Column Start Address	0x00
12	Column Address Increment	0x02
13	LSByte Column Last Address	0x00
14	MSByte Column LAst Address	0x01
15	Soak Strobe Count	5
16	Least Significant Byte Exposure Time	0x50
17	Most Significant Byte Exposure Time	0x00
18	Camera Head MUX Select (None = 0x0f Hex)	0x0f

TABLE A.2 ADDRESS 0X260 - COMMAND REGISTER AND DATA SET

Bit	Indication
1	0 = Single Frame Off, 1 = Single Frame On
2	0 = Continuous Frame Off, 1 = Continuous Frame On
4	0 = Low Exposure Range (.1 ms to 6.5 sec.)
	1 = High Exposure Range (10 ms. to 600 sec.)
6	1 = Sequencer Test Mode On
10	0 = Download Command Register Only
	1 = Data Download to Follow
20	1 = Enable Strobe Input Mode
40	1 = PC Test Vector Mode On
80	not used

Address 261 HEX - TEST DATA INPUT

TABLE A.3 Address 260 HEX - Status Register

Bit	Indication
1	1 = Register Byte Serviced (A 70 μs delay must be implemented after this bit goes high before writing the next byte)
2	1 = Command Set Download Complete
4	Transfer Complete
8	0 - IS32A, 1 = IS256
10	not used
20	not used
40	not used
80	not used

Address 261 HEX - Pixel Count Register (Nibbles 1 and 2)
Address 262 HEX - Pixel Count Register (Nibbles 3 and 4)
Address 263 HEX - Pixel Count/Address Register
Address 264 HEX - Test Data Output Register
Address 265 HEX - Reset Hardware

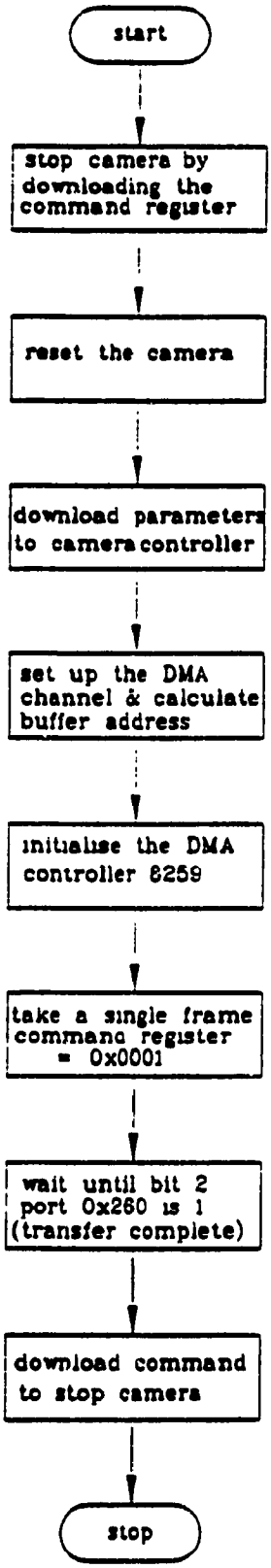


Fig.A.1 Software Driver to Activate the Idetix Camera

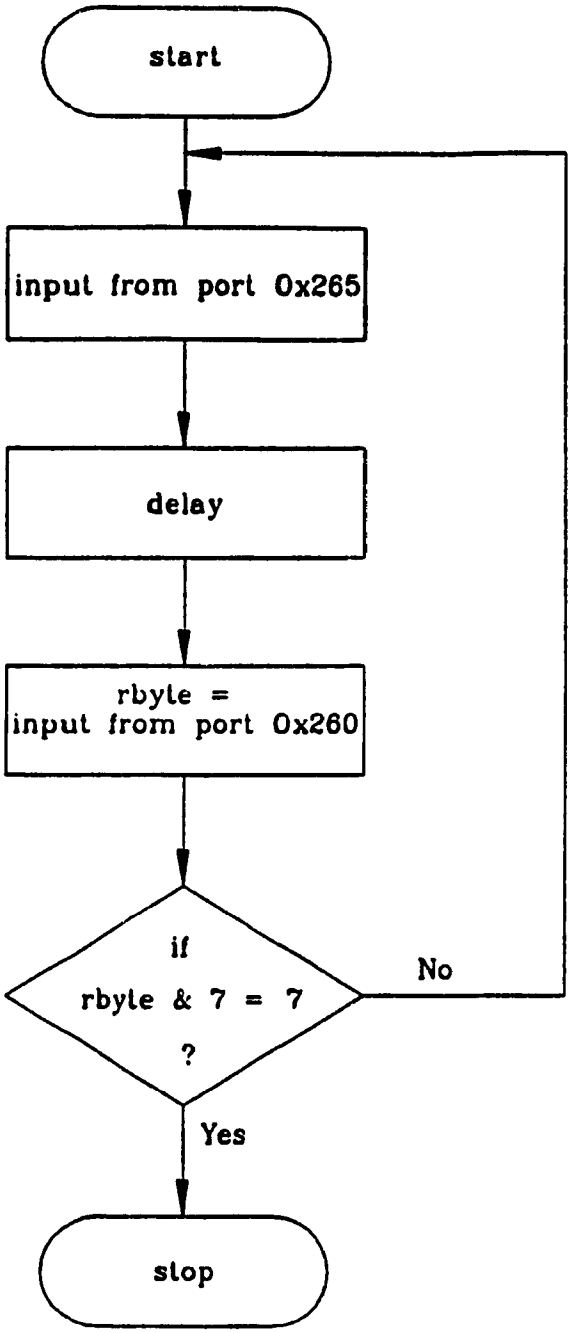


Fig.A.2 Software Driver to Reset the Camera Controller

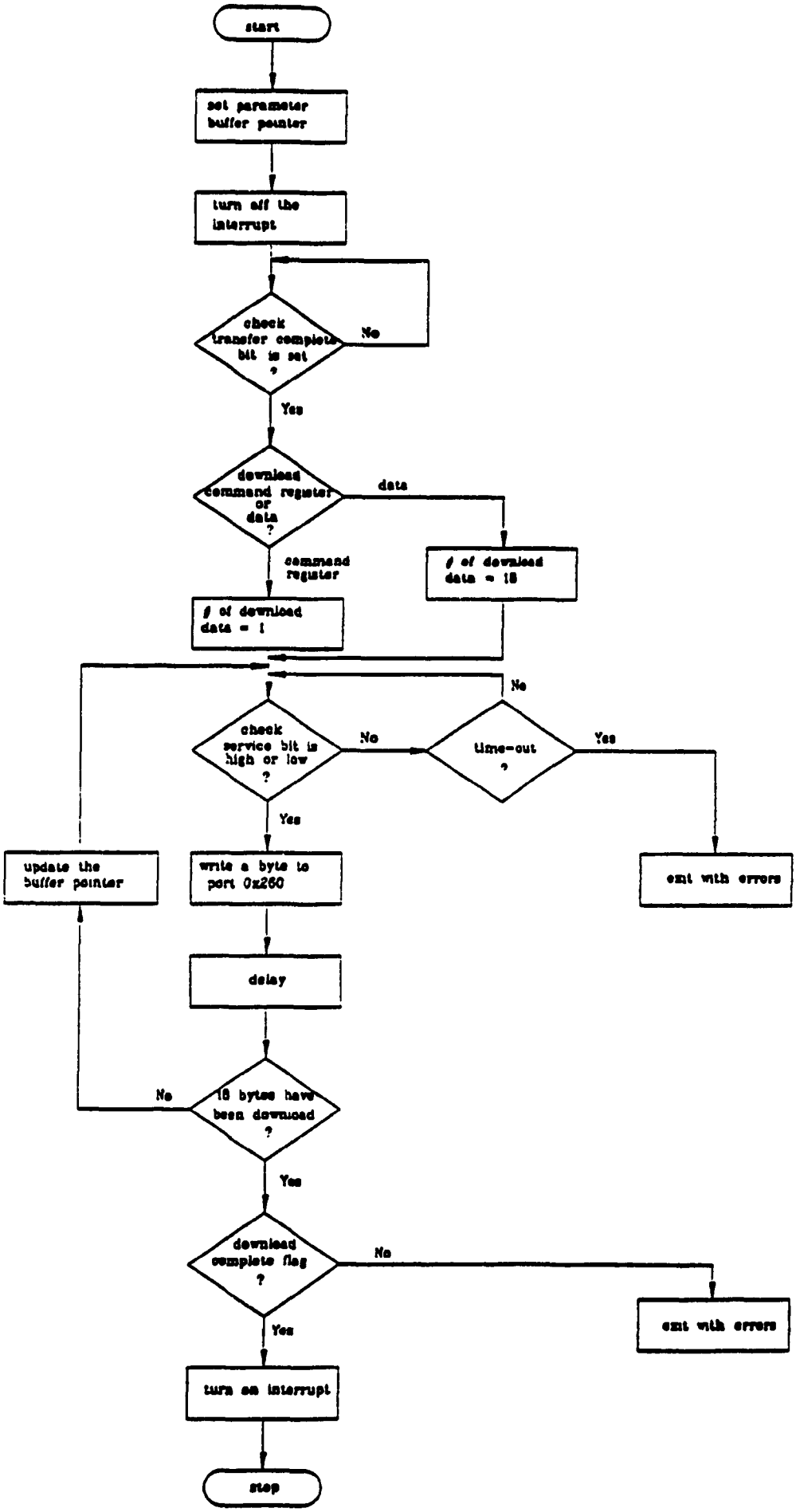


Fig.A.3 Software Driver to Download Parameters to the Camera Controller

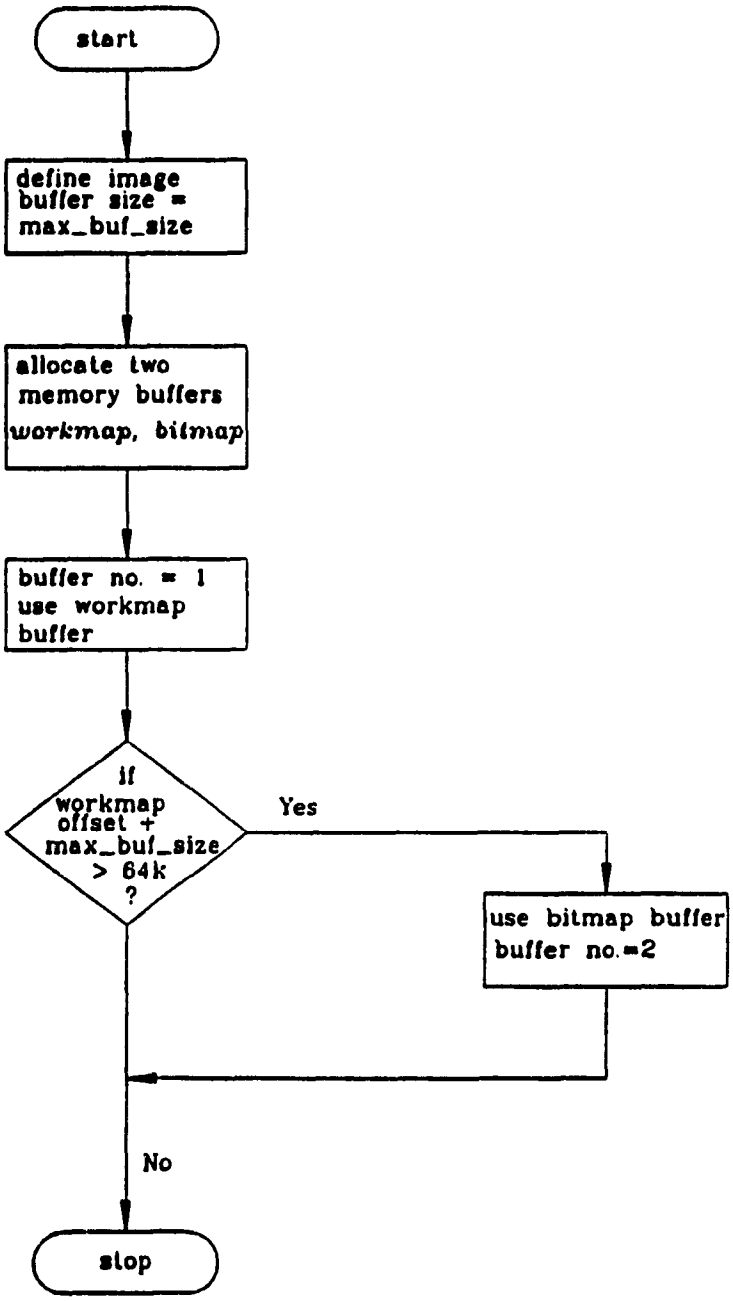


Fig.A.4 Software Driver to Check DMA Buffer

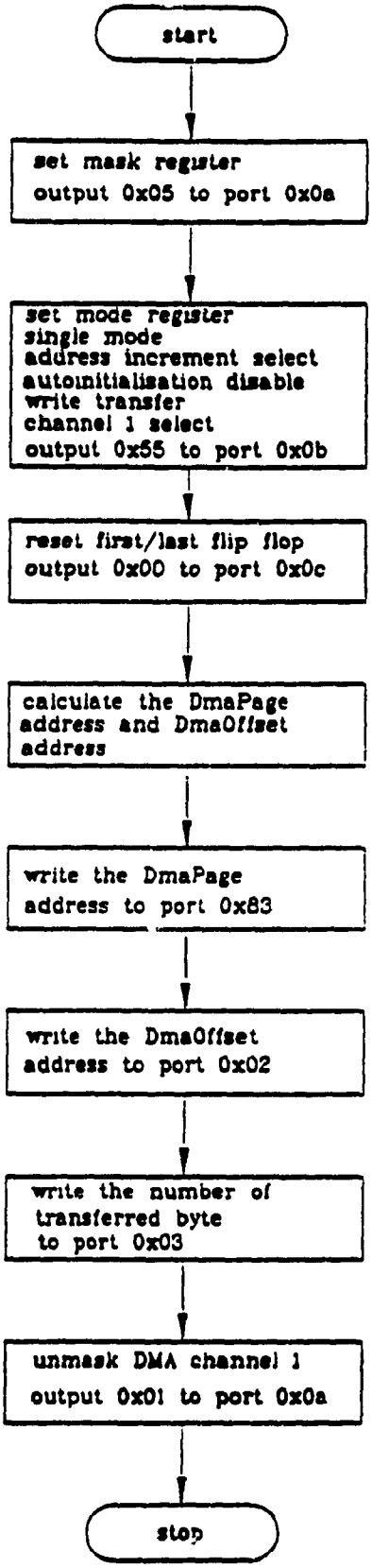


Fig.A.5 Software to Initialise the DMA Channel

APPENDIX B

Joint Feedback Data Acquisition Circuitry

B.1 INTRODUCTION

The purpose of this circuitry is to obtain the robot joint angles in real-time. The joint angles are transformed into cartesian coordinate by a forward kinematic model so that the synchronisation offset can be compensated. The circuitry is composed of the following modeules :

(a) Pulse Conditioning circuit (fig.B.1)

(b) Direction Decoding Circuit (fig.B.2)

(c) Counting Circuit (fig.B.4)

(d) Address Decoding Circuit (fig.B.5)

The data acquisition consists of six modules (a), (b) and (c) for the corresponding six robot joints.

B.2 ENCODER PULSE CONDITIONING CIRCUIT

As shown in the fig.B.1, the output of the encoder A, B are connected to the buffer followers U1A, U1B, U1C and U1C. Between the followers are the low pass filters consisted of R3, C1, R8 and C2. This module has two purposes - isolation and filtering.

Since the joint encoders are connected to the motion controller to carry out the joint motion control, this module is used to provide a high impedance to isolate the data acquisition circuitry and the motion controller. Otherwise, the sink-current back to the robot controller would create unpredicted robot motion. The second reason is to provide the filters to take away the high frequency unwanted signals. The low pass filter limits the bandwidth of the signal sent to the comparators U2 and U3 to approximately 144 KHz. The comparator

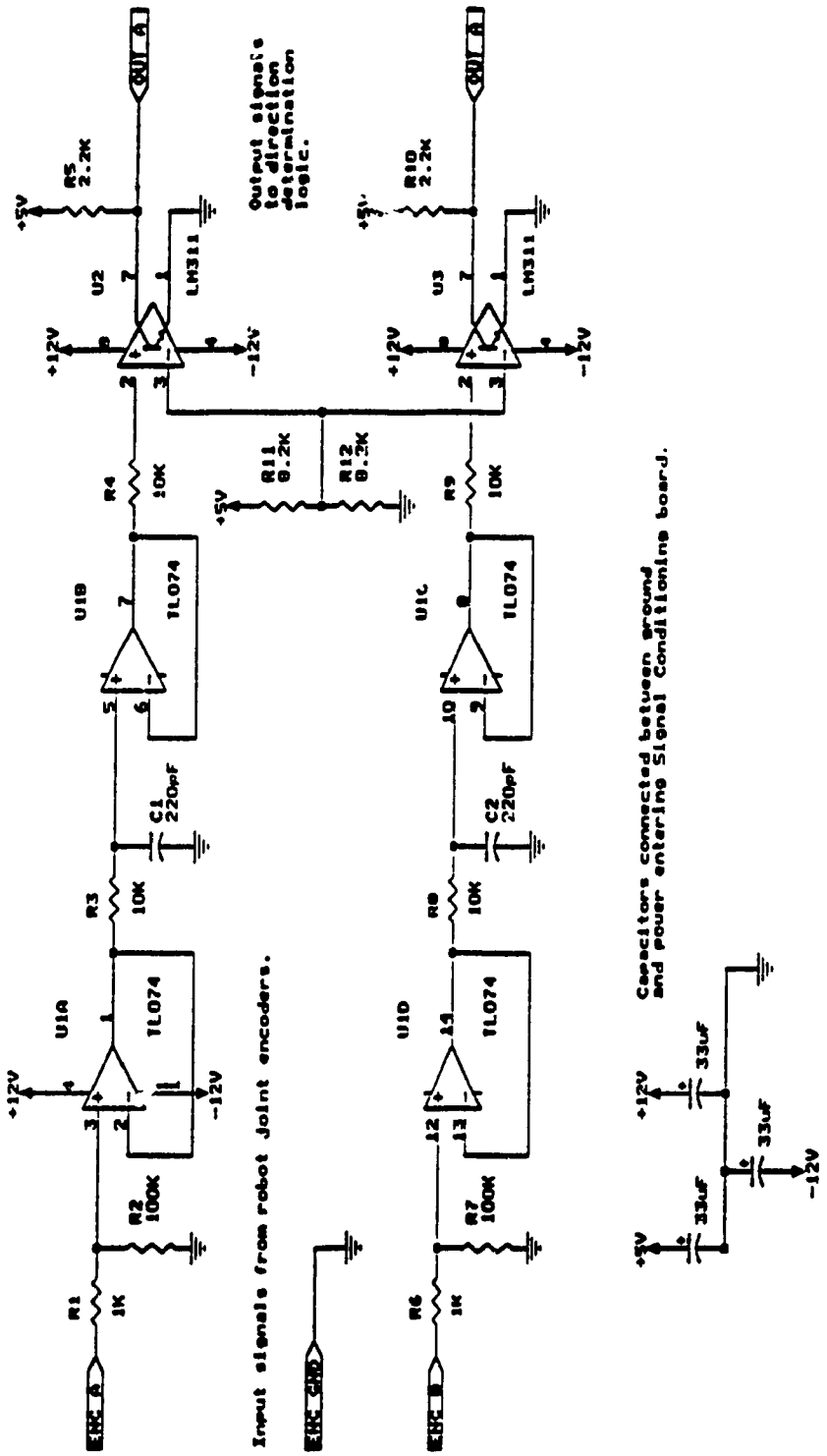


Fig.B.1 Pulse Conditioning Circuit

compares its input signal to a 2.5 volt reference and provides 5 volt output if the input is greater than 2.5 volts, or 0 volt if the signal is less than 2.5 volts.

B.3 DIRECTION DECODING CIRCUIT

The direction decoding circuit (fig.B.2)[76] is a logic to give count-up or count-down signal to the counters according to the lead or lag of the pulse train A and B. Fig.B.3a shows the pulse train of A and B in one direction and fig.B.3b in the opposite direction. The Table B.1 shows the inputs of the joint encoder pulse trains to have the same direction signs with the PUMA 560 robot controller.

B.4 COUNTING CIRCUIT

There are six 16-bit counters (fig.B.4) for each joint to measure the joint angles in real-time. Four binary up/down counters LS193 are casacaded to form a 16-bit counter for each joint. The counters can be preset. The up and down pins of the least significant nibble counter are connected to the direction decoding circuit. The value of the counter is hold in two LS373 latches while in the reading operation. The resolution of the counter for each joint are shown in the Table B.2.

