

# 遠端操作系統之網路延遲訊號估測與影像輔助機制設計

研究生：廖子期

指導教授：徐保羅 博士

國立交通大學電控工程研究所

## 摘要

本論文建構了利用網路攝影機監控的遠端循軌操作系統，發展網路系統控制架構與操作輔助機制，成功降低網路延遲對於系統響應與人為操作造成的循軌誤差，並有效地降低任務完成時間。

網路延遲對系統響應造成的影響可分為兩部分：底層網路系統控制響應與造成上層操作時之困難。針對底層系統響應之改善，模型預測控制器(model predictive controller, MPC)能及時地計算出最符合預期響應之控制訊號，且比一般控制器更能克服複雜且不定因素對系統造成的影響；而通訊擾動估測器(communication disturbance observer, CDOB)不需要延遲時間模型，將網路延遲視為擾動量進行估測後做為回授補償值，本論文並且提出一 CDOB 改善架構，消除在取樣頻率不足時發生之穩態誤差，最後利用權重分配因子將 CDOB 與 MPC 估測之整合迴授訊號最佳化，實驗證明在平均 450 ms 之延遲時間下能夠大幅降低響應震盪幅度 84.5% 與縮短收斂時間 87.6%。

最後，針對網路延遲效應降低系統上層操作透明度進而造成循軌誤差，設計操作輔助機制，改善操作困難。首先，利用不同工作空間座標下之相對位置命令映射，降低循軌誤差並縮短完成時間縮短。但由於在影像定位誤差時需另有校正機制，所以考慮人因與網路延遲效應，提出操作點預測輔助機制，克服影像定位對輔助機制之影響，並在需要進行微調時切換至 perfect delay compensation (PDC) 控制架構，經實驗結果證明可有效降低循軌誤差並縮短完成時間。

關鍵字：遠端操作系統、通訊擾動估測器、模型預測控制器、操作輔助機制

# **Estimation of Network Delayed Signals and Development of Assist Strategies in Teleoperation**

Student : Tzu-Chi Liao

Advisor : Dr. Pau-Lo Hsu

Department of Electrical and Control Engineering  
National Chiao Tung University

## **ABSTRACT**

This thesis develops a control scheme composed by a model predictive controller (MPC), a communication disturbance observer (CDOB), and a feedback weighting mechanism in order to overcome the inevitable delay time in an Internet-based teleoperation system. The CDOB mainly compensates for the time-delay effect by treating it as a disturbance without the knowledge of the delay-time model. Moreover, this thesis proposes an improved scheme of CDOB in order to eliminate the steady-state error existed in a networked control system without a sufficient sampling rate. Meanwhile, the MPC adopted in this thesis predicts the best control signal in order to optimize the system performance. Generally, the time-delay effect usually decreases the transparency of the network controlled system and three assist strategies are presented in this thesis as: (1) virtual operation point, (2) prediction operation point, and (3) fine adjustment switching mechanism. Contouring motion experiments on a 2-DOF virtual manipulator validates the proposed assist mechanism.

Keywords : teleoperation, CDOB, MPC, assist strategy

## 誌 謝

首先我要感謝我的指導教授徐保羅老師這兩年來的指導，在研究上為循循善誘的訓練我思考、解決問題並找到正確的出路，特別感謝老師花了許多時間並且非常有耐心地與我討論我的研究內容，使我能順利地完成碩士學位。同時也要感謝口試委員楊古洋老師以及葉賜旭老師對於本論文的批評指正與寶貴的建議，使本論文能夠更加的完整。

在此亦要感謝在實驗室的日子裡陪伴在我身邊的學長、同學與學弟們：博士班賴建良學長、謝鎮洲學長與黃煒生學長平時給我的指導與幫助，徐煒智學長、許正忠學長、施子強學長、蕭任斌學長、張孟哲學長亦給我許多寶貴意見及解惑，政宏、釗甫、軒正、思翰、志瀚、信佑在學業上的討論與指教，並一起度過了許多實驗室的時光，不管是在研究上的討論或是生活上的經驗分享，都是我碩士班生活不可或缺的美好回憶。在此特別感謝賴建良學長，總是不厭其煩地關心著學弟們並且耐心地與我討論本論文的相關研究內容，在實作上的經驗也不遺餘力地傳承；政宏、軒正、釗甫、志瀚在研究與實驗上的大力幫忙，獻上誠摯的感謝。