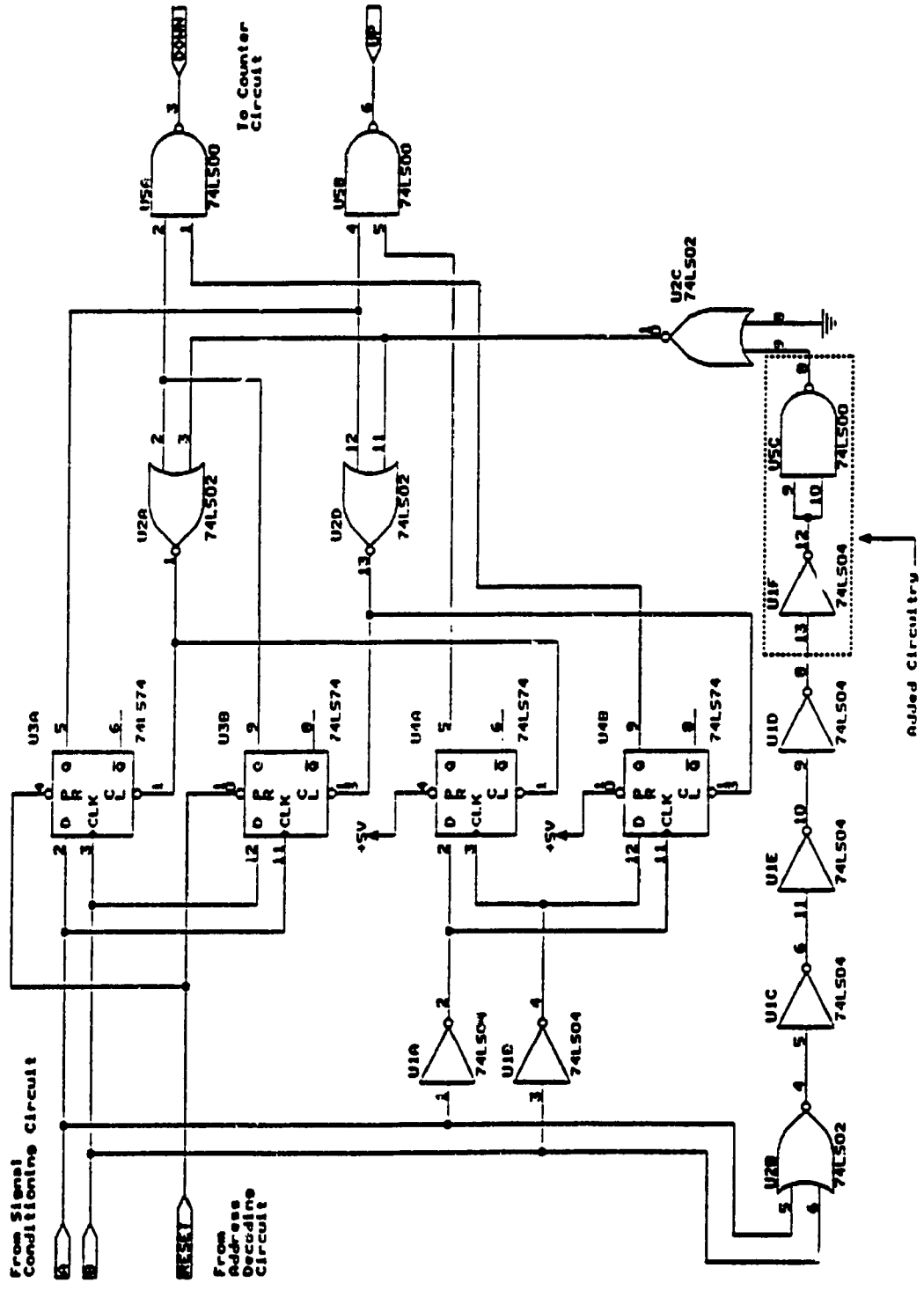


Fig.B.2 Direction Decoding Circuit

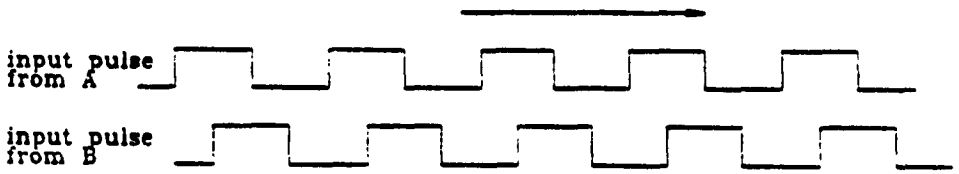


Fig.B.3a Pulse Train of A and B in one Direction

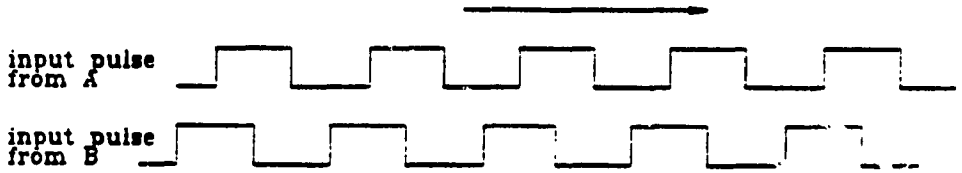


Fig.B.3b Pulse Train of A and B in Opposite Direction

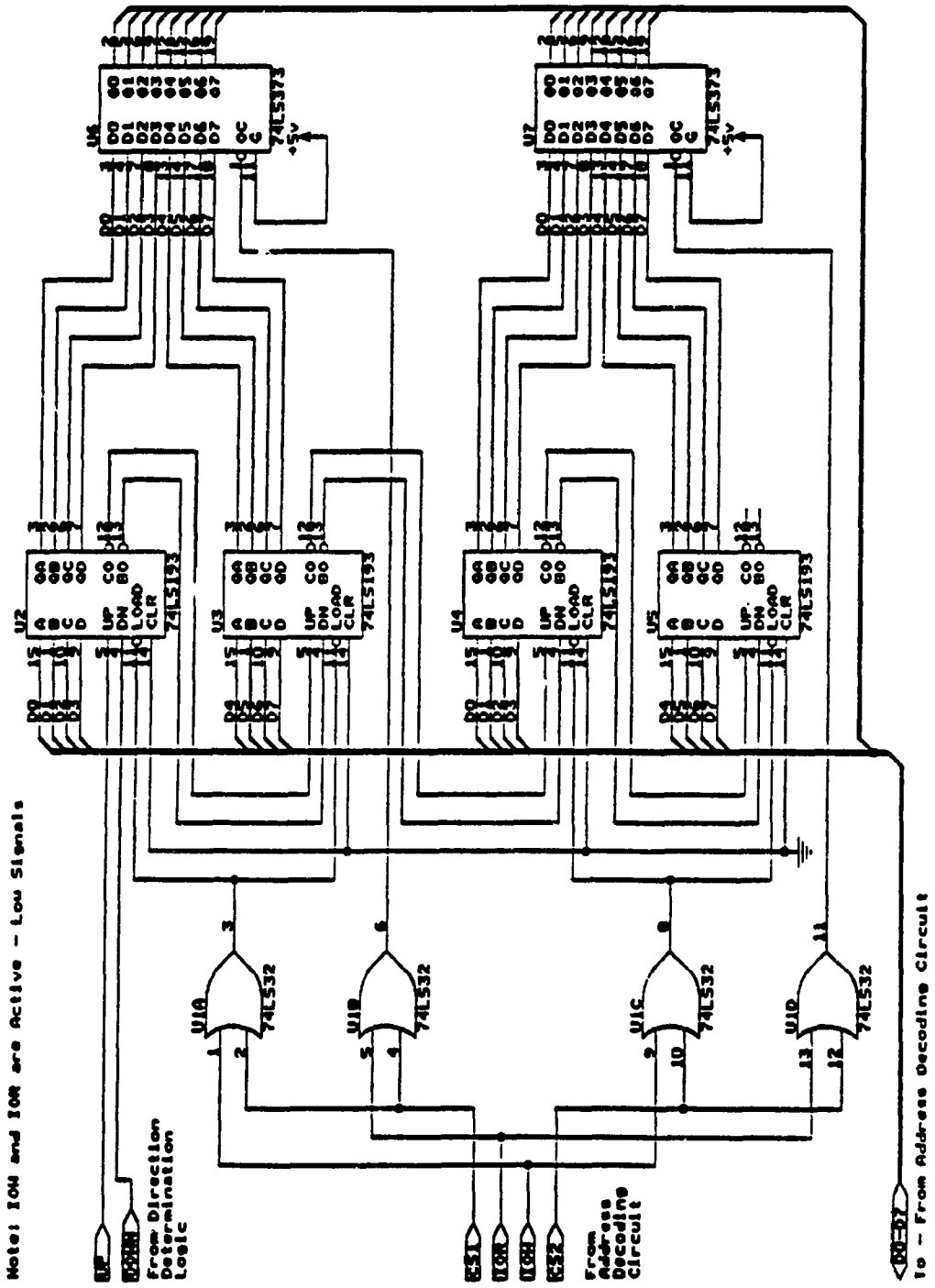


Fig.B.4 Counting Circuit

TABLE B.1 : JOINT PULSE TRAIN INDICATE POSITIVE MOTION

Joint #	Lead	Lag
1	A	B
2	B	A
3	A	B
4	B	A
5	A	B
6	B	A

TABLE B.2 RESOLUTION OF THE JOINT COUNTER FOR PUMA 560 ROBOT

Joint #	resolution(deg.)
1	0.0230
2	0.0167
3	0.0268
4	0.0189
5	0.0200
6	0.0188

B.5 Address Decoding Circuit

The joint counters are the IBM-PC parallel ports. Address decoding circuit (fig.B.5) is required for locating the bytes for each joint counters in the READ and WRITE operations. Each joint counter which has two bytes requires two I/O address spaces. Therefore, totally twelve I/O address are required for the data acquisition. Referring to the circuit diagram shown in fig.B.5, the I/O addresses are starting from Hex 380 to 38F. Only the twelve addresses from Hex 381 to 38C are used.

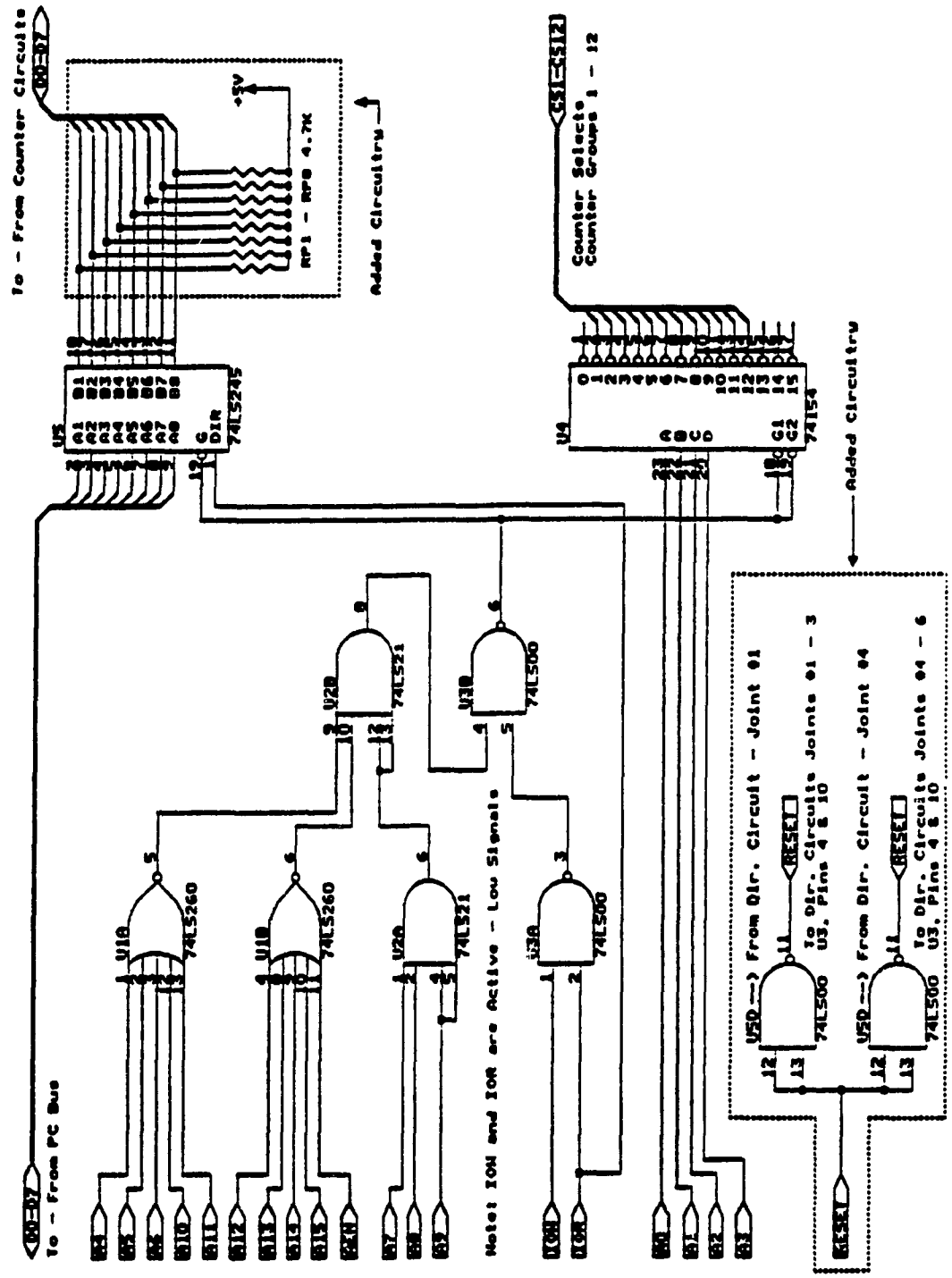


Fig.B.5 Address Decoding Circuit

APPENDIX C

Comparison of Common Multiprocessor Systems

Coprocessor Board	CPU	mips	High Level Lang.	Max. No. of Boards	On-Br. RAM	Host	Price per Board US Dollar
Transputer (20 MHz)	T800/ FPU	10	Yes	four/ Board	1-8 Mb	many *	4600 (2000/ addition processor)
TMS320C25 PC Coprocessor (40 MHz)	TMS320C25	10	No (Asm25)	15	256k	IBM PC	1500
Quantum (33 MHz)	Clipper	7.5	Yes	1	4Mb	IBM PC	5000
PC 4000 (4 MHz)	RISC 4000	5-8	forth	6	512k	IBM PC	1600
D64180LP (6.144 MHz)	HD64180 Z80 based	5.0	Yes	4	64k	IBM PC	600
PC-Turbo 286e (10 MHz)	80286/ 80287	4.0	Yes	2	1Mb	IBM PC	1300
PC68K2 (10 MHz)	68020/ 68881	3.5	Yes	1	1Mb	IBM PC	3300
PC-Elevator (8 MHz)	80386/	3.4	Yes	1	1Mb	IBM PC	2600
Harmony (16 MHz)	68020	3	Yes	16	4Mb	Sun	20000
PC68K1 (10 MHz)	68000/ 68881	1.5	Yes	1	1Mb	IBM PC	2000
DSI-780 (16.67 MHz)	68000/ 68881	1.3	Yes	4	1Mb	IBM PC	2200

* IBM-PC, VAX, Mac, Atari

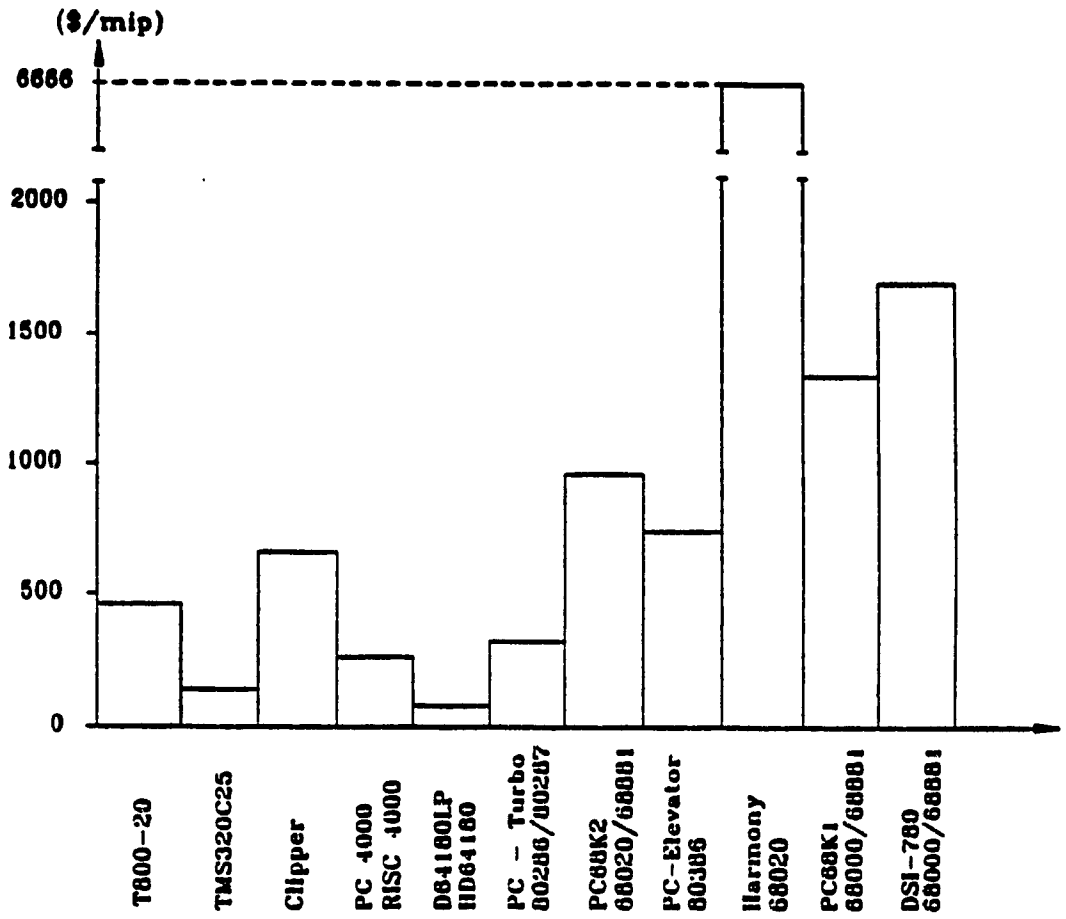


Fig.C.1 Dollar/Mip Comparison

APPENDIX D

Limiting Speed Ratios between Robot and Workpiece of MT Model

The interception scheme described in Section 4.3.1.1 not always produce a rational solution for m[k]. In order to do so, the discriminant of equation (6) has to be greater than zero:

$$\left\{ \frac{C_1[k]v_c[k]}{V_r^2 - v_c^2[k]} \right\}^2 + \frac{C_1^2[k] + C_2^2[k] + C_3^2[k]}{V_r^2 - v_c^2[k]} \geq 0$$

$$\{ C_1[k]v_c[k] \}^2 + \{ V_r^2 - v_c^2[k] \}\{ C_1^2[k] + C_2^2[k] + C_3^2[k] \} \geq 0 \qquad \ldots(D.1)$$

Normalising by the term $C_1[k]v_c[k]$, (D.1) becomes,

$$1 + \left\{ \frac{V_r^2}{v_c^2[k]} - 1 \right\} \left\{ 1 + \frac{C_2^2[k]}{C_1^2[k]} + \frac{C_3^2[k]}{C_1^2[k]} \right\} \geq 0 \qquad \ldots(D.2)$$

Let $r = \frac{V_r}{v_c[k]}$ and

$$f = \left\{ 1 + \frac{C_2^2[k]}{C_1^2[k]} + \frac{C_3^2[k]}{C_1^2[k]} \right\} \qquad \ldots(D.3)$$

The following conclusion may be drawn:

If $r \geq 1$, solution of m[k] always exists. If $r < 1$, then

$$g = 1 + (r^2 - 1) * f \geq 0$$

has to be satisfied in order to solve the real-time path equation. A plot of g and r is shown in fig.D.1. The closer the ratio of r to 1, the higher is the possibility for g to be greater than zero for different values of f.

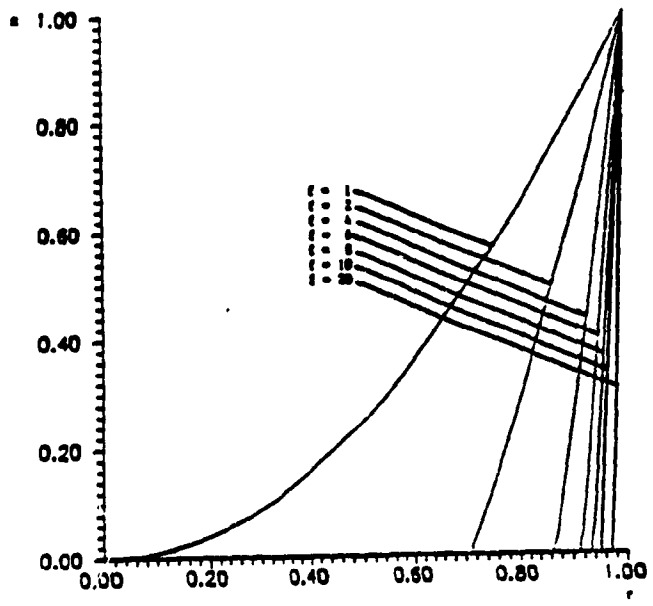


Fig.D.1 Conditions for Existence of Real-time Solution

APPENDIX E

Numerical Solution of the Elliptical Integral

Supposing that there is an ellipse shown in fig.E.1 with a major axis 'a' and a minor axis 'b'. The ellipse has the equations [12] as follows :

$$x = a \cos t \qquad y = b \sin t$$

The length of the arc AP can be expressed to be,

$$\text{arc } AP = \int_0^{\varphi} (a^2\sin^2 t + b^2\cos^2 t \, dt)^{1/2}$$

$$= a \int_0^{\varphi} (1 - \alpha^2\cos^2 t) \, dt \qquad \ldots(E.1)$$

where $\alpha = (a^2 - b^2)^{1/2} / a$ which is called the eccentricity

The lenght of the arc AP of the ellipse is represented by the elliptical integral as follows :

$$\text{arc } AP = a E (\alpha, \varphi) = a \int_0^{\varphi} (1 - \alpha^2\cos^2 t)dt \quad \ldots(E.2)$$

The elliptical integral function $E(\alpha, \varphi)$ cannot be solved by elementary functions. One of the methods is to evaluate the function through numerical method. Fortunately, Janke and Emde[43] have done the numerical computation. Fig.E.2 shows the behavior of the elliptical integral while α is a constant. Fig.E.3 shows the curves while φ is a constant. The computed numerical results are shown in the Table E.1.

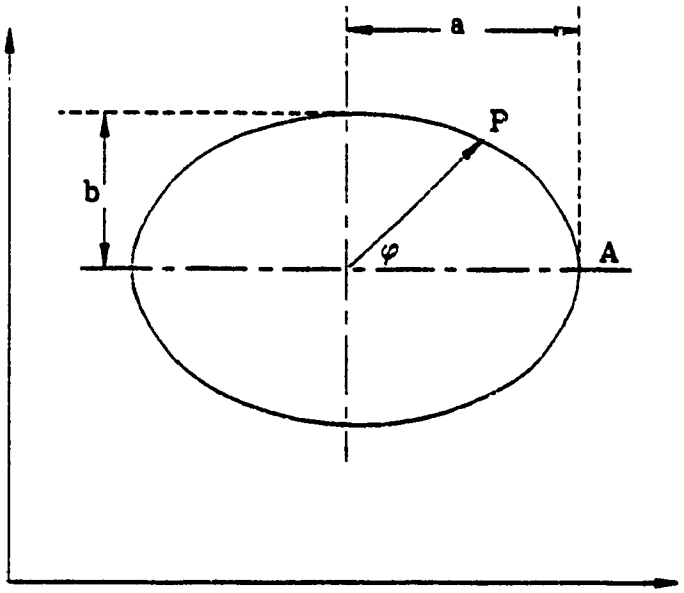


Fig.E.1 Ellipse with a major axis 'a' and minor axis 'b'

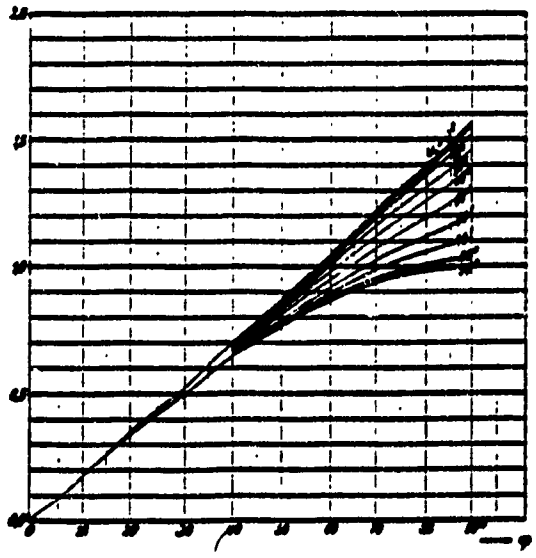


Fig.E.2 E(α, φ), α = constant

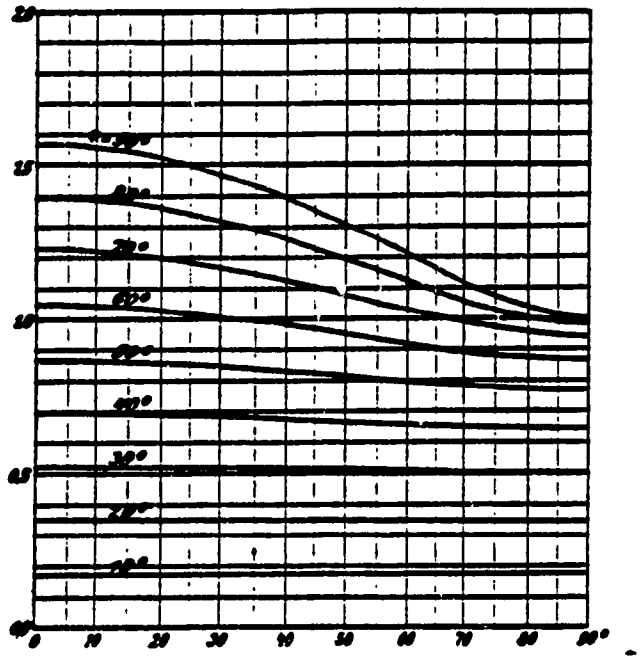


Fig.E.3 $E(\alpha, \varphi)$, φ = constant

Table E.1 Values of the Elliptical Integral

XV. $E(\alpha, \varphi)$

φ	α									
	0°	5°	10°	15°	20°	25°	30°	35°	40°	45°
1°	0.0175	0.0175	0.0175	0.0175	0.0175	0.0175	0.0175	0.0175	0.0175	0.0175
2°	0.0349	0.0349	0.0349	0.0349	0.0349	0.0349	0.0349	0.0349	0.0349	0.0349
3°	0.0524	0.0524	0.0524	0.0524	0.0524	0.0524	0.0524	0.0524	0.0524	0.0524
4°	0.0698	0.0698	0.0698	0.0698	0.0698	0.0698	0.0698	0.0698	0.0698	0.0698
5°	0.0873	0.0873	0.0873	0.0873	0.0873	0.0873	0.0872	0.0872	0.0872	0.0872
6°	0.1047	0.1047	0.1047	0.1047	0.1047	0.1047	0.1047	0.1047	0.1046	0.1046
7°	0.1222	0.1222	0.1222	0.1222	0.1221	0.1221	0.1221	0.1221	0.1221	0.1220
8°	0.1396	0.1396	0.1396	0.1396	0.1396	0.1396	0.1395	0.1395	0.1394	0.1394
9°	0.1571	0.1571	0.1571	0.1570	0.1570	0.1570	0.1569	0.1569	0.1568	0.1568
10°	0.1745	0.1745	0.1745	0.1745	0.1744	0.1744	0.1743	0.1743	0.1742	0.1741
11°	0.1920	0.1920	0.1920	0.1919	0.1919	0.1918	0.1917	0.1916	0.1915	0.1914
12°	0.2094	0.2094	0.2094	0.2093	0.2093	0.2092	0.2091	0.2090	0.2088	0.2087
13°	0.2269	0.2269	0.2268	0.2268	0.2267	0.2266	0.2264	0.2263	0.2261	0.2259
14°	0.2444	0.2443	0.2443	0.2442	0.2441	0.2439	0.2437	0.2436	0.2434	0.2431
15°	0.2618	0.2618	0.2617	0.2616	0.2615	0.2613	0.2611	0.2608	0.2606	0.2603
16°	0.2793	0.2792	0.2791	0.2789	0.2788	0.2786	0.2784	0.2781	0.2778	0.2775
17°	0.2967	0.2967	0.2966	0.2964	0.2962	0.2959	0.2956	0.2953	0.2949	0.2946
18°	0.3142	0.3141	0.3140	0.3138	0.3136	0.3133	0.3129	0.3125	0.3121	0.3116
19°	0.3316	0.3316	0.3314	0.3312	0.3309	0.3306	0.3301	0.3297	0.3291	0.3286
20°	0.3491	0.3490	0.3489	0.3486	0.3483	0.3478	0.3473	0.3468	0.3462	0.3456
21°	0.3665	0.3665	0.3663	0.3660	0.3656	0.3651	0.3645	0.3639	0.3632	0.3625
22°	0.3840	0.3839	0.3837	0.3834	0.3829	0.3823	0.3817	0.3809	0.3802	0.3793
23°	0.4014	0.4014	0.4011	0.4007	0.4002	0.3996	0.3988	0.3980	0.3971	0.3961
24°	0.4189	0.4188	0.4185	0.4181	0.4175	0.4168	0.4159	0.4150	0.4139	0.4129
25°	0.4363	0.4362	0.4359	0.4354	0.4348	0.4339	0.4330	0.4319	0.4308	0.4296
26°	0.4538	0.4537	0.4533	0.4528	0.4520	0.4511	0.4500	0.4488	0.4475	0.4462
27°	0.4712	0.4711	0.4707	0.4701	0.4693	0.4682	0.4670	0.4657	0.4643	0.4628
28°	0.4887	0.4886	0.4881	0.4875	0.4865	0.4854	0.4840	0.4825	0.4809	0.4793
29°	0.5062	0.5060	0.5055	0.5048	0.5037	0.5025	0.5010	0.4993	0.4975	0.4957
30°	0.5236	0.5234	0.5229	0.5221	0.5209	0.5195	0.5179	0.5161	0.5141	0.5121
31°	0.5411	0.5409	0.5403	0.5394	0.5381	0.5368	0.5348	0.5329	0.5306	0.5283
32°	0.5585	0.5583	0.5577	0.5567	0.5553	0.5538	0.5516	0.5494	0.5470	0.5446
33°	0.5760	0.5757	0.5751	0.5740	0.5725	0.5706	0.5684	0.5660	0.5634	0.5607
34°	0.5934	0.5932	0.5924	0.5912	0.5896	0.5876	0.5852	0.5826	0.5797	0.5768
35°	0.6109	0.6106	0.6098	0.6085	0.6067	0.6045	0.6019	0.5991	0.5960	0.5928
36°	0.6283	0.6280	0.6272	0.6258	0.6238	0.6214	0.6186	0.6156	0.6122	0.6087
37°	0.6458	0.6456	0.6445	0.6430	0.6409	0.6383	0.6353	0.6319	0.6283	0.6245
38°	0.6632	0.6629	0.6619	0.6602	0.6580	0.6552	0.6519	0.6483	0.6444	0.6403
39°	0.6807	0.6803	0.6792	0.6775	0.6750	0.6720	0.6685	0.6646	0.6604	0.6559
40°	0.6981	0.6977	0.6966	0.6947	0.6921	0.6888	0.6851	0.6808	0.6763	0.6715
41°	0.7156	0.7150	0.7139	0.7119	0.7091	0.7056	0.7016	0.6970	0.6921	0.6870
42°	0.7330	0.7325	0.7313	0.7291	0.7261	0.7224	0.7180	0.7134	0.7079	0.7025
43°	0.7505	0.7500	0.7486	0.7463	0.7431	0.7391	0.7346	0.7293	0.7237	0.7178
44°	0.7679	0.7674	0.7659	0.7634	0.7600	0.7558	0.7509	0.7453	0.7393	0.7330
45°	0.7854	0.7849	0.7832	0.7806	0.7770	0.7725	0.7672	0.7613	0.7549	0.7482

XV. $E(\alpha, \varphi)$

φ	α									
	0°	5°	10°	15°	20°	25°	30°	35°	40°	45°
46°	0.8029	0.8023	0.8006	0.7978	0.7939	0.7891	0.7835	0.7772	0.7704	0.7633
47°	0.8203	0.8197	0.8179	0.8149	0.8108	0.8057	0.7998	0.7931	0.7858	0.7782
48°	0.8378	0.8371	0.8352	0.8320	0.8277	0.8223	0.8160	0.8089	0.8012	0.7931
49°	0.8552	0.8545	0.8525	0.8491	0.8446	0.8389	0.8322	0.8247	0.8165	0.8079
50°	0.8727	0.8719	0.8698	0.8663	0.8614	0.8554	0.8483	0.8404	0.8317	0.8227
51°	0.8901	0.8894	0.8871	0.8834	0.8783	0.8719	0.8644	0.8560	0.8469	0.8373
52°	0.9076	0.9068	0.9044	0.9005	0.8951	0.8884	0.8805	0.8716	0.8622	0.8518
53°	0.9250	0.9242	0.9217	0.9175	0.9119	0.9048	0.8965	0.8872	0.8770	0.8663
54°	0.9425	0.9416	0.9390	0.9345	0.9287	0.9212	0.9125	0.9026	0.8919	0.8806
55	0.9599	0.9590	0.9562	0.9517	0.9454	0.9376	0.9284	0.9181	0.9068	0.8949
56°	0.9774	0.9764	0.9735	0.9687	0.9622	0.9540	0.9443	0.9335	0.9216	0.9091
57°	0.9948	0.9938	0.9908	0.9858	0.9789	0.9703	0.9602	0.9488	0.9363	0.9232
58°	1.0123	1.0112	1.0080	1.0028	0.9956	0.9866	0.9760	0.9641	0.9510	0.9372
59°	1.0297	1.0286	1.0253	1.0198	1.0123	1.0029	0.9918	0.9793	0.9656	0.9511
60°	1.0472	1.0460	1.0426	1.0368	1.0290	1.0192	1.0075	0.9945	0.9801	0.9650
61°	1.0647	1.0634	1.0598	1.0538	1.0456	1.0354	1.0233	1.0097	0.9946	0.9787
62°	1.0821	1.0808	1.0771	1.0708	1.0623	1.0516	1.0390	1.0249	1.0090	0.9924
63°	1.0996	1.0982	1.0943	1.0878	1.0789	1.0678	1.0546	1.0401	1.0235	1.0060
64°	1.1170	1.1156	1.1115	1.1048	1.0955	1.0839	1.0702	1.0553	1.0381	1.0195
65°	1.1345	1.1330	1.1288	1.1218	1.1121	1.1001	1.0859	1.0705	1.0526	1.0329
66°	1.1519	1.1504	1.1460	1.1387	1.1287	1.1162	1.1013	1.0845	1.0671	1.0463
67°	1.1694	1.1678	1.1632	1.1557	1.1453	1.1323	1.1168	1.0993	1.0811	1.0596
68°	1.1868	1.1852	1.1805	1.1726	1.1619	1.1483	1.1323	1.1141	1.0954	1.0728
69°	1.2043	1.2026	1.1977	1.1895	1.1784	1.1644	1.1478	1.1289	1.1097	1.0859
70°	1.2217	1.2200	1.2149	1.2065	1.1949	1.1804	1.1632	1.1436	1.1221	1.0990
71°	1.2392	1.2374	1.2321	1.2234	1.2115	1.1964	1.1786	1.1583	1.1360	1.1120
72°	1.2566	1.2548	1.2494	1.2403	1.2280	1.2124	1.1938	1.1729	1.1498	1.1250
73°	1.2741	1.2722	1.2666	1.2573	1.2445	1.2284	1.2093	1.1876	1.1635	1.1379
74°	1.2915	1.2896	1.2838	1.2742	1.2609	1.2443	1.2245	1.2021	1.1773	1.1507
75°	1.3089	1.3070	1.3010	1.2911	1.2774	1.2603	1.2398	1.2167	1.1910	1.1635
76°	1.3265	1.3244	1.3182	1.3080	1.2939	1.2762	1.2552	1.2312	1.2047	1.1762
77°	1.3439	1.3418	1.3354	1.3249	1.3104	1.2921	1.2704	1.2457	1.2183	1.1888
78°	1.3614	1.3592	1.3526	1.3417	1.3268	1.3080	1.2857	1.2601	1.2319	1.2015
79°	1.3788	1.3765	1.3698	1.3586	1.3433	1.3239	1.3009	1.2746	1.2454	1.2141
80°	1.3963	1.3939	1.3870	1.3755	1.3597	1.3398	1.3161	1.2890	1.2589	1.2266
81°	1.4137	1.4113	1.4042	1.3924	1.3761	1.3556	1.3312	1.3034	1.2725	1.2391
82°	1.4312	1.4287	1.4214	1.4093	1.3925	1.3715	1.3464	1.3177	1.2860	1.2516
83°	1.4486	1.4461	1.4386	1.4261	1.4089	1.3873	1.3616	1.3321	1.2994	1.2640
84°	1.4661	1.4635	1.4558	1.4430	1.4254	1.4032	1.3767	1.3464	1.3128	1.2765
85°	1.4835	1.4809	1.4729	1.4598	1.4418	1.4190	1.3919	1.3608	1.3262	1.2889
86°	1.5010	1.4983	1.4901	1.4767	1.4582	1.4348	1.4070	1.3751	1.3396	1.3012
87	1.5184	1.5157	1.5073	1.4936	1.4746	1.4507	1.4221	1.3894	1.3530	1.3136
88°	1.5359	1.5330	1.5245	1.5104	1.4910	1.4665	1.4372	1.4037	1.3664	1.3260
89	1.5533	1.5504	1.5417	1.5273	1.5074	1.4823	1.4524	1.4180	1.3798	1.3383
90	1.5708	1.5678	1.5589	1.5442	1.5238	1.4981	1.4675	1.4323	1.3931	1.3506

Table E.1 Values of the Elliptical Integral

φ	α								
	50°	55°	60°	65°	70°	75°	80°	85°	90°
1°	0.0175	0.0175	0.0175	0.0175	0.0175	0.0175	0.0175	0.0175	0.0175
2°	0.0349	0.0349	0.0349	0.0349	0.0349	0.0349	0.0349	0.0349	0.0349
3°	0.0524	0.0523	0.0523	0.0523	0.0523	0.0523	0.0523	0.0523	0.0523
4°	0.0698	0.0698	0.0698	0.0698	0.0698	0.0698	0.0698	0.0698	0.0698
5°	0.0872	0.0872	0.0872	0.0872	0.0872	0.0872	0.0872	0.0872	0.0872
6°	0.1046	0.1046	0.1046	0.1046	0.1046	0.1045	0.1045	0.1045	0.1046
7°	0.1220	0.1220	0.1220	0.1219	0.1219	0.1219	0.1219	0.1219	0.1219
8°	0.1394	0.1393	0.1393	0.1393	0.1393	0.1392	0.1392	0.1392	0.1392
9°	0.1567	0.1567	0.1566	0.1566	0.1565	0.1565	0.1565	0.1564	0.1564
10°	0.1740	0.1739	0.1739	0.1739	0.1738	0.1737	0.1737	0.1737	0.1737
11°	0.1913	0.1912	0.1911	0.1910	0.1910	0.1909	0.1909	0.1909	0.1909
12°	0.2086	0.2084	0.2083	0.2082	0.2081	0.2080	0.2080	0.2079	0.2079
13°	0.2258	0.2256	0.2254	0.2253	0.2252	0.2251	0.2250	0.2250	0.2250
14°	0.2429	0.2427	0.2425	0.2424	0.2422	0.2421	0.2420	0.2419	0.2419
15°	0.2601	0.2598	0.2596	0.2594	0.2592	0.2590	0.2589	0.2588	0.2588
16°	0.2771	0.2769	0.2766	0.2763	0.2761	0.2759	0.2758	0.2757	0.2756
17°	0.2942	0.2938	0.2935	0.2932	0.2929	0.2927	0.2925	0.2924	0.2924
18°	0.3112	0.3107	0.3103	0.3100	0.3096	0.3094	0.3092	0.3091	0.3090
19°	0.3281	0.3276	0.3271	0.3267	0.3263	0.3260	0.3258	0.3256	0.3256
20°	0.3450	0.3444	0.3438	0.3433	0.3429	0.3425	0.3422	0.3421	0.3420
21°	0.3619	0.3611	0.3604	0.3599	0.3593	0.3589	0.3586	0.3584	0.3584
22°	0.3785	0.3777	0.3770	0.3763	0.3757	0.3753	0.3749	0.3747	0.3746
23°	0.3952	0.3943	0.3935	0.3927	0.3920	0.3915	0.3911	0.3908	0.3907
24°	0.4118	0.4108	0.4098	0.4090	0.4082	0.4076	0.4071	0.4068	0.4067
25°	0.4284	0.4272	0.4261	0.4251	0.4243	0.4236	0.4230	0.4227	0.4226
26°	0.4449	0.4436	0.4423	0.4412	0.4402	0.4394	0.4388	0.4385	0.4384
27°	0.4613	0.4598	0.4584	0.4572	0.4561	0.4552	0.4545	0.4541	0.4540
28°	0.4776	0.4760	0.4744	0.4730	0.4718	0.4708	0.4701	0.4695	0.4695
29°	0.4939	0.4920	0.4903	0.4888	0.4874	0.4863	0.4855	0.4850	0.4848
30°	0.5100	0.5080	0.5061	0.5044	0.5029	0.5017	0.5007	0.5002	0.5000
31°	0.5261	0.5239	0.5218	0.5199	0.5182	0.5169	0.5159	0.5153	0.5150
32°	0.5421	0.5396	0.5373	0.5352	0.5334	0.5319	0.5308	0.5302	0.5299
33°	0.5580	0.5553	0.5528	0.5505	0.5485	0.5468	0.5456	0.5449	0.5446
34°	0.5738	0.5709	0.5681	0.5656	0.5634	0.5616	0.5603	0.5595	0.5592
35°	0.5895	0.5863	0.5833	0.5806	0.5782	0.5762	0.5748	0.5739	0.5736
36°	0.6050	0.6017	0.5984	0.5954	0.5928	0.5907	0.5891	0.5881	0.5878
37°	0.6207	0.6169	0.6134	0.6101	0.6073	0.6050	0.6032	0.6022	0.6018
38°	0.6361	0.6321	0.6282	0.6247	0.6216	0.6191	0.6172	0.6161	0.6157
39°	0.6515	0.6471	0.6429	0.6391	0.6357	0.6330	0.6310	0.6297	0.6293
40°	0.6667	0.6620	0.6575	0.6533	0.6497	0.6468	0.6446	0.6432	0.6428
41°	0.6819	0.6768	0.6719	0.6675	0.6636	0.6604	0.6580	0.6566	0.6561
42°	0.6969	0.6914	0.6862	0.6814	0.6772	0.6738	0.6712	0.6697	0.6691
43°	0.7118	0.7059	0.7003	0.6952	0.6907	0.6870	0.6843	0.6826	0.6820
44°	0.7267	0.7204	0.7144	0.7088	0.7040	0.7001	0.6971	0.6953	0.6947
45°	0.7414	0.7347	0.7282	0.7223	0.7172	0.7129	0.7097	0.7078	0.7071

φ	α								
	50°	55°	60°	65°	70°	75°	80°	85°	90°
46°	0.7560	0.7488	0.7420	0.7358	0.7301	0.7255	0.7222	0.7201	0.7193
47°	0.7705	0.7629	0.7557	0.7492	0.7429	0.7380	0.7344	0.7321	0.7314
48°	0.7849	0.7768	0.7690	0.7618	0.7555	0.7503	0.7464	0.7440	0.7431
49°	0.7992	0.7905	0.7823	0.7745	0.7679	0.7623	0.7582	0.7556	0.7547
50°	0.8134	0.8042	0.7954	0.7872	0.7801	0.7741	0.7697	0.7670	0.7660
51°	0.8275	0.8177	0.8084	0.7997	0.7921	0.7858	0.7811	0.7781	0.7772
52°	0.8414	0.8311	0.8212	0.8120	0.8038	0.7972	0.7922	0.7891	0.7880
53°	0.8553	0.8444	0.8339	0.8242	0.8155	0.8084	0.8031	0.7998	0.7986
54°	0.8690	0.8576	0.8464	0.8361	0.8270	0.8194	0.8137	0.8102	0.8090
55°	0.8827	0.8705	0.8588	0.8479	0.8382	0.8302	0.8242	0.8204	0.8192
56°	0.8962	0.8834	0.8710	0.8595	0.8493	0.8408	0.8344	0.8304	0.8290
57°	0.9097	0.8961	0.8831	0.8709	0.8601	0.8511	0.8443	0.8401	0.8387
58°	0.9230	0.9086	0.8950	0.8822	0.8708	0.8612	0.8540	0.8496	0.8481
59°	0.9362	0.9213	0.9068	0.8933	0.8812	0.8711	0.8635	0.8589	0.8572
60°	0.9493	0.9336	0.9184	0.9042	0.8914	0.8808	0.8729	0.8677	0.8660
61°	0.9623	0.9459	0.9298	0.9149	0.9015	0.8903	0.8818	0.8764	0.8746
62°	0.9752	0.9580	0.9412	0.9254	0.9113	0.8995	0.8905	0.8849	0.8830
63°	0.9880	0.9700	0.9524	0.9358	0.9210	0.9085	0.8990	0.8930	0.8910
64°	1.0007	0.9818	0.9634	0.9460	0.9304	0.9173	0.9072	0.9008	0.8988
65°	1.0133	0.9936	0.9743	0.9561	0.9397	0.9258	0.9152	0.9085	0.9063
66°	1.0259	1.0052	0.9850	0.9660	0.9487	0.9341	0.9230	0.9160	0.9136
67°	1.0383	1.0167	0.9956	0.9758	0.9576	0.9422	0.9305	0.9231	0.9205
68°	1.0506	1.0282	1.0061	0.9852	0.9662	0.9501	0.9377	0.9298	0.9272
69°	1.0628	1.0395	1.0164	0.9946	0.9747	0.9578	0.9447	0.9364	0.9336
70°	1.0750	1.0508	1.0266	1.0039	0.9830	0.9652	0.9514	0.9427	0.9397
71°	1.0871	1.0617	1.0367	1.0129	0.9911	0.9724	0.9579	0.9487	0.9455
72°	1.0991	1.0727	1.0467	1.0218	0.9990	0.9794	0.9642	0.9544	0.9511
73°	1.1110	1.0836	1.0565	1.0306	1.0067	0.9862	0.9702	0.9599	0.9563
74°	1.1228	1.0944	1.0662	1.0392	1.0143	0.9928	0.9759	0.9650	0.9613
75°	1.1346	1.1051	1.0759	1.0477	1.0217	0.9992	0.9814	0.9699	0.9658
76°	1.1463	1.1158	1.0854	1.0561	1.0289	1.0053	0.9867	0.9745	0.9703
77°	1.1580	1.1263	1.0948	1.0643	1.0361	1.0113	0.9917	0.9789	0.9744
78°	1.1696	1.1368	1.1041	1.0725	1.0430	1.0171	0.9965	0.9829	0.9782
79°	1.1811	1.1472	1.1133	1.0805	1.0498	1.0228	1.0011	0.9867	0.9816
80°	1.1926	1.1576	1.1225	1.0884	1.0565	1.0282	1.0056	0.9902	0.9848
81°	1.2040	1.1678	1.1316	1.0962	1.0630	1.0335	1.0096	0.9935	0.9877
82°	1.2154	1.1781	1.1408	1.1040	1.0695	1.0387	1.0135	0.9965	0.9903
83°	1.2267	1.1883	1.1495	1.1116	1.0758	1.0437	1.0173	0.9992	0.9926
84°	1.2381	1.1984	1.1584	1.1192	1.0821	1.0486	1.0209	1.0017	0.9945
85°	1.2493	1.2086	1.1673	1.1267	1.0883	1.0534	1.0244	1.0039	0.9962
86°	1.2606	1.2186	1.1761	1.1342	1.0944	1.0581	1.0277	1.0060	0.9976
87°	1.2719	1.2286	1.1848	1.1417	1.1004	1.0628	1.0309	1.0078	0.9986
88°	1.2831	1.2387	1.1936	1.1491	1.1064	1.0674	1.0340	1.0095	0.9994
89°	1.2943	1.2487	1.2023	1.1565	1.1124	1.0719	1.0371	1.0111	0.9999
90°	1.3055	1.2587	1.2111	1.1639	1.1184	1.0764	1.0401	1.0127	1.0000

APPENDIX F

Numerical Differentiation to Evaluate the Joint Velocities and Accelerations

Supposing the joint position, velocity and acceleration at the kth sampling time are represented by p[k], v[k] and a[k]. The velocity and acceleration at the kth sampling time can be differentiated numerically by a 5-point formula[14] which provides a smoother derivative than the 2-point or 3-point formula. Given p[k-2], p[k-1], p[k+1], p[k+2], the velocity at kth sampling time can be calculated as follows :

$$v[k] = \frac{p[k-2] - 8 * p[k-1] + 8 * p[k+1] - p[k+2]}{12\ \tau_s} \qquad ...(F.1)$$

Similiarly, the acceleration at the kth sampling time can be computed as follows,

$$a[k] = \frac{v[k-2] - 8 * v[k-1] + 8 * v[k+1] - v[k+2]}{12\ \tau_s}, \qquad ...(F.2)$$

If k is less than 3, the following 2-point formula is used instead.

$$v[k] = \frac{p[k] - p[k-1]}{\tau_s} \qquad ...(F.3)$$

$$a[k] = \frac{v[k] - v[k-1]}{\tau_s} \qquad ...(F.4)$$

APPENDIX G

Newton-Euler Inverse Dynamics

The objective of the inverse dynamics is to compute the required torques for each joints for a set of given joint positions, velocities and accelerations. There are two common inverse dynamics models - Lagrangian method and Newton-Euler method. The simpler approach, Newton-Euler is used to compute the torques in this thesis. For detailed analysis on these dynamical equations, the reader is advised to refer reference[11]. Detailed formulation is beyond the scope of this thesis. Only the final form of the derivation is shown.

A is the transformation (3x3) matrix

f_i is the force (3x1) vector exerted on link i by link i-1

F_i is the (3x1) force vector acting on the centre of mass of link i

I_c is the (3x3) mass matrix

m_i is the total of link i

N_i is the (3x1) moment vector acting on link i

n_i is the (3x1) torque vector exerted on link i by link i-1

n is the number of link

τ_i is the required torque of link *i*

P_i is the (3x1) position vector of the origin of coordinates which is attached to link i

r_i is the (3x1) position vector from the position P_i tothe centre of mass link i

v is the linear velocity

v_c is the velocity of the centre of mass

ω be the joint angular velocity

Outward Iterations for i=0 to n-1

$$\omega_{i+1} = A_{i+1}^{t}[\ \omega_i + z_i \dot{\theta}_{i+1}] \qquad \ldots(G.1)$$

$$\dot{\omega}_{i+1} = A_{i+1}^{t}[\ \dot{\omega}_i + z_i \dot{\theta}_{i+1} + \omega_i \times z_i \dot{\theta}_{i+1}] \qquad \ldots(G.2)$$

$$\dot{v}_{i+1} = \dot{\omega}_{i+1} \times P_{i+1} + \omega_{i+1} \times (\ \omega_{i+1} \times P_{i+1}) + A_{i+1}^{t} \dot{v}_i \qquad \ldots(G.3)$$

$$\dot{v}_{c_{i+1}} = \dot{\omega}_{i+1} \times r_{i+1} + \omega_{i+1} \times (\omega_{i+1} \times r_{i+1}) + \dot{v}_{i+1} \qquad \ldots(G.4)$$

$$F_{i+1} = m_{i+1} \dot{v}_{c_{i+1}} \qquad \ldots(G.5)$$

$$N_{i+1} = I_{c_i} \dot{\omega}_{i+1} + \omega_{i+1} \times (\ I_{c_i} \omega_{i+1}) \qquad \ldots(G.6)$$

Inward Iterations for i = number of joints to 1

$$f_i = A_{i+1} f_{i+1} + F_i \qquad \ldots(G.7)$$

$$n_i = A_{i+1} n_{i+1} + P_i \times (\ A_{i+1} f_{i+1}) + (\ P_i + r_i) \times F_i \qquad \ldots(G.8)$$

$$\tau_i = n_i^{t} A_i^{t} z_{i-1} \qquad \ldots(G.9)$$

APPENDIX H

PUMA 560 ROBOT FORWARD KINEMATICS MODEL

The objective of the forward kinematic model is to transform the joint positions into the cartesian world coordinates. This model is assigned to one transputer as the overall position feedback in order to compensate the syncronisation error between the moving workpiece and the robot end-effector. The formulation[28] is as follows,

Let A_i be the transformation matrix of link i

n, o, a, be the normal vector, orientation vector and the approach vector

where $n = n_x i + n_y j + n_z k = [\, n_x, n_y, n_z \,]^t$

$o = o_x i + o_y j + o_z k = [\, o_x, o_y, o_z \,]^t$

$a = a_x i + a_y j + a_z k = [\, a_x, a_y, a_z \,]^t$

p is the position vector of the end-effector

where $p = p_x i + p_y j + p_z k = [\, p_x, p_y, p_z \,]^t$

θ_i be the angle between X_{i-1} and X_i measured about Z_i

T be the tool transformation

a_i be the distance from Z_i to Z_{i+1} measured along X_i

d_i be the distance from X_{i-1} to X_i measured along Z_i

The form of the kinematic equation is :

$$A_1A_2A_3A_4A_5A_6T = [\, n,\ o,\ a,\ p \,] = \begin{bmatrix} n_x & o_x & a_x & p_x \\ n_y & o_y & a_y & p_y \\ n_z & o_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$A_i = \begin{bmatrix} \cos\theta_i & -\sin\theta_i\cos\alpha_i & \sin\theta_i\sin\alpha_i & a\cos\theta_i \\ \sin\theta_i & \cos\theta_i\cos\alpha_i & -\cos\theta_i\sin\alpha_i & a\sin\theta_i \\ 0 & \sin\alpha_i & \cos\alpha_i & d \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