最後要感謝我的父母、妹妹、我的女朋友顧冰清以及其他支持我的好朋友們，長久以來在生活上與課業研究上的陪伴支持與鼓勵，讓我能更加有信心與動力地去克服每一個難關，最後得以完成我的碩士學位，在此獻上最誠摯的感謝之情給每一位幫助過我的人，願將此榮耀與喜悅與他們一起分享。

## 目 錄

摘要 .....	I
ABSTRACT .....	II
目 錄 .....	IV
圖目錄 .....	VI
表目錄 .....	X
第一章 緒論 .....	1
1.1 研究動機與目的 .....	1
1.2 研究背景與發展概況 .....	2
1.3 問題陳述 .....	4
1.4 研究方法 .....	6
1.5 預期達成目標 .....	8
1.6 論文架構 .....	9
第二章 遠端操作系統 .....	10
2.1 相關網路協定與介面介紹 .....	10
2.1.1 實體與資料連結層：乙太網路(Ethernet) .....	10
2.1.2 網路與傳輸層：TCP/IP 通訊協定[11] .....	11
2.1.3 傳輸層到應用層之通訊介面：Windows Sockets .....	12
2.2 遠端操作系統設計 .....	14
2.2.1 開發環境 .....	14
2.2.2 操作介面與軟體架構 .....	15
2.2.3 遠近端傳送封包格式 .....	18
2.2.4 循軌誤差(Contouring error)定義 .....	19



第三章 通訊擾動估測器之實現與改進技術.....	22
3.1 通訊擾動估測器(COMMUNICATION DISTURBANCE OBSERVER, CDOB).....	22
3.2 網路擾動之概念.....	22
3.3 通訊擾動估測器之設計.....	24
3.4 CDOB 設計頻寬限制與系統取樣頻率不足時之估測誤差 .....	26
3.5 系統取樣頻率不足時之 CDOB 改良架構.....	31
3.6 實驗結果.....	34
3.7 CDOB 與 ADAPTIVE SMITH PREDICTOR 之比較 .....	37
第四章 模型預測控制器.....	40
4.1 模型預測控制器(MODEL PREDICTIVE CONTROLLER)基本概念 .....	40
4.2 架構實現方法.....	42
4.3 整合 MPC 與 CDOB 估測輸出之實驗結果 .....	48
4.4 MPC 與 PI 控制器於網路控制系統之比較.....	54
第五章 估測迴授訊號之權重分配機制.....	56
5.1 MPC + CDOB + (FEEDBACK WEIGHTING MECHANISM).....	56
5.2 各種控制架構之實驗結果比較.....	61
第六章 操作輔助策略.....	66
6.1 虛擬操作點影像輔助機制.....	66
6.2 操作點預測輔助機制.....	73
6.3 微調切換機制.....	76
6.4 實驗結果.....	78
6.4.1 虛擬操作點影像輔助機制： .....	80
6.4.2 操作點預測輔助機制.....	83
6.4.3 微調切換機制.....	84

6.5 小結.....	99
第七章 結論與未來發展.....	100
參考文獻.....	103

## 圖目錄

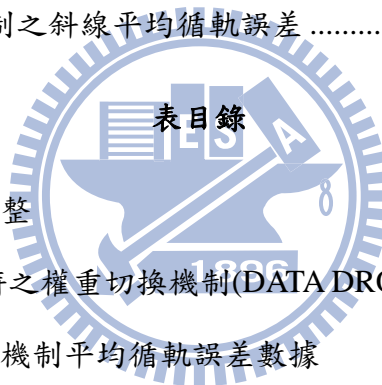
圖 1-1 閉迴路控制系統 .....	4
圖 1-2 加入網路的閉迴路控制系統 .....	4
圖 1-3 網路監控系統操作示意圖 .....	5
圖 1-4 虛擬操作點: 近端, OPEN LOOP 操作(無影像定位誤差).....	7
圖 1-5 操作點預測: 遠端影像迴授, CLOSED LOOP 操作(影像定位誤差) .....	8
圖 1-6 微調切換機制: 遠端影像迴授, CLOSED LOOP 操作(影像定位誤差) ....	8
圖 1-7 期望響應速度示意圖 .....	9
圖 2-1 TCP SOCKET 客戶端/伺服器 .....	13
圖 2-2 遠端操作系統實體照片 .....	15
圖 2-3 遠端網路系統控制架構 .....	15
圖 2-4 操作端(CLIENT)操作介面 .....	16
圖 2-5 受控端(SERVER)操作介面.....	17
圖 2-6 訊號名稱定義 .....	18
圖 2-7 操作端傳送封包 .....	18
圖 2-8 受控端傳送封包 .....	19
圖 2-9 目前線段之循軌未完成 .....	19
圖 2-10 前一線段循軌完成 .....	20
圖 2-11 方形直線循軌誤差定義 .....	20
圖 2-12 循軌狀態判斷 .....	20