For the PUMA 560, the joints #4, #5 and #6 are attached to a gear- mechanism. Therefore, the mechanical coupling causes joints #5 and #6 to move when joint #4 moves. The joints #5 and #6 can be compensated as follows[89] :

$$\Delta\theta_4 = \theta_4 - \theta_{4o}$$

$$\Delta\theta_5 = \theta_5 - \theta_{5o}$$

$$\theta_{5c} = \theta_5 - \Delta\theta_4 * c_{45}$$

$$\theta_{6c} = \theta_6 - (\Delta\theta_4 * c_{46} + \Delta\theta_5 * c_{56})$$

where θ_{4o} is the starting position of joint #4

θ_{5o} is the starting position of joint #5

θ_{5c} is the given joint position after compensation

θ_{6c} is the given joint position after compensation

c_{45} is the correction factor for coupling between joint #4 and joint #5

c_{46} is the correction factor for coupling between joint #4 and joint #6

c_{56} is the correction factor for coupling between joint #5 and joint #6

For the Puma 560 robot, the values of the correction factors are as follows,

$$c_{45} = 0.0139037$$

$$c_{46} = -0.0130401$$

$$c_{56} = 0.1805560$$

The A_i matrices of the Puma 560 robot are as the followings :

$$A_1 = \begin{bmatrix} \cos\theta_1 & 0 & -\sin\theta_1 & 0 \\ \sin\theta_1 & 0 & \cos\theta_1 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$A_2 = \begin{bmatrix} \cos\theta_2 & \sin\theta_2 & 0 & a_1\cos\theta_2 \\ \sin\theta_2 & -\cos\theta_2 & 0 & a_1\sin\theta_2 \\ 0 & 0 & -1 & d_1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$A_3 = \begin{bmatrix} \cos\theta_3 & 0 & \sin\theta_3 & a_2\cos\theta_3 \\ \sin\theta_3 & 0 & -\cos\theta_3 & a_2\sin\theta_3 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$A_4 = \begin{bmatrix} \cos\theta_4 & 0 & -\sin\theta_4 & 0 \\ \sin\theta_4 & 0 & \cos\theta_4 & 0 \\ 0 & -1 & 0 & d_2 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$A_5 = \begin{bmatrix} \cos\theta_{5c} & 0 & \sin\theta_{5c} & 0 \\ \sin\theta_{5c} & 0 & -\cos\theta_{5c} & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$A_6 = \begin{bmatrix} \cos\theta_{6c} & -\sin\theta_{6c} & 0 & 0 \\ \sin\theta_{6c} & \cos\theta_{6c} & 0 & 0 \\ 0 & 0 & 1 & d_3 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

where $a_1 = 431.80$ mm

$a_2 = 20.32$ mm

$d_1 = 149.09$ mm

$d_2 = 433.07$ mm

$d_3 = 56.25$ mm

APPENDIX I

INCREMENTAL IMAGE PROCESSING

The following symbols are extensively used in this appendix :

SYMBOLS

P_r	position vector of the end-effector to centre the auxiliary camera on the calibrating position M
r_e	resultant compensation vector without drift
r'_e	resultant compensation vector with drift
$r_{i\Theta}$	position vector of the critical point i in the workpiece coordinates
$r'_{i\Theta}$	drifted position vector of the critical point i in workpiece coordinate
$r_{ij\Theta}$	vector pointing from critical point i to critical point j in the workpiece coordinate (e.g. $x_{ab\Theta}$, $y_{ab\Theta}$)
r_p	vector offset between the workpiece position and the auxiliary camera calibrating position
r_r	position vector of the workpiece in reference coordinate (components x_r, y_r)
Δr_r	drift vector of the workpiece position in reference coordinate (components Δx_r, Δy_r)
Δr_Θ	drift vector of the workpiece position in workpiece coordinate (components Δx_Θ, Δy_Θ)
r_w	vector offset between the camera calibrating point and the robot end-effector
α	angle between the X_Θ axis and the vector r_p
θ	computed workpiece orientation by the primary image processing
$\Delta\Theta$	orientation drift

I.1 INTRODUCTION

The accuracy of the primary image processing is affected by the following factors :

1. low resolution of the camera
2. resolution of the parameter tables
3. image blurring due to the relative velocity between the moving workpiece and the stationary camera
4. image blurring due to the lighting condition
5. relative drift between the conveyor and the workpiece due to random vibrations

- low resolution of the camera

The resolution of the camera is 128pixels x 64 pixels which corresponds a window of 265mm x 46mm on the conveyor belt. The longitidual(x) resolution is 0.719 mm/pixel and the latitude(y) resolution is 2.07 mm/pixel. Therefore, any movement in x less than 0.719mm and y less than 2.07mm may not be detected.

- resolution of the parameter table

It has been discussed in the section 2.1.2.3 that the high resolution of the parameter may not be able to increase the accuracy of the system. It even generates a time loss and therefore the production time is reduced within a fixed working envelop. The formulation and the concept is explained with details in the Appendix J.

- image blurring due to the relative velocity between the moving workpiece and the primary camera.

Image blurring may be due to the movement of the workpiece while it is under exposure. For example, if the workpiece moves at a speed of 150 mm/sec. The workpiece has already been moved 150 mm/sec x 0.008sec = 1.2 mm at the duration of the exposure time 8 msec.

- image blurring due to the lighting condition

Two flood lights are used as the lighting of the primary camera station. The intensity of the lighting is calibrated at the centre of the conveyor belt. Workpieces which are distant from the central position may cause shadows and thus the image may be distorted.

- relative drift between the conveyor and the workpiece

Unpredictable vibrations and disturbances may be occuredin industrial environment to create relative drifts.

I.2 CONCEPT OF THE INCREMENTAL IMAGE PROCESSING

Even though a high resolution primary camera is used, the unpredictable errors cannot be compensated totally, especially the item (5). Therefore, a methodology of incremental image processing is introduced to deal with this problem. The application of the methodology must have the following assumptions,

1. The workpiece has been identified by the primary image processing
2. Only limited amount of drifts are allowed.

The methodology of the incremental image processing is to detect any incremental change of the workpiece in realtime. By incorporating an auxiliary camera suitably attached to the end-effector of the robot, it appears to be an efficient and

economical method to solve the above problems and the workcell becomes possible to achieve a greater degree of adaptive containment of the system errors. The resulting workcell will be suitable for a wider range of industrial applications. Many researchers[29,34,66,90] also use the wrist-mounted camera for absolute detection. Here the application of the auxiliary camera is different. The approach is to use the primary camera to approximately identify the location and orientation θ of the workpiece on the conveyor. It also identifies the locations of 1 or more specified invariant critical features which are called the *critical points* in the latter discussions. While the robot achieves rendezvous with the workpiece, the robot will naturally rotate and approach its end-effector with the auxiliary camera to the processed orientation θ such that the window of the auxiliary camera is concentrated on the designated invariant critical features of the workpiece. Under such situation, the secondary partial workpiece image may have a better resolution to process. The current positions of these critical points are compared with those obtained in the primary image processing. Then, the incremental changes of the position and orientation can be calculate through the transformations described in section I.5.

I.3 Incremental Image Acquisition System

The incremental image acquisition system shown in the fig.I.1 consists of the following three modules :

- auxiliary binary camera
- data transceiver module
- flash control module

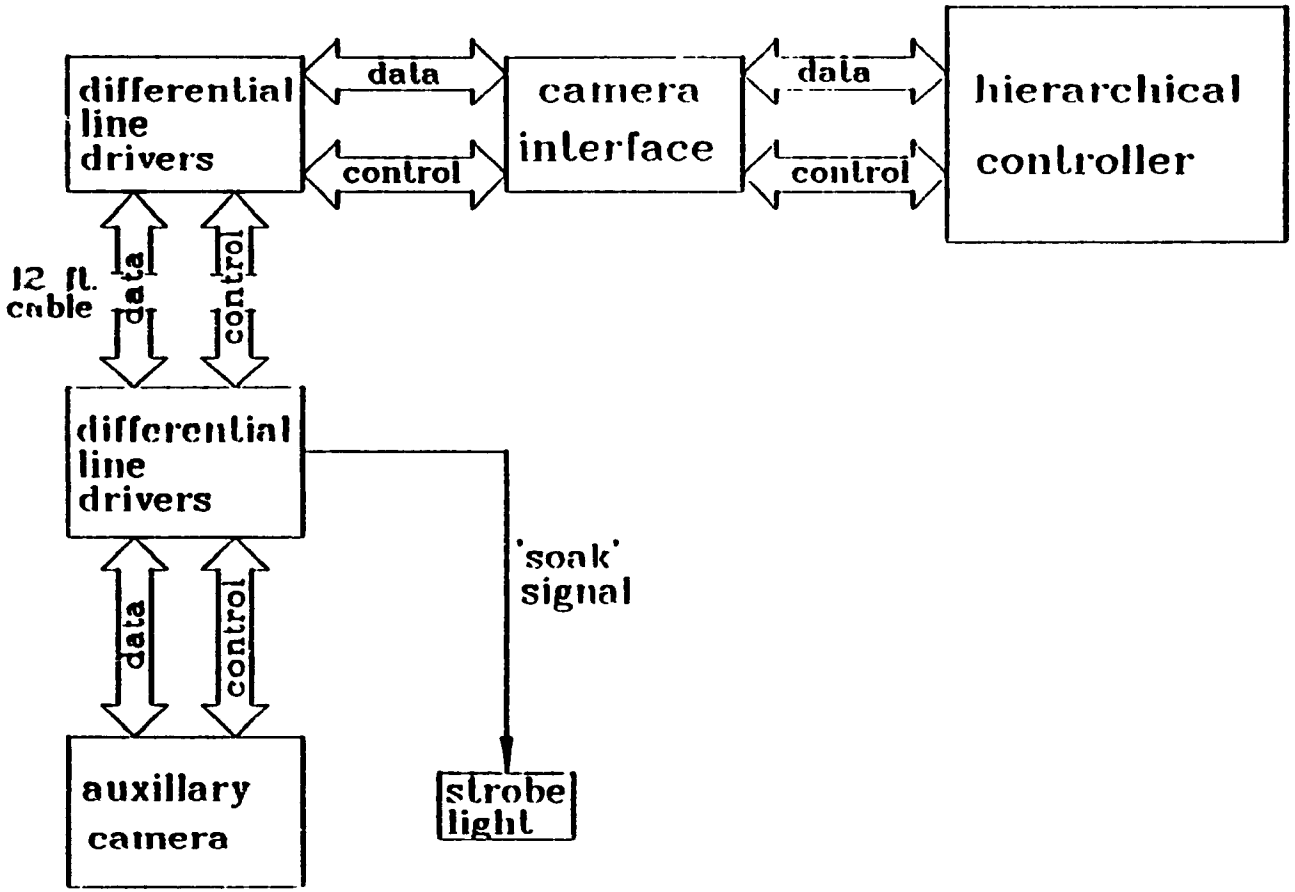


Fig.I.1 Incremental Image Acquisition Schematics

A MicronEye binary camera is used as the auxiliary camera. The heart of the MicronEye is an OpticRAM. The OpticRAM is composed of 65536 individual pixels. These pixels are organised into two rectangles of 128 x 256 pixels each. Each array of cells is separated by an optical "dead" zone of about 25 elements in width. The electronics in the MicroEye provide an interface between the OpticRAM and computer. It also provides a means by which the MicronEye can receive commands from the computer. Unlike the idetix camera, there is no hardware here to provide the DMA action. Inputing image from the interface is to poll the status register and then inport the image data. The details of this camera operation can be referred to the reference[64].

The second module is the data transeceiver module. This module is basically composed of two driver circuits - one is installed on the camera interface board in the host computer and another one is installed at the camera side.

The lighting source of the wrist-mounted auxiliary camera has to be compact such that the arm movement would not be affected. The better choice is to use a strobe light because of its compactness and sufficient light intensity. It is triggered by the camera 'SOAK' signal. The limitation of this lighting system is that frames cannot be taken continuously because of the strobe light charging time. The details of the transeceiver and the strobe light control circuit can be referred to the Appendix K.

I.4 CRITICAL FEATURES

Before introducing the technique, a concept of critical points is described here. A critical point is defined as the intersection

point of two lines or curves which are part of the profile of the workpiece. Those which are within the workpiece boundaries are defined as the *'physical critical points'* (see fig.I.2a). Otherwise, are called *'virtual critical points'* (see fig.I.2b). Three important types of critical points and they are generated by

1. two straight lines
2. two polynomial curves
3. one straight line and one polynomial curve

Following such definition, it is possible to locate certain critical points. Appendix L shows that critical points can be identified in most of the industrial components. In order to locate the critical point accurately, a statistical approach is recommended to reconstruct the profile of the workpiece by best-fit equations.

<u>case 1 :Two Straight Lines</u>

Fig.I.2a shows two physical critical points A and B which are the intersections of the lines l_1, l_2 and l_3. Fig.I.2b shows two virtual critical points C and D which are the intersections of the lines l_4, l_5 and l_6. Supposing that the straight lines l_1 and l_2 are defined by n points each. Using the method of least square fit, the equation of l_1 and l_2 can be represented as the following linear equations,

$$\begin{aligned} l_1 &: \quad y_{1i} = a_1 x_{1i} + a_0 \\ l_2 &: \quad y_{2i} = b_1 x_{2i} + b_0 \end{aligned} \qquad \ldots(I.1)$$

where $1 \leq i \leq n$

a_1 and b_1 are the gradients of the line l_1 and l_2

a_0 and b_0 are the zero intercept of the line l_1 and l_2

Using the least square fit method[32], the value of a_0, a_1, b_0, and b_1 are evaluated as follows,

$$a_1 = \frac{n\sum_{i=1}^{n} x_{1i}y_{1i} - \sum_{i=1}^{n} x_{1i} \sum_{i=1}^{n} y_{1i}}{n\sum_{i=1}^{n} x_{1i}^2 - \left[\sum_{i=1}^{n} x_{1i}\right]^2}$$

$$a_0 = \frac{\sum_{i=1}^{n} x_{1i}^2 \sum_{i=1}^{n} y_{1i} - \sum_{i=1}^{n} x_{1i}y_{1i} \sum_{i=1}^{n} x_{1i}}{n\sum_{i=1}^{n} x_{1i}^2 - \left[\sum_{i=1}^{n} x_{1i}\right]^2}$$

$$b_1 = \frac{n\sum_{i=1}^{n} x_{2i}y_{2i} - \sum_{i=1}^{n} x_{2i} \sum_{i=1}^{n} y_{2i}}{n\sum_{i=1}^{n} x_{2i}^2 - \left[\sum_{i=1}^{n} x_{2i}\right]^2}$$

$$b_0 = \frac{\sum_{i=1}^{n} x_{2i}^2 \sum_{i=1}^{n} y_{2i} - \sum_{i=1}^{n} x_{2i}y_{2i} \sum_{i=1}^{n} x_{2i}}{n\sum_{i=1}^{n} x_{2i}^2 - \left[\sum_{i=1}^{n} x_{2i}\right]^2}$$

...(I.2)

The intersection of l_1 and l_2 yields the the critical point $A(x_a, y_a)$. Then, it can be shown that

$$x_a = \frac{b_o - a_o}{a_1 - b_1}$$

$$y_a = a_1\left[\frac{b_o - a_o}{a_1 - b_1}\right] + a_o$$

...(I.3)

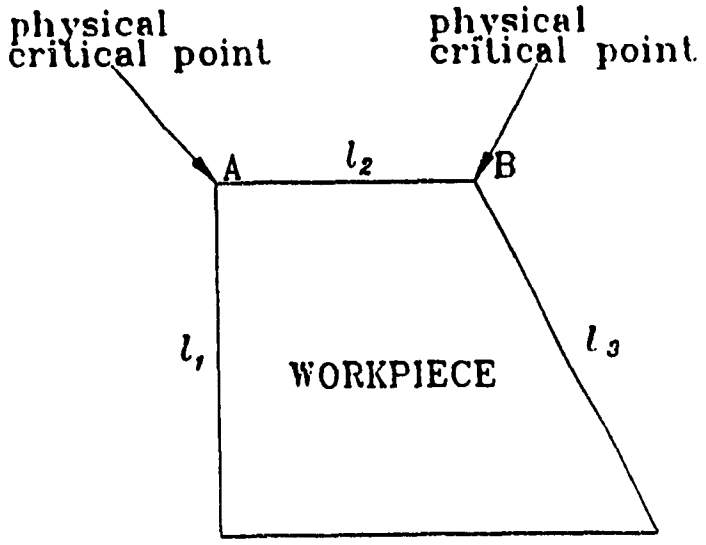


Fig. I.2a Physical Critical Point of Two Straight Lines

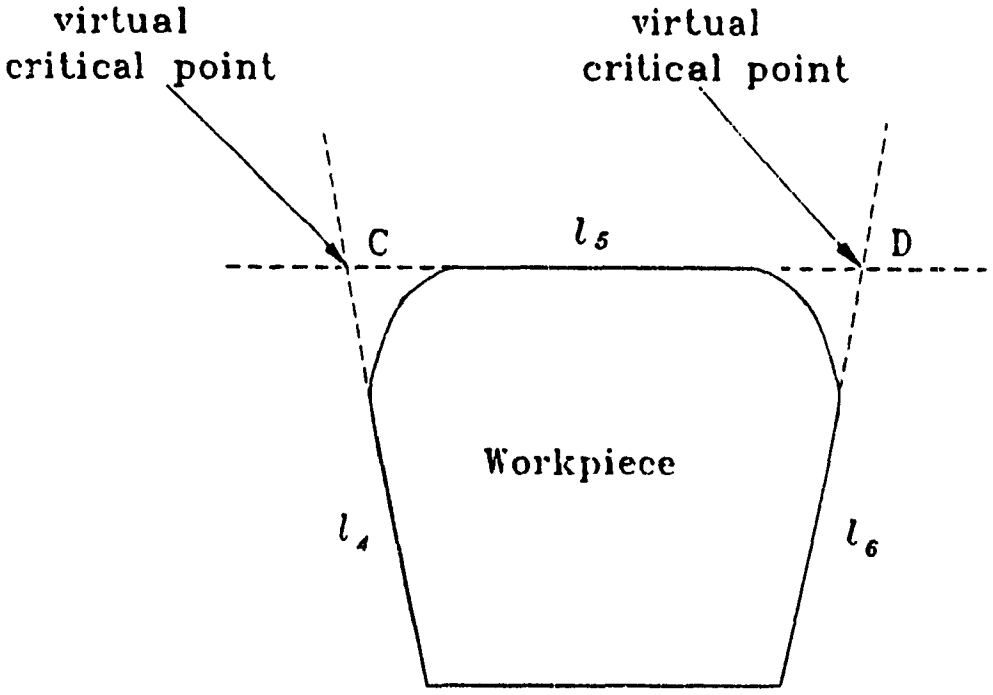


Fig. I.2b Virtual Critical Point of Two Straight Lines

Case 2 : Two Polynomial Curves

Fig.I.3 shows a physical critical point $F(x_f, y_f)$ at the intersections of two polynomials P_1 and P_2. Supposing that the curves P_1, P_2 are estimated to be second-order polynomials (m=2) and each also has n points. Then, P_1 and P_2 can be represented by the following equations,

$$P_1(x_{1i}) : y_{1i} = a_o + a_1 x_{1i} + a_2 x_{1i}^2$$
$$P_2(x_{2i}) : y_{2i} = b_o + b_1 x_{2i} + b_2 x_{2i}^2 \qquad \ldots(I.4)$$

where $1 \le i \le n$

a_o, a_1, a_2, b_0, b_1 and b_2 are the coefficients of the polynomials P_1 and P_2

Using the least square fit method to obtain the constants a_0, a_1, a_2, b_0, b_1, and b_2 statistically, the expression of the least-square errors are

$$E_1 = \sum_{i=0}^{n} \left(y_{1i} - P_1(x_{1i}) \right)^2$$

$$E_2 = \sum_{i=0}^{n} \left(y_{2i} - P_2(x_{2i}) \right)^2 \qquad \ldots(I.5)$$

E_1 and E_2 are to be minimum, the following conditions are satisfied,

$$\frac{\partial E_1}{\partial a_j} = -2 \sum_{i=0}^{n} y_{1i} x_{1i}^j + 2 \sum_{k=0}^{m} a_k \sum_{i=0}^{n} x_{1i}^{j+k} = 0$$

$$\therefore \quad \sum_{i=0}^{n} y_{1i} x_{1i}^j = \sum_{k=0}^{m} a_k \sum_{i=0}^{n} x_{1i}^{j+k} \qquad \begin{array}{l} 0 \le i \le n \\ 0 \le j \le 2 \\ 0 \le m \le 2 \\ 0 \le k \le 2 \end{array}$$

$$a_0 \sum_{i=0}^{n} x_{1i}^0 + a_1 \sum_{i=0}^{n} x_{1i}^1 + a_2 \sum_{i=0}^{n} x_{1i}^2 = \sum_{i=0}^{n} y_{1i} x_{1i}^0$$

$$a_0 \sum_{i=0}^{n} x_{1i}^1 + a_1 \sum_{i=0}^{n} x_{1i}^2 + a_2 \sum_{i=0}^{n} x_{1i}^3 = \sum_{i=0}^{n} y_{1i} x_{1i}^1$$

$$a_0 \sum_{i=0}^{n} x_{1i}^2 + a_1 \sum_{i=0}^{n} x_{1i}^3 + a_2 \sum_{i=0}^{n} x_{1i}^4 = \sum_{i=0}^{n} y_{1i} x_{1i}^2$$

...(I.6)

Hence, a_0, a_1 and a_2 are obtained by solving equation (I.6).

Similarly, the condition for polynomial P_2

$$\frac{\partial E_2}{\partial b_j} = -2 \sum_{i=0}^{n} y_{2i} x_{2i}^j + 2 \sum_{k=0}^{m} a_k \sum_{i=0}^{n} x_{2i}^{j+k}$$

$$\therefore \quad \sum_{i=0}^{n} y_{2i} x_{2i}^j = \sum_{k=0}^{m} a_k \sum_{i=0}^{n} x_{2j}^{j+k} \qquad \begin{array}{l} 0 \le i \le n \\ 0 \le j \le 2 \\ 0 \le m \le 2 \\ 0 \le k \le 2 \end{array}$$

$$b_0 \sum_{i=0}^{n} x_{2i}^0 + b_1 \sum_{i=0}^{n} x_{2i}^1 + b_2 \sum_{i=0}^{n} x_{2i}^2 = \sum_{i=0}^{n} y_{2i} x_{2i}^0$$

$$b_0 \sum_{i=0}^{n} x_{2i}^1 + b_1 \sum_{i=0}^{n} x_{2i}^2 + b_2 \sum_{i=0}^{n} x_{2i}^3 = \sum_{i=0}^{n} y_{2i} x_{2i}^1$$

$$b_0 \sum_{i=0}^{n} x_{2i}^2 + b_1 \sum_{i=0}^{n} x_{2i}^3 + b_2 \sum_{i=0}^{n} x_{2i}^4 = \sum_{i=0}^{n} y_{2i} x_{2i}^2$$

...(I.7)

Hence, b_0, b_1 and b_2 are obtained. The coordinate $F(x_f, y_f)$ is found by solving (I.4)

$$a_o + a_1 x_f + a_2 x_f^2 = b_o + b_1 x_f + b_2 x_f^2$$

$$x_f = -\frac{1}{2}\left[\frac{a_1 - b_1}{a_2 - b_2}\right] \pm \left[\frac{1}{4}\left[\frac{a_1 - b_1}{a_2 - b_2}\right]^2 - \left[\frac{a_0 - b_0}{a_2 - b_2}\right]\right]^{1/2}$$

$$y_f = a_0 + a_1 x_f + a_2 x_f^2$$

...(I.8)

There are two sets of (x_f, y_f). The set which is closer to the primary image will be chosen as the solution.

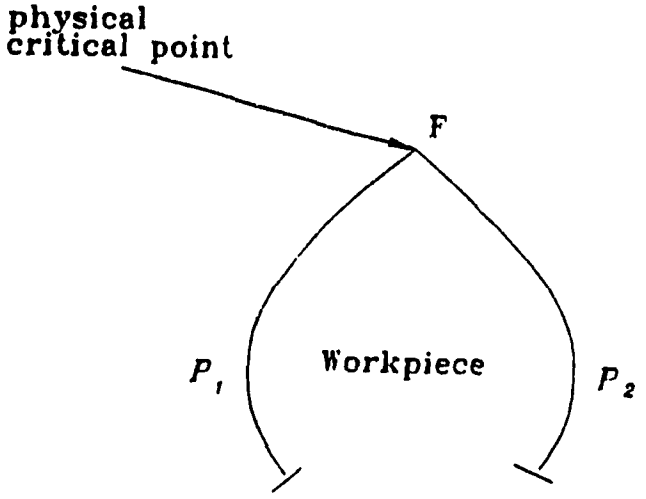


Fig.I.3 Physical Critical Point of Two Polynomial Curves

Case 3 : One Polynomial Curve and One Straight Line

Fig.I.4 shows a critical point $H(x_h, y_h)$ as the intersection of the stright line l_5 and polynomial P_5 and each is defined by n-points,

$$l_5 : y_{5i} = b_1 x_{5i} + b_0$$

$$P_5 : y_{5i} = a_0 + a_1 x_{5i} + a_2 x_{5i}^2$$

...(I.9)

where $1 \leq i \leq n$

The constants a_0, a_1, a_2, b_0, and b_1 of the equations of l_5 and P_5 can be obtained by using the statistical method in case 1 and case 2. At the critical point (x_h, y_h),

$$x_h = -\frac{1}{2}\left[\frac{a_1 - b_1}{a_2}\right] \pm \left[\frac{1}{4}\left[\frac{a_1 - b_1}{a_2}\right]^2 - \left[\frac{a_0 - b_0}{a_2}\right]\right]^{1/2}$$

$$y_f = a_0 + a_1 x_f + a_2 x_f^2$$

...(I.10)

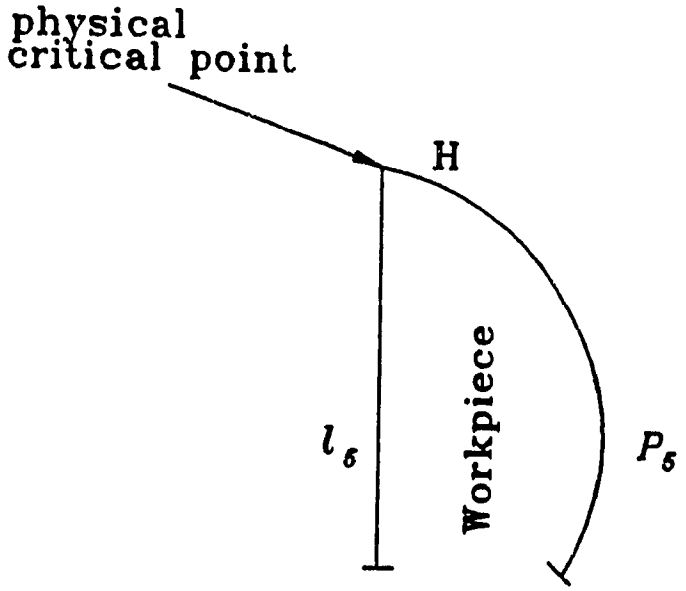
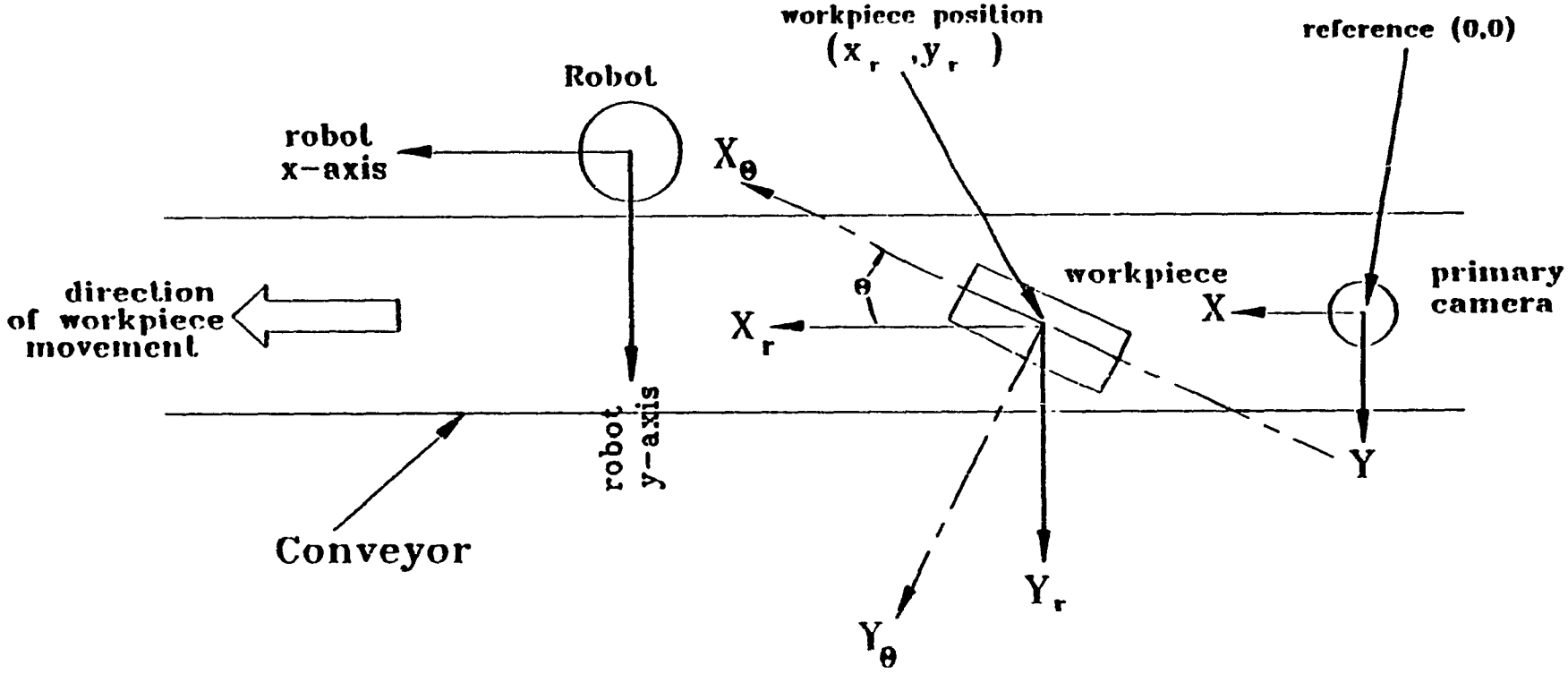


Fig. I.4 Physical Critical Points of a Straight Line and a Polynomial Curve

I.5 ANALYSIS OF THE INCREMENTAL IMAGE PROCESSING TECHNIQUE

This section describes the transformation necessarily to obtain the position and orientation drift of the workpiece from the auxiliary image. First of all, two coordinate systems (fig.I.5) have to be defined. The first one is the reference coordinate. It is parallel to the reference frame with its origin located at the centre of the workpiece. Its axes are the same as the robot cartesian axes. It is subscripted with an 'r' in the latter formulation. Another coordinate system is the workpiece coordinate. It axes are determined by the orientation of the workpiece θ. The coordinate is subscripted with a 'Θ'. The relationship between two coordinate systems are related by the following transformation[28],

$$\begin{bmatrix} x_\Theta \\ y_\Theta \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x_r \\ y_r \end{bmatrix} \qquad \ldots(I.11)$$



X_r Y_r are the axes of the reference coordinate

X_θ Y_θ are the axes of the workpiece coordinate

Fig.I.5 Coordinate Systems in Incremental Image Processing Technique

There are two types of offsets which should be taken into consideration. The first type of offset is the distance between the camera and the wirst. Fig.I.6 shows that the auxiliary camera is mounted on the axis of joint #6 with an vector offset r_w. The second type of offset is the distance between the workpiece position and the critical features. Supposing that a workpiece in fig.I.7 has been identified. The position vector r_r (x_r, y_r) and orientation θ are obtained from the primary image processing. The auxiliary camera is then rotated θ degree and is directed to a position (x_p, y_p) such that the window of the auxiliary camera contains the critical features. This is achieved by directing the centre of the window to a pre-determined point of the workpiece called the calibrating position M. Let the position vector of the robot end-effector be P_r. The offset vector from the workpiece position to the calibrating position M is r_p. Let the angle between the axis X_Θ and vector r_p be α. While the auxiliary camera is taking the secondary image , the robot end-effector position is

$$P_r = r_r + r_p + r_w$$

$$\begin{bmatrix} x_p \\ y_p \end{bmatrix} = \begin{bmatrix} x_r \\ y_r \end{bmatrix} + \begin{bmatrix} |r_p| \cos (\theta + \alpha) \\ -|r_p| \sin (\theta + \alpha) \end{bmatrix} + \begin{bmatrix} |r_w| \cos \theta \\ -|r_w| \sin \theta \end{bmatrix}$$

$$\begin{bmatrix} x_p \\ y_p \end{bmatrix} = \begin{bmatrix} x_r \\ y_r \end{bmatrix} + \begin{bmatrix} \cos (\theta + \alpha) & \cos \theta \\ -\sin (\theta + \alpha) & -\sin \theta \end{bmatrix} \begin{bmatrix} |r_p| \\ |r_w| \end{bmatrix}$$

If the workpiece has no drifts, the compensation vector r_e is

$$r_e = -r_p - r_w \qquad \ldots(I.12)$$

If the workpiece undergoes a drift Δr_r(fig.I.8) with the components Δx_r and Δy_r, the resultant compensation vector r'_e becomes,

$$r'_e = - r_w - r_p + \Delta r_r \qquad \ldots(I.13)$$

The components of the vector r'_e (x_e, y_e) are as follows,

$$\begin{bmatrix} x_e \\ y_e \end{bmatrix} = \begin{bmatrix} -|r_w| \cos\theta \\ |r_w| \sin\theta \end{bmatrix} + \begin{bmatrix} -|r_p| \cos(\theta + \alpha) \\ |r_p| \sin(\theta + \alpha) \end{bmatrix} + \begin{bmatrix} \Delta x_r \\ \Delta y_r \end{bmatrix}$$

$$\begin{bmatrix} x_e \\ y_e \end{bmatrix} = \begin{bmatrix} -\cos(\theta + \alpha) & -\cos\theta \\ \sin(\theta + \alpha) & \sin\theta \end{bmatrix} \begin{bmatrix} |r_p| \\ |r_w| \end{bmatrix} + \begin{bmatrix} \Delta x_r \\ \Delta y_r \end{bmatrix}$$

The vector r_w is the invariant mechanical offset of the camera mounting and the vector r_p is the invariant geometrical property of the workpiece. Once the Δr_r is solved, the resultant compensation vector r'_e can be obtained.

Considering an example of a rectanglar workpiece (a, b, c, d) in fig.I.9, the position and orientation of the workpiece obtained from the primary image processing is represented by (x_r, y_r) and θ. The four critical points are a, b, c and d. The position vectors of the critical points (a, b, c, d) w.r.t the position (x_r, y_r) are $r_{a\Theta}$, $r_{b\Theta}$, $r_{c\Theta}$, $r_{d\Theta}$ in workpiece coordinate. Supposing that there is a drift in position and orientation while the workpiece travels on the conveyor resulting in a', b', c' and d'(fig.I.10). While the robot achieve rendezvous with the workpiece at the interception phase, the auxiliary camera is dictated to rotate θ degree. In addition, the camera is directed to concentrate its window on certain critical points(fig.I.11). Because of the finite width of the window, only limited amount of the drifts of the critical points are allowed. The drifts of the position and orientation are represented by a translation vector Δr_r and a rotation vector $\Delta\Theta$.

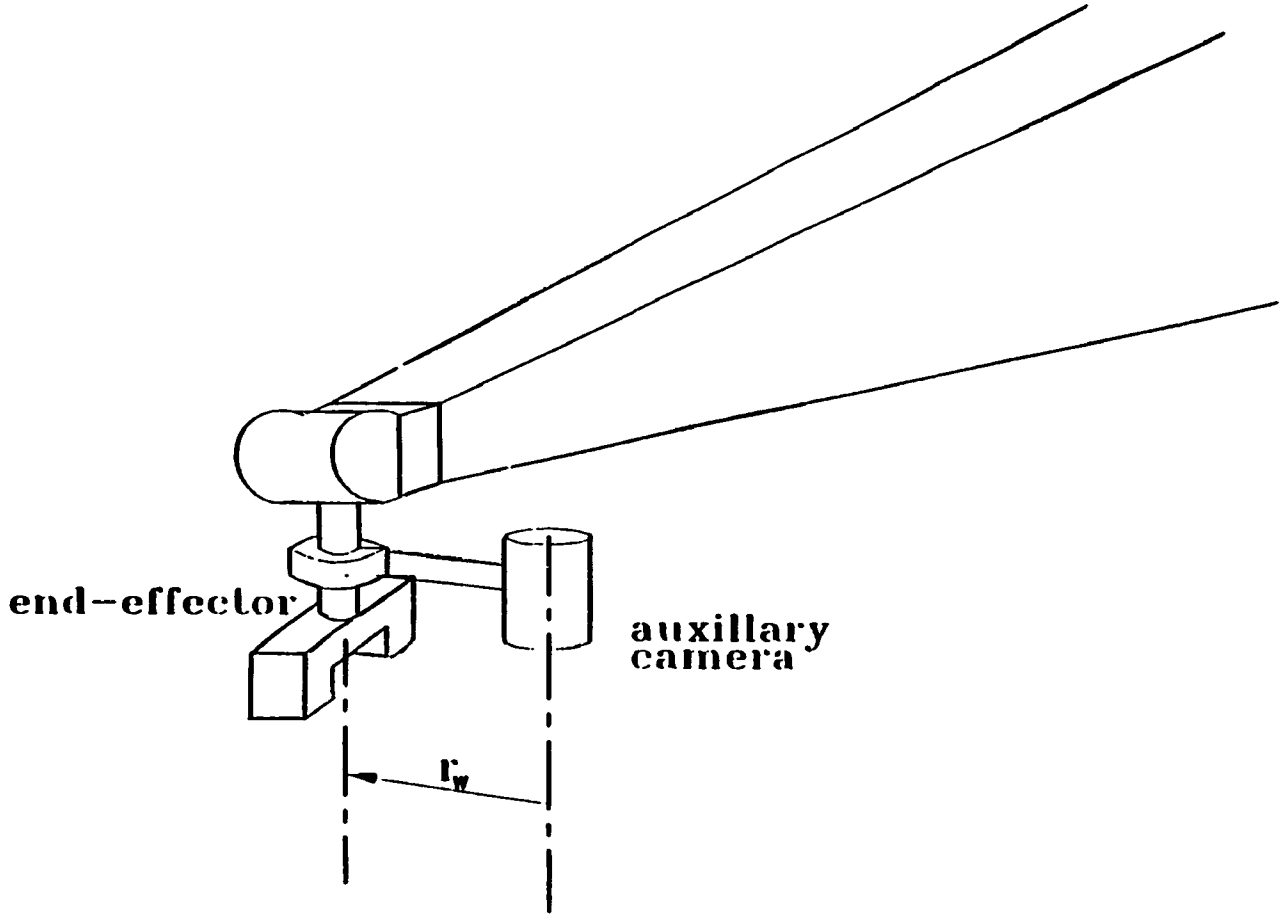


Fig.I.6 Auxiliary Camera Mounting

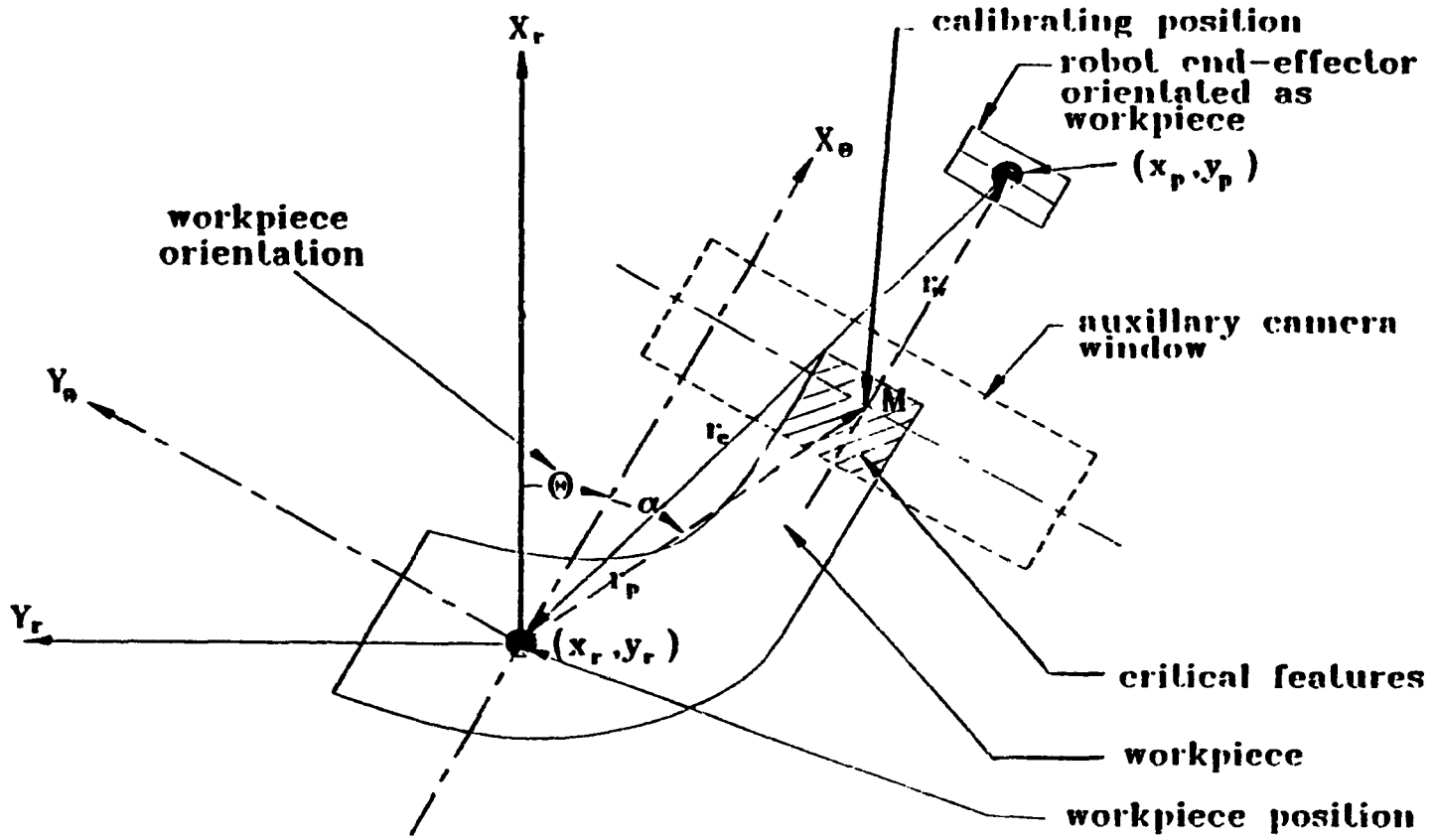


Fig. I.7 End-Effector and Critical Feature Offset ($r_p + r_w$)

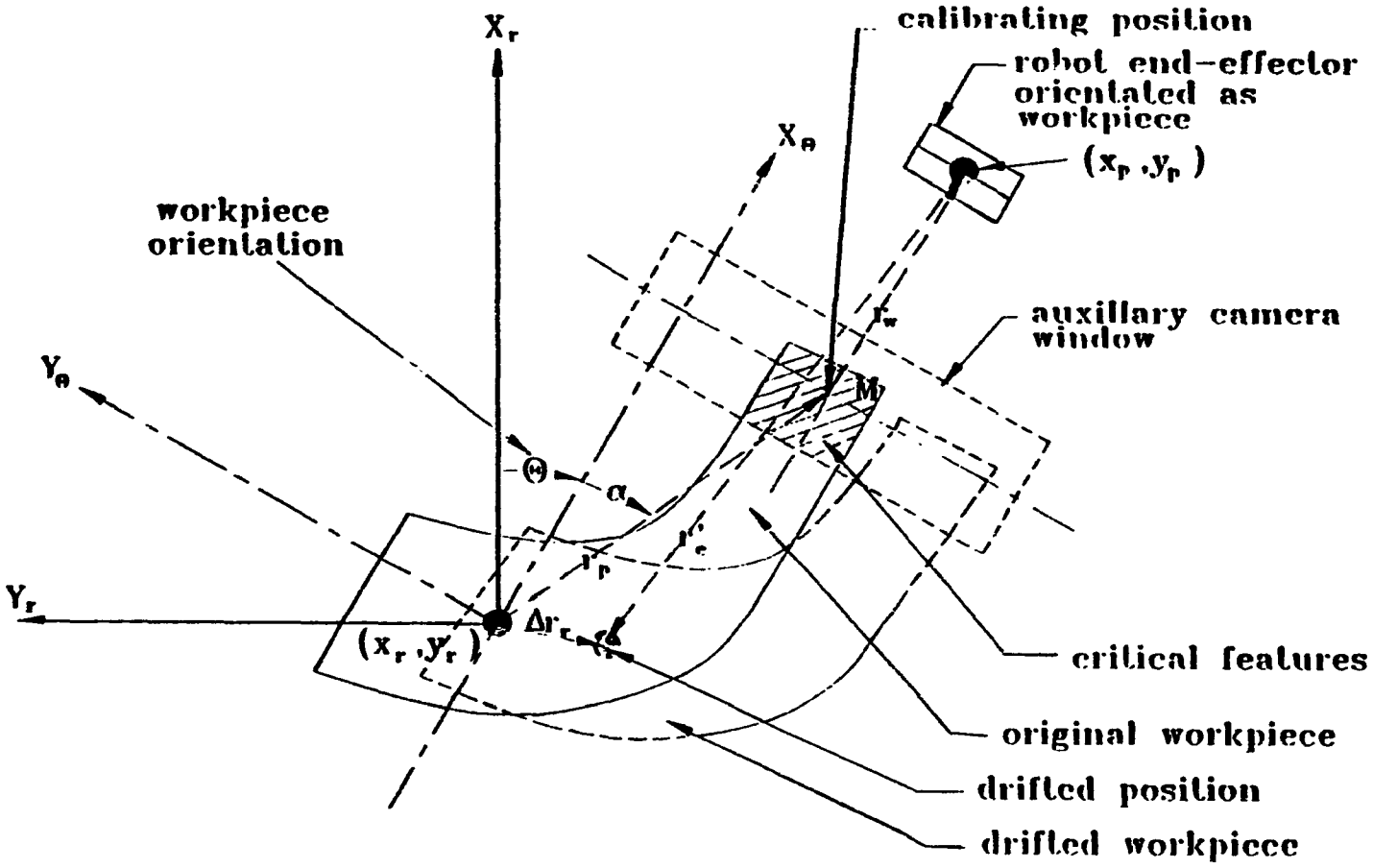


Fig. I.8 Compensation Vector with drifts ($r'_e = -r_w - r_p + \Delta r_r$)