圖 2-13 菱形斜線循軌誤差定義 .....	21
圖 3-1 網路控制系統示意圖 .....	22
圖 3-2 網路控制系統等效圖 .....	22
圖 3-3 將網路延遲視為擾動量之示意圖 .....	23
圖 3-4 網路擾動估測示意圖 .....	25
圖 3-5 CDOB 與網路擾動模型極點關係 .....	26
圖 3-6 CDOB 中網路延遲補償架構示意圖 .....	26
圖 3-7 CDOB 內部架構 .....	27
圖 3-8 離散系統中 $g_{net}/(S+g_{net})$ 之波德圖 ( $g_{net} = 15$ ) .....	28
圖 3-9 CDOB 估測值 .....	29
圖 3-10 兩等效控制系統架構 .....	29
圖 3-11 相同系統於不同取樣時間下之模擬結果 .....	29
圖 3-12 實現於離散系統之架構圖 .....	30
圖 3-13 CDOB 極點略小於網路擾動模型極點時之網路擾動估測值 .....	31
圖 3-14 CDOB 之改善後內部架構 .....	31
圖 3-15 改善架構於離散系統實現 .....	32
圖 3-16 CDOB 極點略小於網路擾動模型極點時之網路擾動估測值(簡化後架構) .....	32
圖 3-17 CDOB 回授補償架構 .....	33
圖 3-18 CDOB 原始架構之估測迴授 .....	33
圖 3-19 CDOB 改善架構之估測迴授 .....	34
圖 3-20 ROUND-TRIP DELAY ESTIMATION .....	34
圖 3-21 閉迴路網路控制系統架構圖 .....	35
圖 3-22 未加入網路之系統響應 .....	35
圖 3-23 加入網路後之系統響應 .....	35

圖 3-24 加入 CDOB 之原始架構 .....	36
圖 3-25 使用原始 CDOB 架構與改善後架構之系統響應比較 .....	36
圖 3-26 網路控制系統 UNIT GAIN 與加入 CDOB 後響應比較 .....	37
圖 3-27 SMITH PREDICTOR 架構 .....	38
圖 3-28 網路延遲時間變動劇烈時之 RTT 量測值 .....	38
圖 3-29 CDOB 與 SMITH PREDICTOR 於網路延遲變動劇烈時之響應比較 .....	39
圖 3-30 CDOB 與 ADAPTIVE SMITH PREDICTOR 於網路延遲變動劇烈時之響應比較 .....	39
圖 4-1 MPC 設計概念示意圖 .....	41
圖 4-2 離散系統中之 MPC 控制架構 .....	47
圖 4-3 MPC 控制訊號計算方法實現流程圖 .....	48
圖 4-4 不同 $rw$ 之模擬結果比較 .....	49
圖 4-5 未加入網路 .....	50
圖 4-6 加入網路，未使用估測器 .....	50
圖 4-7 網路延遲時間變動劇烈時之 RTT 量測值 .....	50
圖 4-8 分別使用 ADAPTIVE SMITH PREDICTOR 與 CDOB 做為系統估測器於延遲時間劇烈變動下之系統響應 .....	51
圖 4-9 MPC+CDOB 控制架構示意圖 .....	51
圖 4-10 不同 $rw$ 於網路控制系統之實驗結果比較 .....	52
圖 4-11 CDOC 與 MPC+CDOB 響應比較 .....	53
圖 4-12 未加入網路時之 MPC 與 PI 控制器響應比較 .....	54
圖 4-13 加入網路且參數不變時之系統響應比較 .....	55
圖 4-14 RISE TIME 相同時之響應比較 .....	55
圖 5-1 MPC + CDOB + (FEEDBACK WEIGHTING MECHANISM) 之控制架構 .....	56