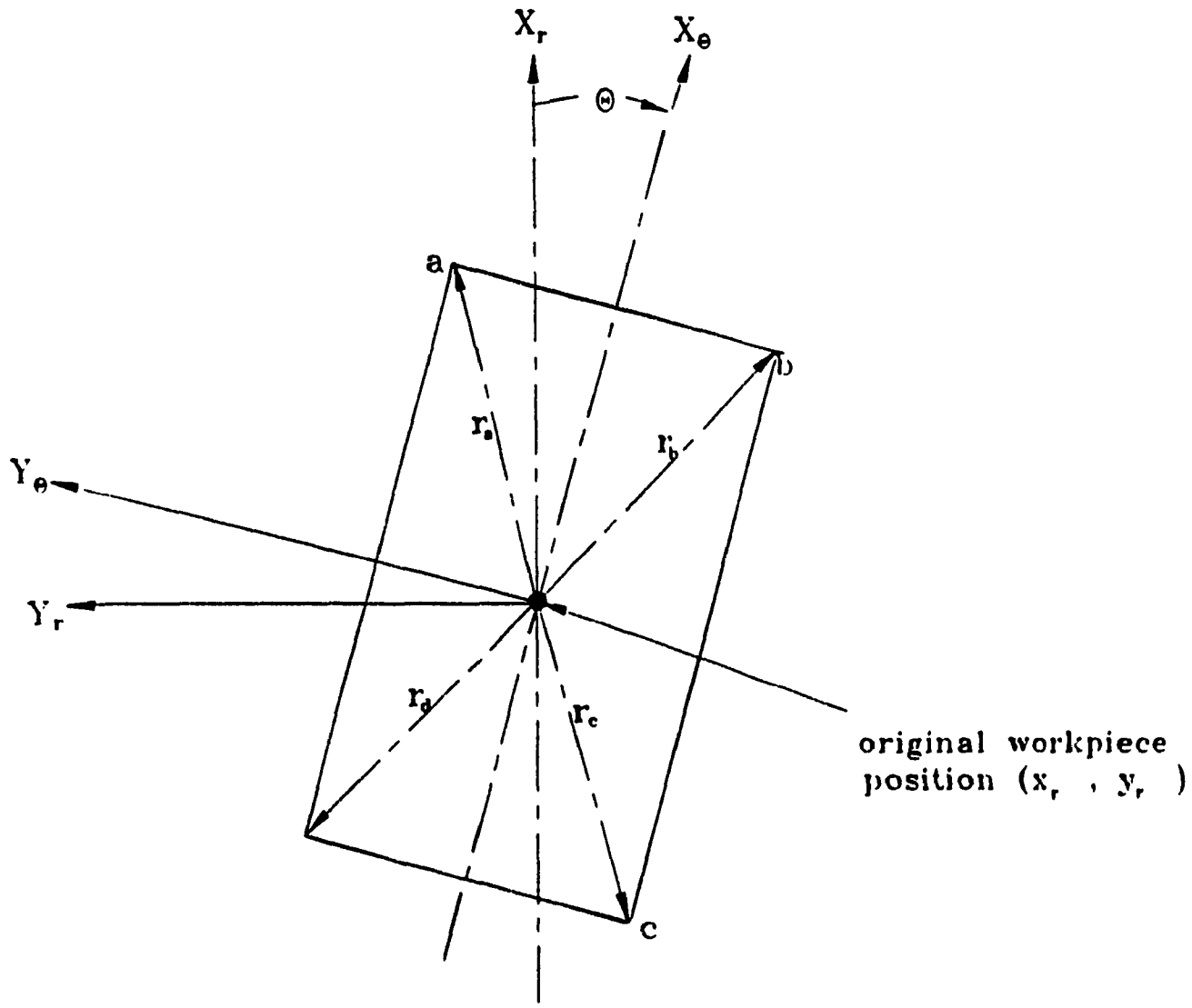


Fig.I.9 Original Workpiece

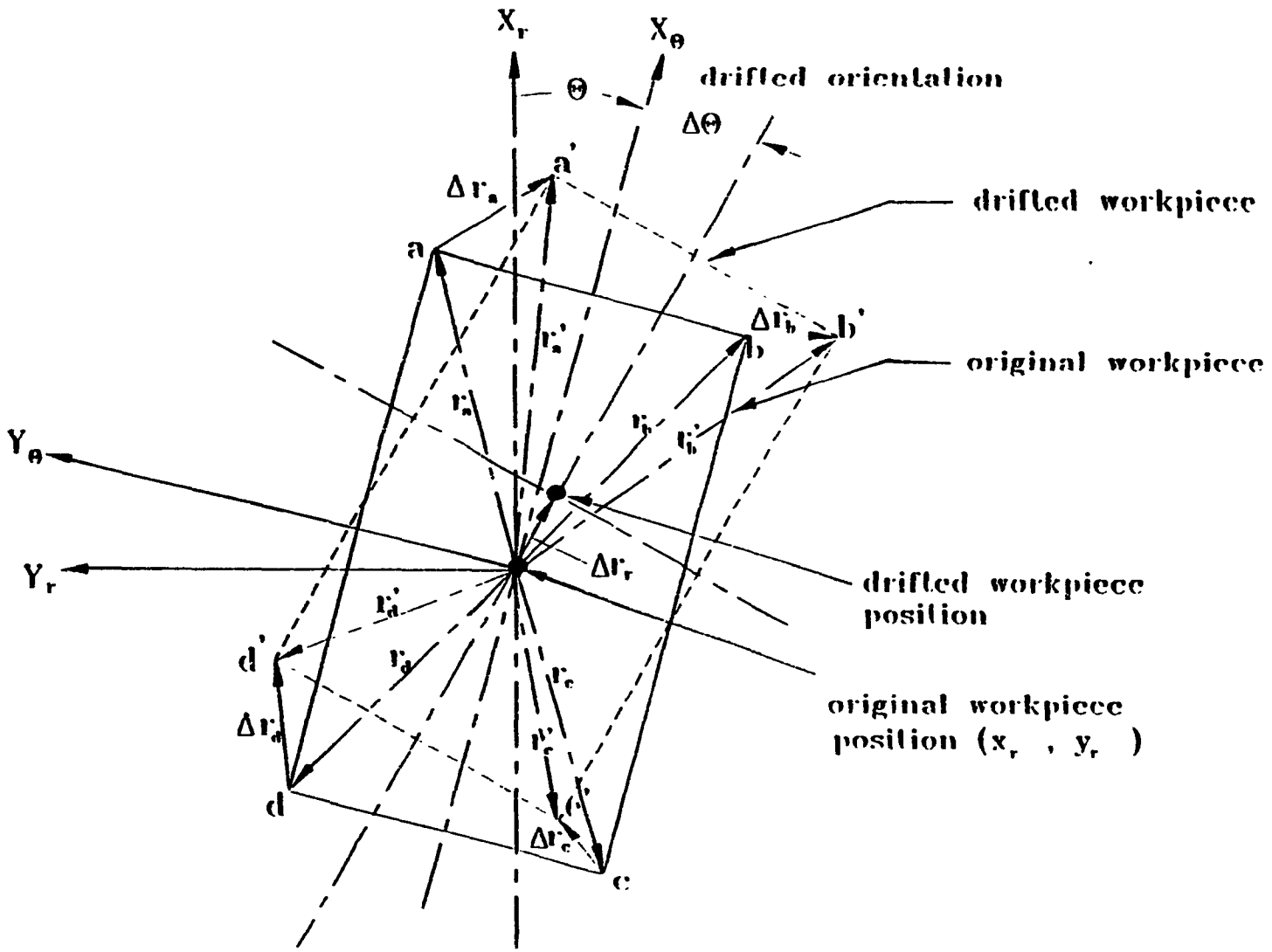


Fig.I.10 Drifted Workpiece

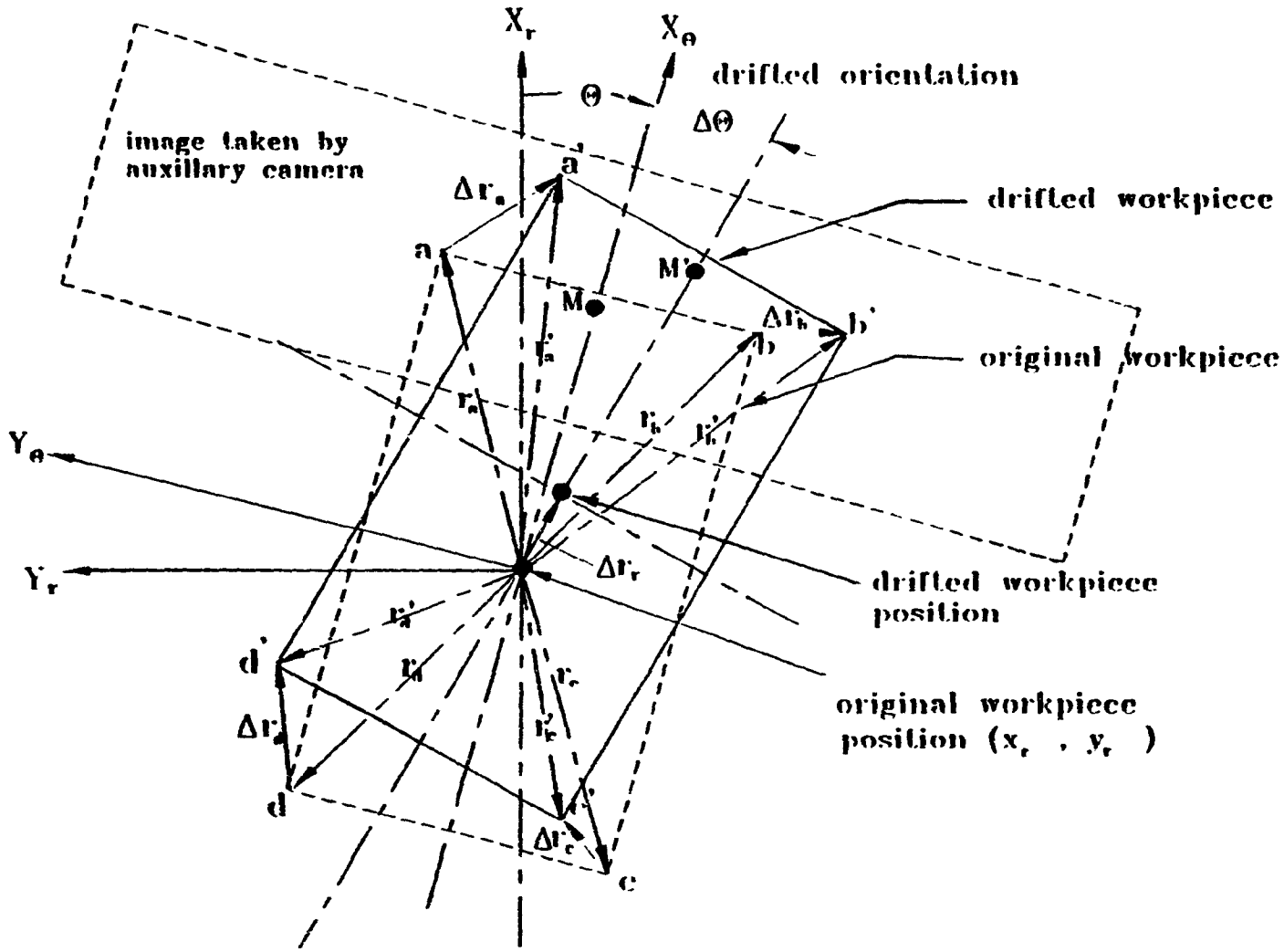


Fig.I.11 Auxiliary Camera Taking the Secondary Image

Let Δx_Θ is the position drift in x-direction in the workpiece coordinate

Δy_Θ is the position drift in y-direction in the workpiece coordinate

$\Delta\Theta$ is the drift in orientation

trans$[\Delta r_\Theta]$ be a translational operator for a translation Δr_Θ and it is expressed by

$$\text{trans}\,[\,\Delta\, r_\Theta] = \begin{bmatrix} 1 & 0 & \Delta\, x_\Theta \\ 0 & 1 & \Delta\, y_\Theta \\ 0 & 0 & 1 \end{bmatrix}$$

Let the rot$[\, z_\Theta,\ \Delta\Theta\ ]$ be a rotational operator for a rotation $\Delta\Theta$ about the normal axis z_Θ and it is expressed by

$$\text{rot}\,[z,\ \Delta\Theta] = \begin{bmatrix} \cos\Delta\Theta & \sin\Delta\Theta & 0 \\ -\sin\Delta\Theta & \cos\Delta\Theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Hence the drifted vectors of the critical point a′, b′, c′ and d′ can be expressed as follows,

$$r'_{a\Theta} = \text{trans}\,[\,\Delta r_\Theta]\ \text{rot}\,[\,z_\Theta,\ \Delta\Theta\,]\ r_{a\Theta}$$

$$r'_{b\Theta} = \text{trans}\,[\,\Delta r_\Theta]\ \text{rot}\,[\,z_\Theta,\ \Delta\Theta\,]\ r_{b\Theta}$$

$$r'_{c\Theta} = \text{trans}\,[\,\Delta r_\Theta]\ \text{rot}\,[\,z_\Theta,\ \Delta\Theta\,]\ r_{c\Theta}$$

$$r'_{d\Theta} = \text{trans}\,[\,\Delta r_\Theta]\ \text{rot}\,[\,z_\Theta,\ \Delta\Theta\,]\ r_{d\Theta}$$

...(I.14)

Take the critical point 'a' in equation (I.14), it becomes,

$$\begin{bmatrix} x'_{a\theta} \\ y'_{a\theta} \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & \Delta x_\theta \\ 0 & 1 & \Delta y_\theta \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos \Delta\theta & \sin \Delta\theta & 0 \\ -\sin \Delta\theta & \cos \Delta\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_{a\theta} \\ y_{a\theta} \\ 1 \end{bmatrix}$$

$$r'_\theta = \begin{bmatrix} x'_{a\theta} \\ y'_{a\theta} \\ 1 \end{bmatrix} = \begin{bmatrix} x_{a\theta} \cos \Delta\theta + y_{a\theta} \sin \Delta\theta + \Delta x_\theta \\ -x_{a\theta} \sin \Delta\theta + y_{a\theta} \cos \Delta\theta + \Delta y_\theta \\ 1 \end{bmatrix} \quad \ldots(I.15)$$

(I.15) yields two algebraic equations containing three unknowns ($\Delta x_\theta, \Delta y_\theta, \Delta\theta$). Hence, an additional critical point is required to solve the equations. By taking the critical points 'a' and 'b',

$$x'_{a\theta} = x_{a\theta} \cos \Delta\theta + y_{a\theta} \sin \Delta\theta + \Delta x_\theta \quad \ldots(I.16)$$

$$x'_{b\theta} = x_{b\theta} \cos \Delta\theta + y_{b\theta} \sin \Delta\theta + \Delta x_\theta \quad \ldots(I.17)$$

Subtracting (I.16) from (I.17) to eliminate the Δx_θ term,

$$(x'_{a\theta} - x'_{b\theta}) = (x_{a\theta} - x_{b\theta}) \cos \Delta\theta + (y_{a\theta} - y_{b\theta}) \sin \Delta\theta \quad \ldots(I.18)$$

Similiarly,

$$(y'_{a\theta} - y'_{b\theta}) = - (x_{a\theta} - x_{b\theta}) \sin \Delta\theta + (y_{a\theta} - y_{b\theta}) \cos \Delta\theta \quad \ldots(I.19)$$

Let

$$x'_{ab\theta} = x'_{a\theta} - x'_{b\theta} \qquad y'_{ab\theta} = y'_{a\theta} - y'_{b\theta}$$

$$x_{ab\theta} = x_{a\theta} - x_{b\theta} \qquad y_{ab\theta} = y_{a\theta} - y_{b\theta}$$

(I.18) and (I.19) becomes,

$$x'_{ab\theta} = x_{ab\theta} \cos \Delta\theta + y_{ab\theta} \sin \Delta\theta \quad \ldots(I.20)$$

$$y'_{ab\theta} = - x_{ab\theta} \sin \Delta\theta + y_{ab\theta} \cos \Delta\theta \quad \ldots(I.21)$$

Solving (I.20) and (I.21) by introducing an auxiliary angle $\Delta\delta$

Let

$$\cos \Delta\delta = \frac{x_{ab\theta}}{(x_{ab\theta}^2 + y_{ab\theta}^2)^{1/2}} \qquad \sin \Delta\delta = \frac{y_{ab\theta}}{(x_{ab\theta}^2 + y_{ab\theta}^2)^{1/2}}$$

Equation (I.20) becomes,

$$x'_{ab\theta} = (x_{ab\theta}^2 + y_{ab\theta}^2)^{1/2} (\cos \Delta\delta \cos \Delta\theta + \sin \Delta\delta \sin \Delta\theta)$$

$$= (x_{ab\theta}^2 + y_{ab\theta}^2)^{1/2} \cos(\Delta\delta - \Delta\theta) \qquad \ldots(I.22)$$

$$\therefore \quad \Delta\delta - \Delta\theta = \cos^{-1}\left[\frac{x'_{ab\theta}}{(x_{ab\theta}^2 + y_{ab\theta}^2)^{1/2}}\right]$$

$$\Delta\delta = \frac{x_{ab\theta}}{(x_{ab\theta}^2 + y_{ab\theta}^2)^{1/2}}$$

$$\therefore \quad \Delta\theta = \cos^{-1}\left[\frac{x_{ab\theta}}{(x_{ab\theta}^2 + y_{ab\theta}^2)^{1/2}}\right] - \cos^{-1}\left[\frac{x'_{ab\theta}}{(x_{ab\theta}^2 + y_{ab\theta}^2)^{1/2}}\right] \qquad \ldots(I.23)$$

The translation drifts (Δx_θ and Δy_θ) can be solved by substituting the value of $\Delta\theta$ in equation (I.15).

$$r'_\theta = \begin{bmatrix} x'_{a\theta} \\ y'_{a\theta} \\ 1 \end{bmatrix} = \begin{bmatrix} x_{a\theta} \cos \Delta\theta + y_{a\theta} \sin \Delta\theta + \Delta x_\theta \\ -x_{a\theta} \sin \Delta\theta + y_{a\theta} \cos \Delta\theta + \Delta y_\theta \\ 1 \end{bmatrix} \qquad \ldots(I.15)$$

Arranging the terms (Δx_Θ, Δy_Θ) of equation (I.15) to the left hand side,

$$\Delta r_\Theta = \begin{bmatrix} \Delta x_\Theta \\ \Delta y_\Theta \\ 1 \end{bmatrix} = \begin{bmatrix} x'_{a\Theta} - x_{a\Theta} \cos \Delta\Theta + y_{a\Theta} \sin \Delta\Theta \\ y'_{a\Theta} + x_{a\Theta} \sin \Delta\Theta - y_{a\Theta} \cos \Delta\Theta \\ 1 \end{bmatrix} \quad \ldots(I.24)$$

The vector Δr_Θ in equation (I.24) is expressed in the workpiece coordinate. In order to instruct the robot to compensate the drift and error in realtime, the vector of Δr_Θ has to be transformed into the robot coordinate Δr_r by the following transformation,

$$\Delta r_r = \begin{bmatrix} \Delta x_r \\ \Delta y_r \end{bmatrix} = \begin{bmatrix} \cos \Theta & \sin \Theta \\ -\sin \Theta & \cos \Theta \end{bmatrix} \Delta r_\Theta \quad \ldots(I.25)$$

The Δr_r is then substituted into the equation (I.13) to obtain the compensation vector r'_e.

I.6 IMPLEMENTATION RESULTS

The methodology has been implemented in a quasi-static condition (i.e. robot and workpiece velocities are zero). A rectanglar workpiece is placed under the wristed-mounted auxiliary camera with an known orientation θ and position. The auxiliary camera is rotated such that the camera axis is coincided with the workpiece longititude axis. In addition, the auxiliary camera window is concentrated on certain features which consists of two critical points ('a' and 'b'). The workpiece is then rotated and shifted separately. The translational drift Δr_Θ and orientation drift $\Delta\Theta$ can be obtained by the technique descibed in section I.5. The preliminary implementation results are listed in Tabls I.1 and I.2. The average detected error of the translational drifts is 1.6mm and the average detected error of the orientation drift is 0.45°.

It can be found that there are greater errors in th. negative sides. For the negative drifts, the area of the workpiece appeared in the auxiliary camera window is less. Therefore, there are less statistical data to identify the critical points accurately. Thus, the error of the detected drifts is appeared to be greater.

TABLE I.1 EXPERIMENTAL RESULTS OF THE INCREMENTAL IMAGE PROCESSING (POSITION DRIFTS)

actual position drift (mm)		*detected position drift (mm)*		*errors (mm)*	
Δx	Δy	Δx	Δy	Δ(Δx)	Δ(Δy)
trial #1					
6	6	7.594	7.367	1.594	1.367
4	4	5.344	3.330	1.344	-0.670
2	2	3.000	1.110	1.000	-0.890
-2	-2	-2.625	-3.885	-0.625	-1.885
-4	-4	-6.844	-6.660	-2.884	-2.660
-6	-6	-8.719	-9.435	-2.719	-3.435
trial #2					
6	6	7.166	6.497	1.166	0.497
4	4	5.344	3.330	1.344	-0.670
2	2	3.000	1.665	1.000	-0.335
-2	-2	-2.625	-3.885	-0.625	-1.885
-4	-4	-6.844	-6.660	-2.844	-2.660
-6	-6	-8.719	-9.435	-2.719	-3.435
trial #3					
6	6	7.166	6.497	1.166	0.497
4	4	5.344	3.330	1.344	-0.670
2	2	2.531	1.110	0.531	-0.890
-2	-2	-2.625	-3.330	-0.625	-1.330
-4	-4	-6.844	-7.215	-2.844	-3.215
-6	-6	-8.719	-9.435	-2.719	-3.435
		average abs. error=		1.620	1.690

TABLE I.2 EXPERIMENTAL RESULTS OF THE INCREMENTAL IMAGE PROCESSING (ORIENTATION DRIFTS)

actual orient drift (deg.)	*detected orient. drift (deg.)*	*error (deg.)*
trial #1		
6	6.15	0.15
4	4.31	0.31
2	2.76	0.76
0	0.00	0.00
-2	-1.84	0.16
-4	-3.07	0.93
-6	-5.23	0.77
trial #2		
6	6.46	0.46
4	4.30	0.30
2	2.76	0.76
0	0.00	0.77
-2	-1.54	0.46
-4	-3.69	0.31
-6	-5.23	0.77
trial #3		
6	6.15	0.15
4	4.39	0.39
2	2.45	0.45
0	0.00	0.00
-2	-1.85	0.15
-4	-3.38	0.62
-6	-5.23	0.77
	averge abs. error =	0.45

I.7 CONCLUSION

A statistical approach of an incremental image processing technique has been described. Not only does this methodology provides a fast and low cost vision system, but also it can compensate any unpredictable on-line relative drifts between the conveyor and the workpiece. The preliminary implementation results the average detected error of the translational drift to be 1.6mm and orientation drift to be 0.45°. By incorporating such auxiliary camera on the robot wrist, the low-cost robotic workcell would be suitable for more industrial applications.

APPENDIX J

Architecture of an Intelligent Robotic Workcell for Synchronisation Control

ARCHITECTURE OF AN INTELLIGENT ROBOTIC WORKCELL FOR SYNCHRONISATION CONTROL

R.M.H. Cheng
Professor of Engineering

S.C.L. Poon
Res. Assistant

Centre for Industrial Control
Department of Mechanical Engineering
Concordia University
Montréal, Canada

Abstract

This paper is to describe some possible synchronisation control schemes and the architecture of an intelligent robotic workcell which includes a robot, a camera station shaddling over a conveyor. The camera station is used to extract the position and orientation of a workpiece transferred via the conveyor belt. To accordance with the computed centroid and orientation, the robot would be driven to synchronise with the workpiece within the working envelop to perform predetermined operation. Three control schemes are described, evaluated and criticised in this paper. For the scheme which is judged to be the most suitable one, a methodology is introduced to determine an appropriate distance between the robot and the camera station in order to maximise the time for the robot to perform its operation within the envelop. Under such appropriate distance, the size of the workcell can also be compacted to minimium.

1. Introduction

In many manufacturing factories, intelligent robotic workcell which mainly consists of a robot and a camera station is often used as an dominant part of the automation activities to transfer parts from one station to another via a conveyor belt medium. The robot is used to perform operations on an industrial workpiece while the workpiece is moving on the conveyor.

The allowable productive time for the robot to operate on the workpiece seems to be the most critical factor affecting the production flexibility. Because of this, different alternate schemes approaching to the workpiece should be evaluated in order to increase the productive time within the envelop. Three schemes have been studied as the followings :

- Robot stays at a pre-determined 'home' position. It is activated to intercept with the workpiece after the image analysis has been finished. Then, it is synchronised with the workpiece until it reaches the downstream limit.

- Robot stays at a pre-determined 'home' position. It is activated to a given upstream position while the image analysis is being taken place. After the image analysis has been finished, the robot is activated to trace the moving workpiece and begins the synchronisation until it reaches the downstream limit.

- Robot stays at the boundary of the working envelop until the workpiece arrives. While the workpiece enters the working envelop as well as the image analysis has been finished, the robot is activated to trace the moving workpiece and begins the synchronisation until it reaches the downtream limit.

Each of the schemes has its own disadvantages and advantages which would be discussed in the section 3. For a particular control scheme, it is important that the workcell should achieves the following criteria :

- eliminating the possible idling time
- maximise the productive time within the envelop

In order to achieve the above criteria, the elements of the workcell must be well-coordinated. For an intelligent robotic workcell, one of the prominent factors to satisfy the above criteria is the choice of the distance between the robot and the camera station. For example, an over-sized workcell (i.e. one with an over-estimated distance) would ensure that the productive time on the workpiece within the envelop can be maximised. However, the idling time of the robot is increased because the robot has to wait for the workpiece to arrive at the boundary of the working envelop. Also, over-sized workcell tends to reduce the number of the workcells that could be installed in a fixed industrial space. On the contrary, if the distance is under-estimated (i.e. in the case of an under-sized workcell), a part of the working envelop can never been utilised. In this case, the flexibility of the application of the workcell would be decreased. Under such situation, a methodology is important for determining the appropriate distance between the robot and the camera station in order to achieve the above criteria.

Some researchers [2, 3, 4] have also developed a similiar workcell, however, this paper describes more deeply the various candidates for a control scheme and a methodology for the suitable definition on the distance between the robot and the camera station. A typical setup consisting of a PUMA robot is used as an example whereas this methodology is implemented.

2. Workcell Element

An experimental robotic workcell [1] installed in the Robotic Laboratory of the Concordia Centre for Industrial

Control shown in fig.1 consists of the following six modules :

- workpiece detector
- binary camera
- robot
- conveyor
- tracking circuitry
- microcomputer

The details of the individual elements is described in the appendix. As a summary, this workcell is described to recognise the arrival of a workpiece as well as its geometrical characteristics, with the objective to instruct in real-time. The robot is to achieve synchronous rendezvous with the workpiece in minimium time lapse. The analytical basis to achieve this minimality in time lapse is designated as an appropriate control scheme.

It is an objective of the control system to achieve the rendezvous between the robot and the workpiece at a point as close to the upstream limit of the working envelop of the robot as possible.

3. Overview of Control Schemes

Three control schemes have been evaluated and the schematic representation of the control schemes is generically shown in fig.2. However, three variations are shown in fig.4, 5 ,6 and the details are described as follows :

3.1 Scheme 1

As shown in fig.2, this control scheme consists of three stages – intercepting stage (A to B), approaching stage (B to C) and the production stage (C to D).

Intercepting stage :

An industrial workpiece is transferred from the upstream of the cell via a conveyor. The robot stays at a pre-determined 'home' position after finishing the previous cycle. Statistically, the homing point is located approximately at the mid-stream of the working envelop. When the workpiece arrives at the workpiece detector, the overhead camera would be activated to get the image of the workpiece. While the workpiece is still moving with the conveyor belt, the microcomputer analyses the image to extract the identity, centroid and orientation of the image. It is obvious that depending on the size of the workpiece, one or more frames have to be gathered and cacatenated to form a complete image. The number of frames has an effect on the image processing time, and hence the different timings have to be taken into consideration in different schemes. After finishing the image analysis, the microcomputer would send the geometrical data of the workpiece to the robot controller via the first serial line. Following is to activate the Real-time Path Control to move the robot intercepting with the workpiece. The time required for the robot moving from the home position to the intercepting location is called the intercepting time t_m.

The robot motion and the timings of this stage can be described by the following equations :

$$V_{r1}^2 = V_{x1}^2 + V_{y1}^2 + V_{z1}^2 \qquad (1)$$

The time t_m required for the robot to intercept the workpiece can be calculated separately by considering velocity components in the y - direction and then in the z - direction

$$t_m = \frac{Y_o - y_m}{V_{y1}} \qquad (2)$$

$$t_m = \frac{Z_o - z_m}{V_{z1}} \qquad (3)$$

The workpiece is intercepted by the robot after a distance d_m

$$d_m = V_c\,(T_{p1}N_f + t_m) \qquad (4)$$

From this, the home position of the robot can be related to d_m as :

$$D = V_{x1}t_m + d_m \qquad (5)$$

The time interval between the robot intercepting the workpiece, and reaching its downstream limit is

$$t_{ay} = \frac{L_1 + L_o - d_m}{V_c} \qquad (6)$$

Solving equations (1) to (6),

$$t_m = \frac{-\dfrac{V_c(D - V_c t_{im})}{V_{r1}^2 - V_c^2} + \sqrt{\left[\dfrac{V_c(D - V_c t_{im})}{V_{r1}^2 - V_c^2}\right]^2 - \dfrac{(D - V_c t_{im})^2 + Y_{om}^2 + Z_{om}^2}{V_{r1}^2 - V_c^2}}}{} \qquad (7)$$

$$d_m = V_c\,(t_{im} + t_m) \qquad (8)$$

let

$$\mu = \frac{V_{r1}}{V_c}$$

$$\tau_m = \frac{t_m}{D/V_c}$$

$$\tau_{im} = \frac{t_{im}}{D/V_c}$$

$$\tau_{y_{om}} = \frac{Y_o - y_m}{D} = \tau_{y_o} - \tau_{y_m}$$

$$\tau_{z_{om}} = \frac{Z_o - z_m}{D} = \tau_{z_o} - \tau_{y_m}$$

equations (7) and (8) become,

for $\mu > 1$

$$\tau_m = \frac{-(1 - \tau_{im})}{\mu^2 - 1} +$$

$$\sqrt{\left[\frac{1 - r_{im}}{\mu^2 - 1}\right]^2 - \frac{(1 - r_{im})^2 + r_{y_{om}}^2 + r_{s_{om}}^2}{\mu^2 - 1}} \qquad (9)$$

for $\mu < 1$

$$r_m = \frac{-(1 - r_{im})}{\mu^2 - 1} - \sqrt{\left[\frac{1 - r_{im}}{\mu^2 - 1}\right]^2 - \frac{(1 - r_{im})^2 + r_{y_{om}}^2 + r_{s_{om}}^2}{\mu^2 - 1}} \qquad (10)$$

for $\mu = 1$

$$r_m = \frac{(1 - r_{im})^2 + r_{y_{om}}^2 + r_{s_{om}}^2}{2\,(1 - r_{im})} \qquad (11)$$

Similarly, equation (8) can be normalised as follow :

$$d_m = V_c\,(t_{im} + t_m)$$

$$\frac{d_m}{D} = \frac{V_c\,(t_{im} + t_m)}{D} = \frac{t_{im} + t_m}{D/V_c}$$

$$r_m = \frac{d_m}{D} = r_{im} + r_m \qquad (12)$$

Using equations (9), (10) and (11), r_m is plotted vs μ as shown in fig.3. It is seen that r_m is a decreasing function of μ. Therefore, in order to reduce the arrival time, μ should be as large as possible, i.e. higher robot speed.

Approaching stage :

After intercepting the workpiece, the robot descends from this intercepting location to the workpiece level. At this stage, the robot is required to synchronise with the workpiece while the latter is moving on the conveyor. The time required for the robot to approach the moving workpiece from the intercepting location is called the approaching time t_{ap}, while the arrival time t_{arr} is defined as the time required for the robot travelling from the home position to the arrival position (i.e. $t_{arr} = t_m + t_{ap}$).

The robot motion and the timings of this stage may be described by the following equations :

$$v_{s2} = V_c \qquad (13)$$

$$v_{y2} = 0 \qquad (14)$$

$$v_{s2} = \text{constant}$$

$$t_{ap} = s_m / v_{s2} \qquad (15)$$

$$t_{arr} = t_m + t_{ap} \qquad (16)$$

normalising (15) and (16),

$$r_{ap} = \frac{t_{ap}}{D/V_c} = \frac{s_m/v_{s2}}{D/V_c} \qquad (17)$$

$$r_{arr} = r_m + r_{ap} \qquad (18)$$

Production stage :

At this stage, the robot stays at the workpiece level and perform its productive operation on the workpiece.

$$v_{s3} = V_c \qquad (19)$$

$$v_{y3} = 0 \qquad (20)$$

$$v_{s3} = 0 \qquad (21)$$

The normalised timings of different stages in this scheme are shown in fig.4 and a workpiece requiring typically three frames and μ=2 is used as illustration.

The advantage of this scheme is that it is relatively easy to compute the location of the workpiece since the centroid and orientation are already known before the robot is activated. The shortcoming is that the robot can only be activated after the image analysis has been finished. Thus, extra time is spent for the interception, resulting in reduction of the productive time of the robot.

3.2 Scheme 2

The second scheme is basically similar to the first scheme except that the robot can be activated before the image analysis is finished in order to reduce the time to achieve interception. As soon as the workpiece arrives at the workpiece detector, the robot is commanded to move to a given upstream waiting position. Typically, a value of d/D = 1/3 may be used as shown in fig.5. At this position, the robot waits until the image analysis is finished. Then, it is activated to intercept the workpiece.

The advantage of this scheme is to save the time for the robot moving from the home position to intercept with the workpiece, since the robot is driven to a given upstream position before the image analysis is finished. However, it is only beneficial under the situation that the image analysis is finished while the workpiece is not yet passed the waiting position. Otherwise, more time would be spent to catch up the workpiece and eventually reduces the productive time as well. In fig.5, the first cycle has more productive time since the robot almost starts to synchronise with the workpiece at the sleeping position. However, with a 4-framed or 5-framed workpiece which takes longer image analysis time, it may eventually reduce the productive time.

3.3 Scheme 3

In the third scheme, the upstream limiting position of the working envelop of the robot is taken as the waiting position. Typically, it may be arranged for d/D=0.02 from the camera as shown in fig.6. In operation, the robot moves to this waiting position ever before the workpiece arrives at the workpiece detector. After the image analysis is finished, the robot is driven to intercept the workpiece.

The advantage of this scheme is also to save the time for the robot moving from the home position to intercept with the workpiece. Similar to the second scheme, it is only beneficial under the situation that the image analysis

is finished while the workpiece is not yet passed the upstream limit. Otherwise, more time would be spent to catch up the workpiece and eventually reduces the productive time. Comparing to the second scheme, this scheme is worse because it takes extra time for the robot to go back to the upstream limit for every cycle. Thus, the cycle time is longer than the second scheme.

3.4 Control Scheme Summary

In summary, scheme 2 and 3 are beneficial while the robot is near the upstream limit of the working envelop at the time the image analysis of the workpiece is finished. Otherwise, extra time is spent for the robot to catch up with the workpiece. The control algorithm for these schemes are not deterministic as in the first scheme. Therefore, scheme 1 is chosen for the implementation and control strategy.

4. Architecture of the Workcell

Corresponding to the choices of the control scheme for synchronisation, an appropriate distance between the robot and the camera station can be determined according to the following criteria :

- eliminating possible idling time
- maximise the productive time within the envelop

In order to achieve the above criteria, the normalised timings of an operation should be analysed by the following equations at different values of d_m/D :

Referring to fig.2, maximium allowable productive time is

$$t_w = t_{sy} - t_{ap} = \frac{L_1 + L_e - d_m}{V_c} - \frac{z_m}{v_{s2}}$$

$$for \quad d_m > L_1$$

t_w can be normalised to

$$r_w = \frac{L_1 - L_e - d_m}{V_c} \times \frac{1}{D/V_c} - \frac{z_m}{v_{s2}} \times \frac{1}{D/V_c}$$

$$= \frac{L_1 - L_e - d_m}{D} - \frac{z_m/D}{v_{s2}/V_c}$$

$$= (r_1 + r_e - r_m) - (r_{zm}/\mu_{s2}) \quad for \; r_m > r_1 \qquad (22)$$

r_w has a limiting value as

$$r_w = r_e - \frac{r_{zm}}{\mu_{s2}} \quad for \; r_m < r_1$$

This value is also the maximium attainable for r_w. Consequently, if one designs the workcell to be such that $r_1 = r_m$ is evaluated as function of the starting position of the robot (X_o, Y_o, Z_o), the velocity ratio (μ), the image analysis time (t_{im}) and the required intercepting height (z_m) and the lateral position (y_m) (see equations (12) and (10)), then the workcell will provide maximium productive time (it follows that productive distance is also maximium). This situation constitutes an optimal cell size, with zero idling time.

If, however, r_m is designed to be smaller than r_1, then one results in a so called oversized cell. The productive time is at its maximium value as in the case of the optimal cell. However, an idling time r_{id} appears, such that

$$t_{id} = \frac{L_1 - d_m}{V_c}$$

normalised idling time

$$r_{id} = \frac{L_1 - d_m}{V_c} \times \frac{1}{D/V_c} = \frac{L_1 - d_m}{D}$$

$$r_{id} = r_1 - r_m \quad for \; r_1 > r_m \qquad (25)$$

Conversely, an undersized cell would result in less time for productive operation.

5. Example

The illustrated workcell is taken as an example under the situation that the robot is too close to the camera station.

upstream limit	L_1	= 170 mm
downstream limit	L_1+L_e	= 1400 mm
distance from effector home position to camera	D	= 1000 mm
image analysis time (for a 1- framed workpiece)	t_{im}	= 0.43 sec.
robot velocity in intercepting stage	V_{s1}	= 600 mm/sec
conveyor velocity (assume it is fixed)	V_c	= 300 mm/sec
distance between Y_o and y_m	y_{om}	= 135 mm
robot z - velocity in intercepting stage	v_{s1}	= 0

If the above system data are substituted in the equations (7), (4) and (8), For a one-framed workpiece, it takes about 0.91 sec to intercept with the workpiece, the intercept location d_m is 425 mm from the camera station and the maximium allowable synchronisation time is 3.25 sec.. If the robot is moved downstream further by 255 mm (i.e. 425 - 170) such that the ratio of r_1 is equal to r_m. By using the same computation, the maximium allowable time of synchronisation with is increased to 4.28 sec. which has 30 % more compared to the original value. This is especially useful for an operation which required longer time to perform.

6. Conclusion

This paper describes three possible control schemes in the synchronisation control of a robotic workcell. The first mentioned scheme is chosen as the control strategy because of its elegant control algorithm. Also, its deterministic properties makes the system capable of operating on more than one workpiece without much affecting its productive time. After selecting the control scheme, a methodology was introduced to determine the distance between the robot and the camera in order to eliminate the idling time and maximise the productive time. The concept introduced here may not be very beneficial for the pick and place operation. However, it is much useful for other synchronisation operation. By considering such concept in the workcell architecture, there'll be more time for the robot to perform operation on the industrial workpiece within the working envelop.

References

1. Cheng, R. and Mentor T., " Synchronisation Control of an Industrial Robotic Manipulator Using Camera Vision", Proc. of Intl. Conf. on Intelligent Auton. System, Netherland, 1986.

2. Dunne, J.M., " A Robot for Conveyor Loading / Unloading", Proc. of 10th ISIR, Italy, 1980.

3. Mo, X.J., and Liu, L.Z., " A Robot System with Vision, Touch and Slide Senses for the Grip onto a Moving Conveyor Belt", Proc. of 15th ISIR, Japan, 1985.

4. Ward, M.R., Rheame, D.P., Holland, S.W., Dunseth, J.h., Production PlantConsight Installations, General Motors Research Laboratories, Warren, Michigan, internal report no. GMR-4156, 1982.

NOMENCLATURES

V_{ri}	robot velocity in the intercept phase (mm/sec)
v_{i1}	i-component of robot velocity in the intercept stage
v_{i2}	i-component of robot velocity in the synchronisation stage
v_{i3}	i-component of velocity in the production stage
V_c	conveyor velocity (mm/sec)
μ	normalised robot velocity
μ_{d2}	normalised robot descending velocity in approach stage
X_o, Y_o, Z_o	home position of the robot
x_m, y_m, z_m	intercept position of the robot with the workpiece
y_{om}	difference between the y_o and the y_m
z_{om}	difference between the z_o and the y_m
D	distance between the home position and the camera
d_m	intercept distance measured from the camera
d/D	normalised distance
L_1	upstream limit of the robot working area
L_2	span of the robot working area
$r_{y_{om}}$	normalised distance of y_{om}
$r_{z_{om}}$	normalised distance of z_{om}
r_m	normalised distance of d_m
r_1	normalised distance of L_1
r_2	normalised distance of L_2
N_f	number of frame of the workpiece
T_{pi}	time required to process one frame of image
t_{ap}	time required to approach to the workpiece from the height z_m to the workpiece level
t_{id}	idling time
t_{im}	time required to analyse the image
t_m	time required to intercept with the workpiece
t_{arr}	$t_m - t_{ap} - t_{id}$
t_{sy}	maximum allowable time for synchronisation
t_w	maximum allowable productive time
τ_{ap}	normalised approach time
τ_{arr}	normalised arrival time
τ_{id}	normalised idling time
τ_{im}	normalised image analysis time
τ_m	normalised intercept time
τ_{sy}	normalised synchronisation time
τ_w	normalised productive time

APPENDIX : WORKCELL ELEMENTS

a. workpiece detector

The workpiece detector consists of a pair of LEDs installing at either sides of the conveyor belt to generate an interrupt signal to activate the overhead camera for image mapping.

b. camera

A binary camera with resolution 128x64 pixels/frame is installed at upstream of the conveyor for picture mapping. An image of a workpiece is composed of constant-sized number of frames which are taken while the workpiece is moving on the conveyor. The image would be analysed by the microcomputer to extract the identity, centroid and the orientation of the workpiece.

c. robot

Situated downstream of the conveyor is a PUMA 560 robot. The robot is linked with the microcomputer via two RS-232C serial lines. The first line with 9600 Baud rate is to inform the robot when the image analysis has been finished so that the Real Time Path Control can be activated. The second line with 19200 Baud rate is to provide communication between the microcomputer and the robot through the ALTER port of the robot controller in order to achieve the Real-Time Path Control purpose. Data for generating the motion of the robot would be sent to the alter port of the robot controller from the microcomputer every 28.8 ms.

d. conveyor

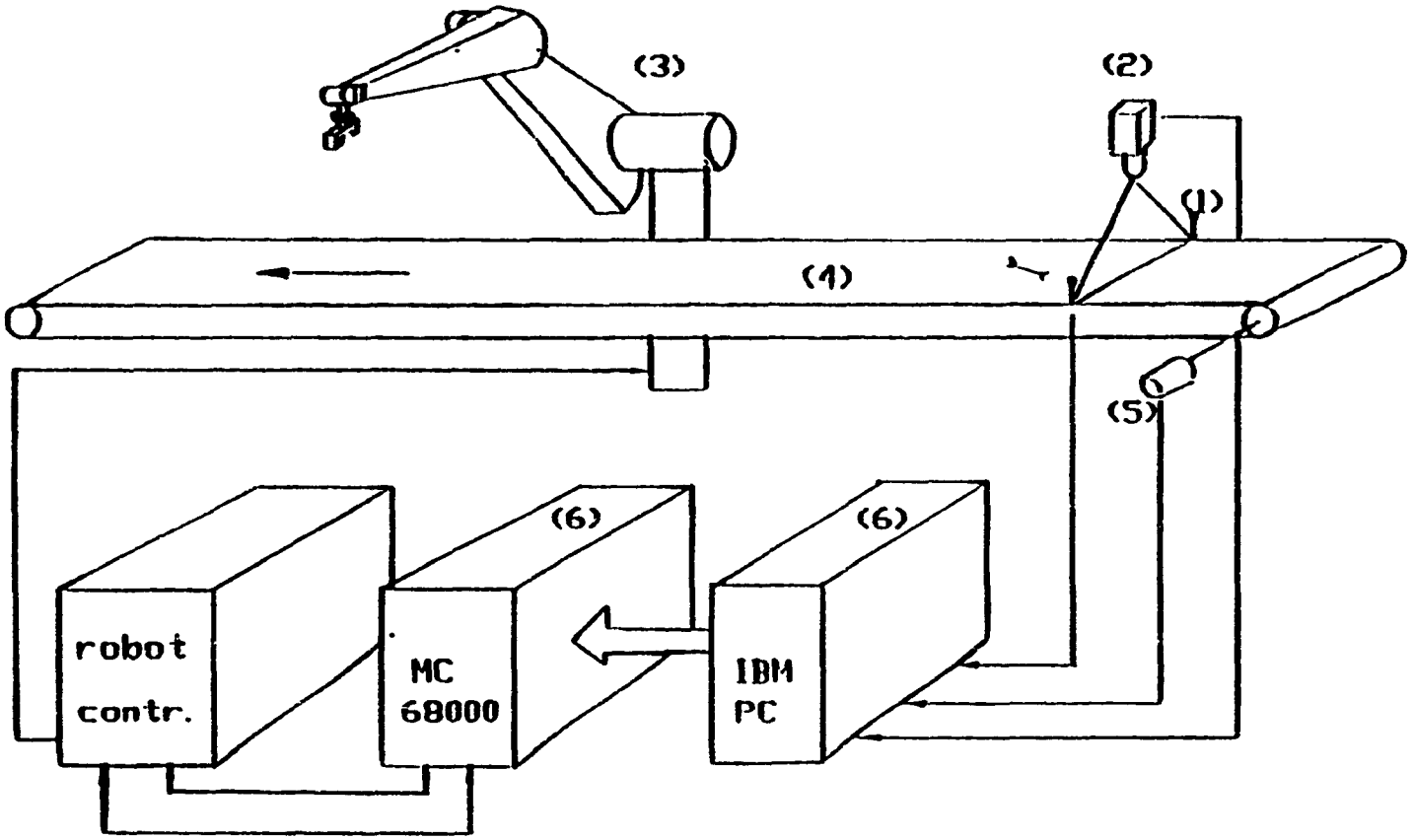
A conveyor provides a transferring action of the workpiece from the upstream to the downstream. It is driven by a DC motor with maximum velocity 400 mm/sec..

e. tracking circuit

A tracking circuit with an optical belt encoder is used to keep track the belt velocity and the x-displacement of the workpiece so that the change of the workpiece position can be incorporated in the real-time computation of the robot motion.

f. microcomputer

The microcomputer consists of two mother boards and runs in two different operating systems. One of them is an IBM PC running at 4.77 MHz with a 8088 CPU with a 8087 floating point math processor which is mainly responsible for the image analysis. Another mother board running under OS9 operating system with a MC 68000 CPU at 10 MHz which is mainly used to generate the motion commands to move the robot in the RTPC mode.



(1) workpiece detector (3) robot manipulator (5) belt encorder

(2) binary camera (4) conveyor (6) microcomputer

Fig.1 Workcell Schematic

AB -- intercept stage
BC -- approaching stage
CD -- productive stage

A -- home position (Xo, Yo, Zo)
B -- intercept position (xm, ym, zm)
C -- arrival position

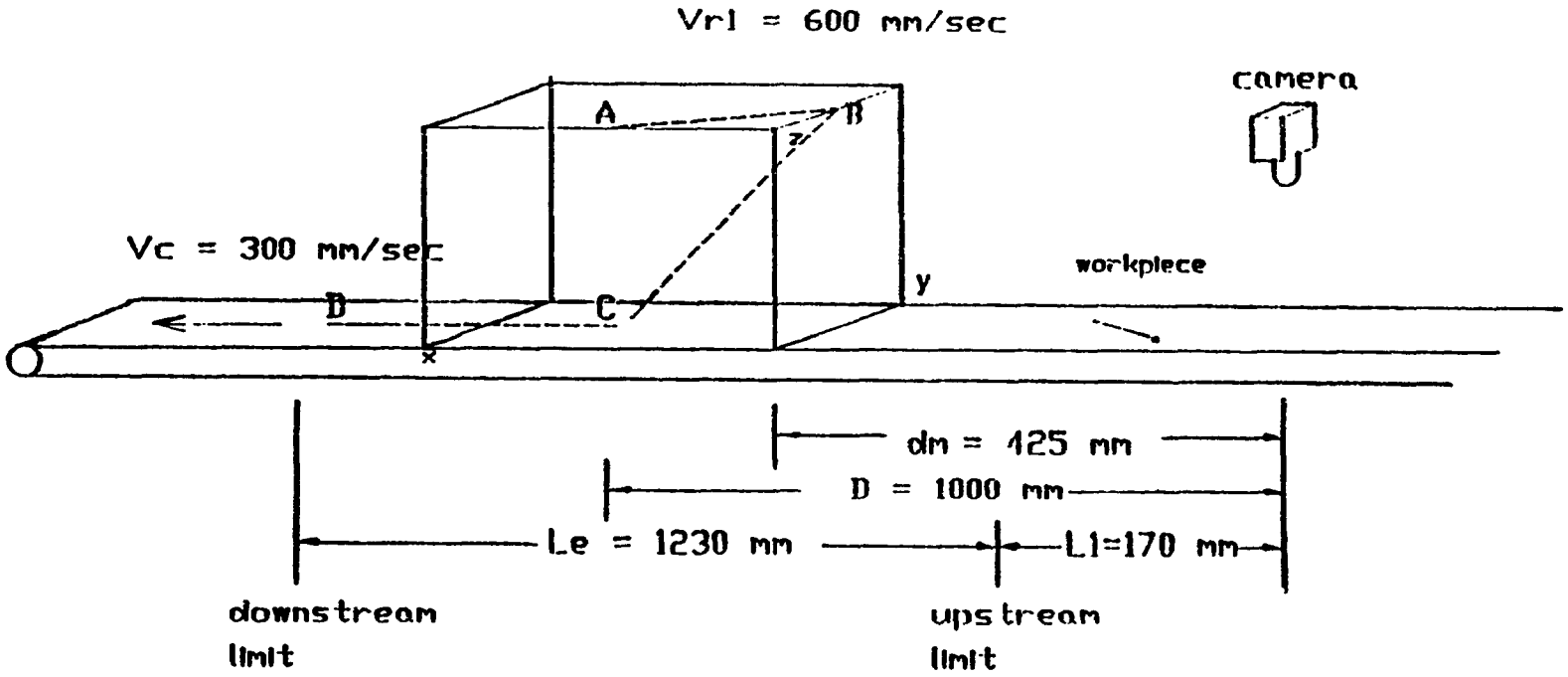


Fig.2 Schematic of the Control Scheme

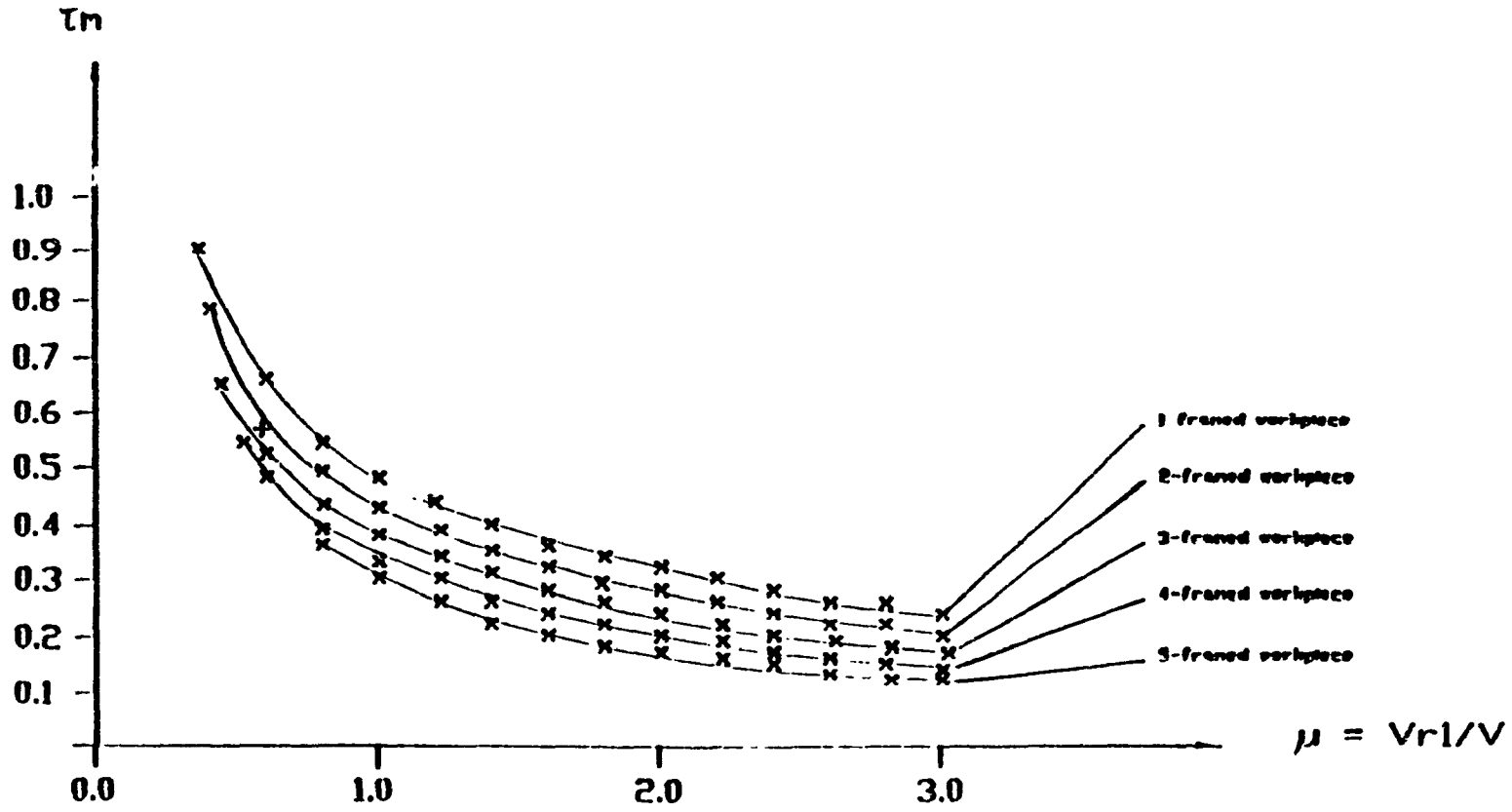


Fig.3 Relationship of μ and τ_m

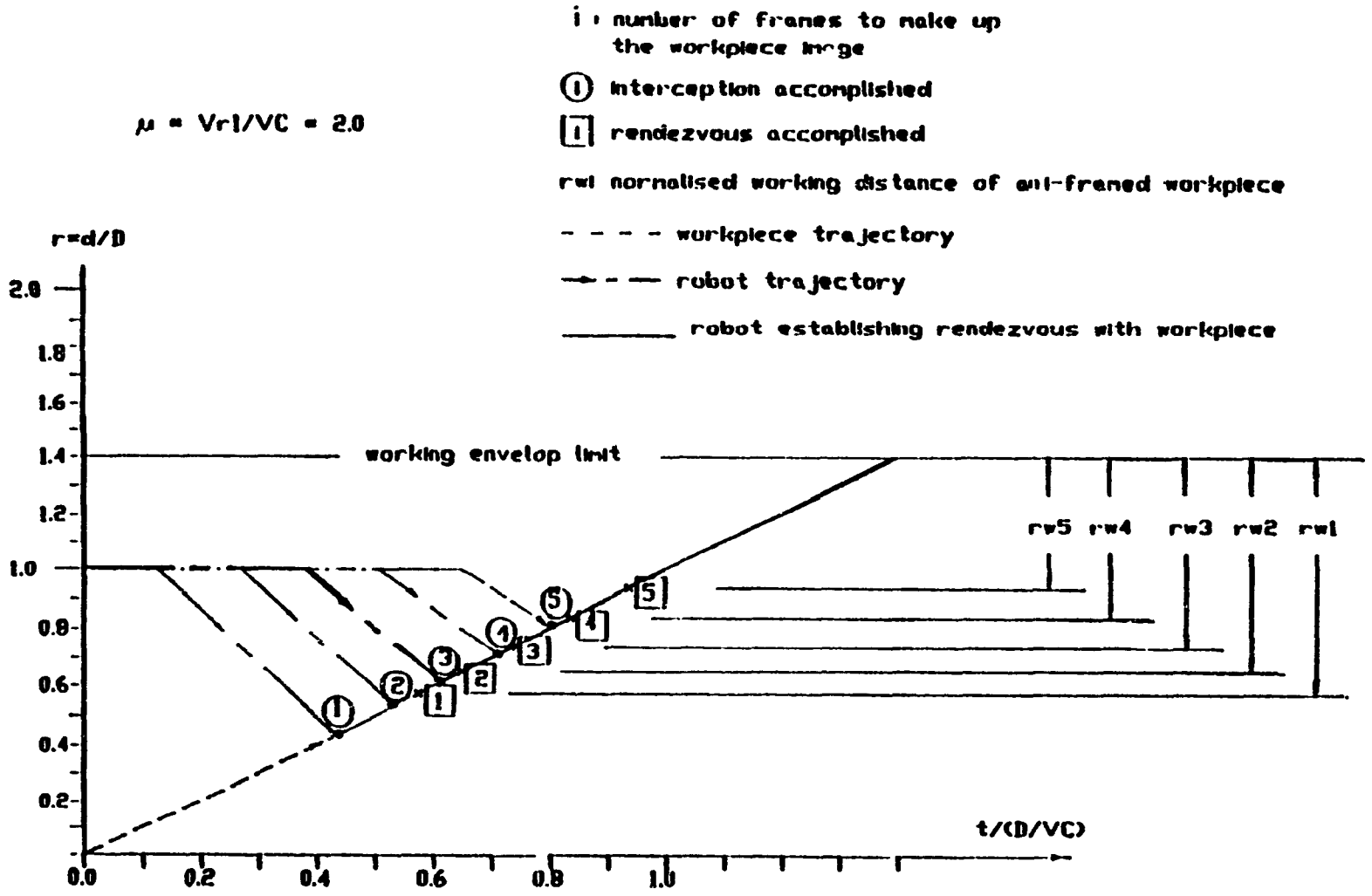


Fig.4 Control Scheme 1

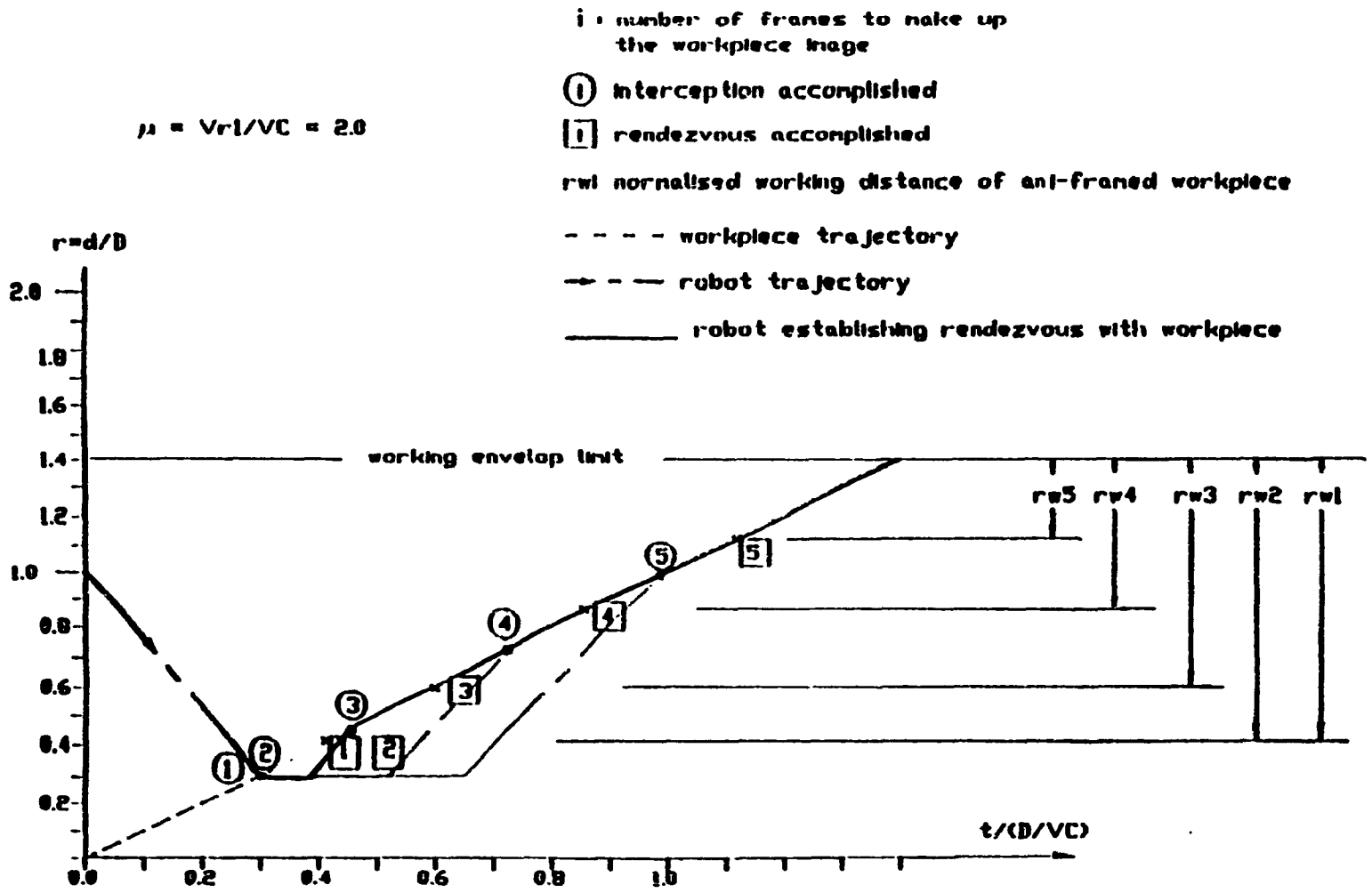


Fig.5 Control Scheme 2

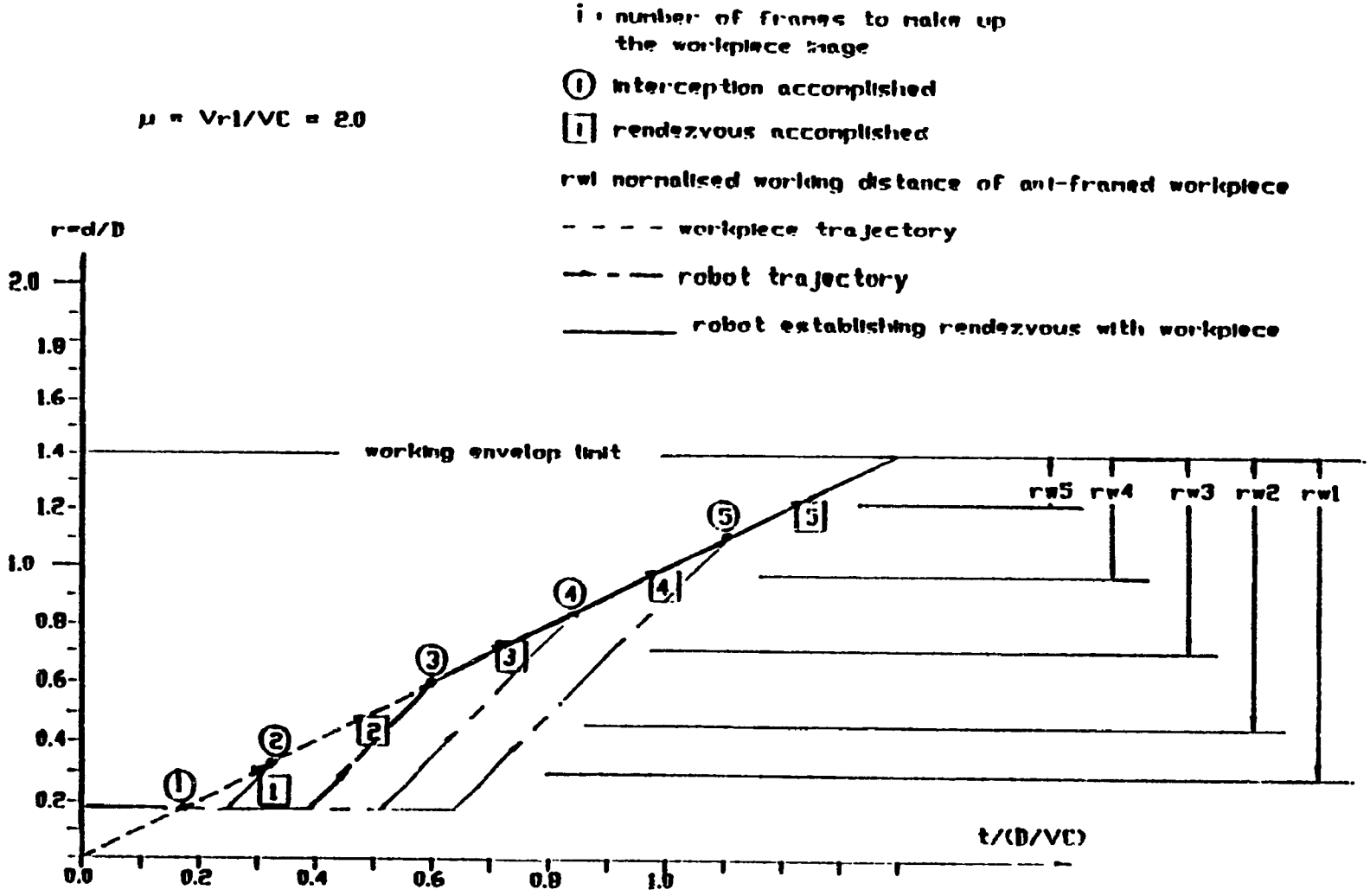


Fig.6 Control Scheme 3

APPENDIX K

Auxiliary Camera Transeceiver/Strobe Light Circuit

K.1 INTRODUCTION

A MicronEye camera is used as an wrist-mounted auxiliary camera to implement the incremental image processing. One of the difficulties is that the provided cable can only transmit to a 5-ft distance without any interference. Therefore, a transeceiver circuit is designed to provide buffering for an extended cable (12 ft).

The binary camera requires lighting shining on the workpiece to provide the contrast. For a wrist-mounted camera, it is not realistic to mount large flood lights on the robot arm. Therefore, the strategy is to use a small strobe light as the light source. While the auxiliary camera is triggered to take a frame for the incremental image, the strobe light will be activated. After the strobe light have finished, it will be charged up again for the next flash.

K.2 CIRCUIT DESCRIPTIONS

The circuit consists of three modules : 1) transmitter module; 2) receiver module; 3) flash control module

K.2.1 Transmitter Module

The transmitter circuit(fig.K.1) consists of four AM26LS31 quad differential line drivers, one AM26LS32 quad differential line receiver, and one 74LS04 hex inverter. The signals sending to the camera are through the line drivers. The signal for the camera's SOAK command is inverted and then sent to the line drivers (this is used to flash the strobe light). A 120 Ohm resistor is used to terminate the data signals. Ground and +5 Volts are sent to the receiver. The only

connection not made is the IS32 optic RAM threshold level, which is generated on the receiver board. All signals and power are sent via a fourty conductor cable with male 37 pin D-connectors ateach end. Decoupling capacitors (0.1 uF) are connected between power and ground of each chip, and a 10 uF capacitor is connected across the power supply input for the board.

K.2.2 Receiver Module

The receiver circuit(fig.K.2) consists of four AM26LS32 quad differential line receivers and one AM26LS31 quad line driver. The input signals and power are recieved on a female 37 pin D-connector. All signals except the flash signal are sent directly to the camera. The flash signal is connected to a female phone jack on the exterior of the case which houses the receiver circuitry. The threshold level for the camera is set by a resistor and a potentiometer to 2.5 Volts. The module is moumted to the underside of the Puma robot.

K.2.3 Flash Control Module

The light source is a modified strobe light and it is controlled by a flash conrol module. The flash control module(fig.K.3) is powered by a 1.5 Volt Alkaline battery and is housed in a small box which is attached to the underside of the receiver module. A female phone jack on the exterior of the module is used to receive the flash signal from the receiver. A small two wire cable with male phone jacks at either end connects the receiver and the flash control modules. The other part consists of an MOC 3011 optically isolated triac which is used to gate a C106D SCR to trigger the flash. This design permits

safe control of the high voltages in the flash control circuitry. A connector on the flash control module connects to the strobe light itself, which is mounted on the robot wrist. However, the 1.5 Volt battery arrangement in the flash control module proved to be inadequate for practical purposes. The time required to charge the strobe light was to long (about 20 seconds), and the battery would have to be changed often (almost once a day during test procedures). The solution to this problem was to use two Varta P125 batteries in parallel, and an LM317 voltage regulator, to power the flash control module. The voltage of the batteries is 6 volts, so the regulator is adjusted via R2 for a 1.5 volt output when the flash is fully charged. The new chargetime is about 8 seconds.

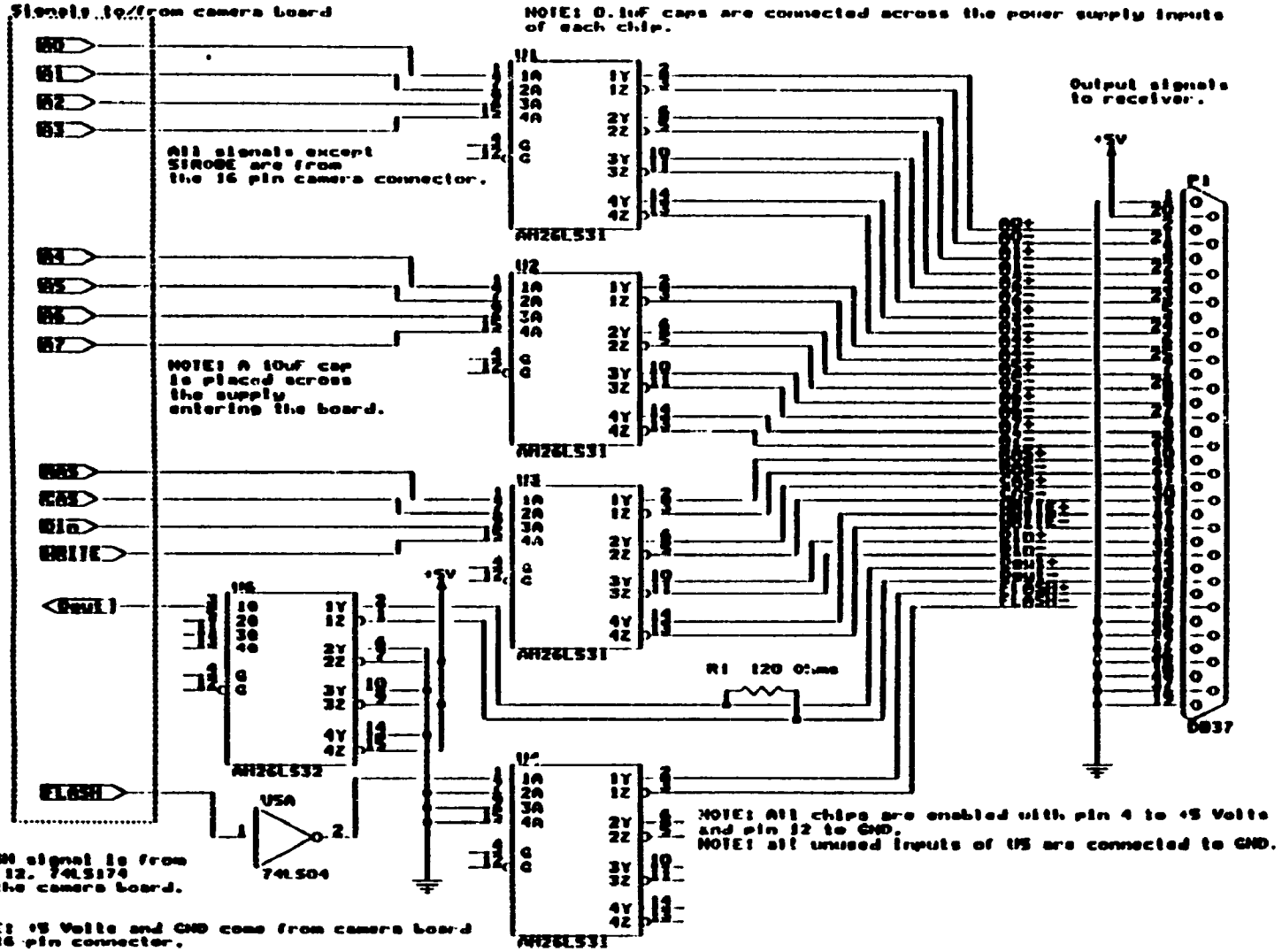


Fig.K.1 Transmitter Circuit for the Camera Buffer

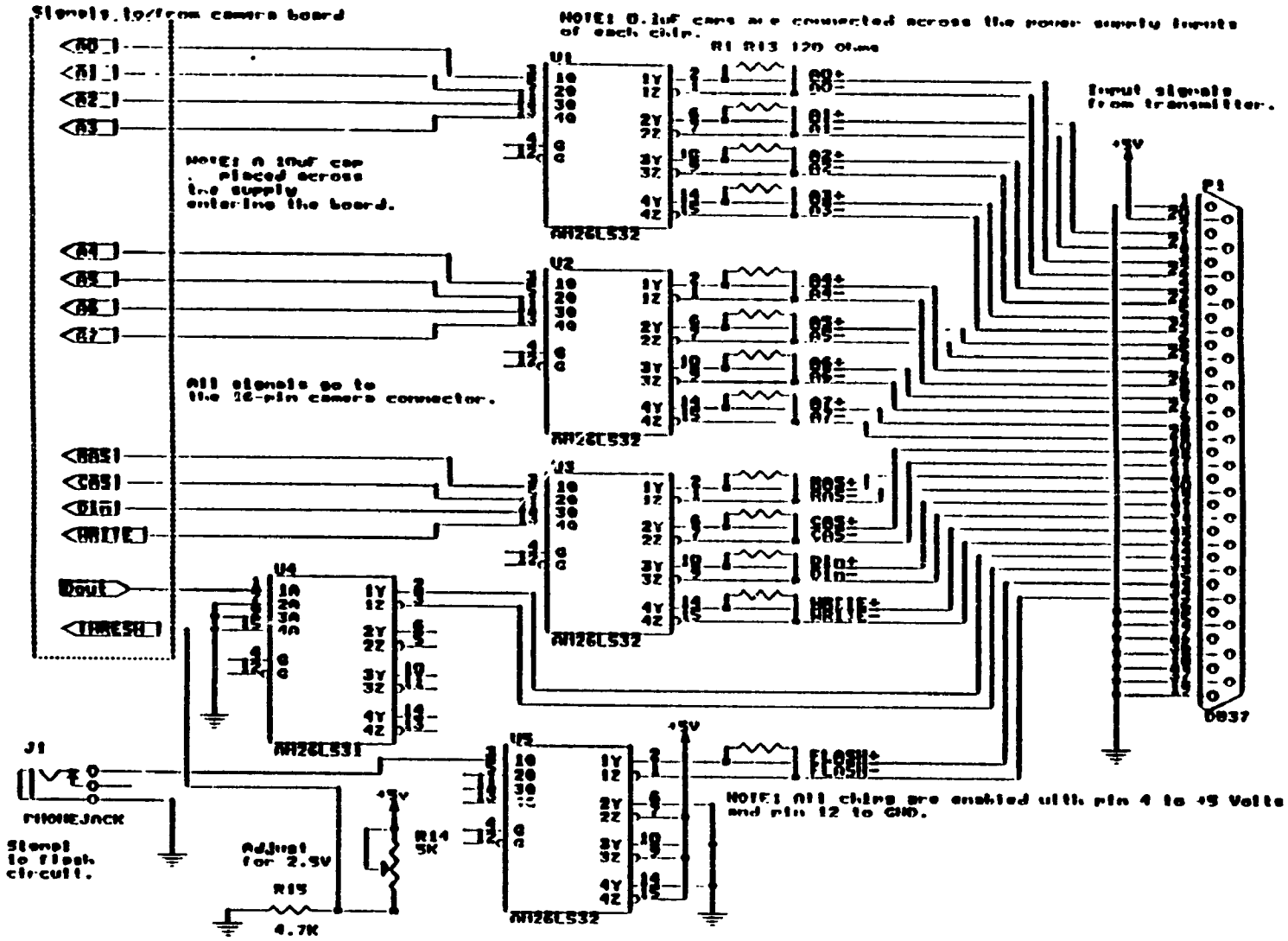


Fig.K.2 Receiver Circuit for the Camera Buffer

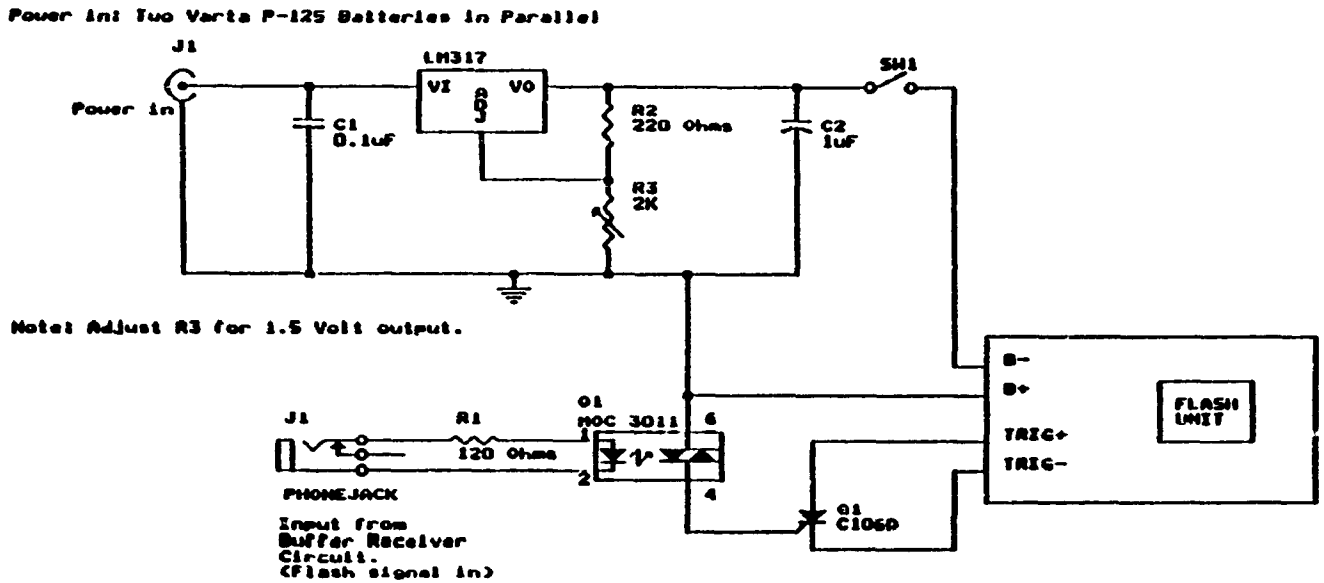


Fig.K.3 Flash Control Circuit