圖 5-2 RTT 為 450 MS 下之各參數響應比較.....	58
圖 5-3 RTT 為 340 MS 下之各參數響應比較.....	58
圖 5-4 RTT 為 145 MS 下之各參數響應比較.....	59
圖 5-5 加入資料遺失時之權重切換機制(DATA DROPOUT RATE: 30.9%) .....	60
圖 5-6 加入權重切換機制之 SETTLING TIME 變化 .....	60
圖 5-7 各種控制架構之實驗結果比較 .....	62
圖 5-8 PDC 控制架構示意圖 .....	63
圖 5-9 MPC+CDOB+(FEEDBACK WEIGHTING MECHANISM) .....	64
圖 5-10 PDC 控制架構響應 .....	64
圖 6-1 虛擬操作典影響輔助機制架構圖 .....	67
圖 6-2 工作空間於視覺空間點投影示意圖 .....	67
圖 6-3 工作空間與視覺空間座標系統轉換示意圖 .....	69
圖 6-4 WEBCAM 與虛擬工作影像相對位置圖 .....	70
圖 6-5 由 WEBCAM 鏡頭朝工作區看之透視示意圖 .....	71
圖 6-6 WEBCAM 與工作區距離與成像焦距關係示意圖 .....	71
圖 6-7 虛擬操作點與實際操作點映射示意圖 .....	72
圖 6-8 操作點預測輔助機制架構圖 .....	73
圖 6-9 操作點預測之學習機制 .....	74
圖 6-10 預測操作點之概念示意圖 .....	75
圖 6-11 循軌誤差發生處 .....	77
圖 6-12 微調切換機制架構圖 .....	78
圖 6-13 未加入操作輔助機制之平均循軌誤差 .....	79
圖 6-14 加入操作點影像輔助機制之直線平均循軌誤差 .....	81
圖 6-15 影像擷取範圍確認誤差示意圖 .....	82
圖 6-16 影像定位誤差之操作端示意圖 .....	82

圖 6-17 影像定位誤差時之平均循軌誤差 .....	82
圖 6-18 加入操作點預測輔助機制之直線平均循軌誤差 .....	84
圖 6-19 加入微調切換機制之直線平均循軌誤差 .....	85
圖 6-20 長度短且轉折點多之循軌實驗 .....	87
圖 6-21 僅使用預測點輔助機制之平均循軌誤差 .....	88
圖 6-22 加入微調切換機制之平均循軌誤差 .....	89
圖 6-23 未加入操作輔助機制之斜線平均循軌誤差 .....	91
圖 6-24 加入操作點影像輔助機制之斜線平均循軌誤差 .....	93
圖 6-25 影像定位誤差下之循軌誤差 .....	94
圖 6-26 加入操作點預測輔助機制之斜線平均循軌誤差 .....	96
圖 6-27 加入微調切換機制之斜線平均循軌誤差 .....	98



表格 5-1 權重分配參數調整 .....	57
表格 5-2 加入資料遺失時之權重切換機制(DATA DROPOUT RATE: 30.9%) .....	60
表格 6-1 未加入操作輔助機制平均循軌誤差數據 .....	79
表格 6-2 加入操作點影像輔助直線平均循軌誤差數據 .....	81
表格 6-3 加入操作點預測機制直線平均循軌誤差數據 .....	84
表格 6-4 加入微調切換機制直線平均循軌誤差數據 .....	86
表格 6-5 僅使用預測點輔助機制之平均循軌誤差數據 .....	88
表格 6-6 加入微調切換機制之平均循軌誤差數據 .....	90
表格 6-7 未加入操作輔助機制斜線平均循軌誤差數據 .....	92
表格 6-8 加入操作點影像輔助機制於斜線平均循軌誤差數據 .....	93
表格 6-9 加入操作點預測機制斜線平均循軌誤差數據 .....	96
表格 6-10 加入微調切換機制斜線平均循軌誤差數據 .....	98
表格 6-11 操作點預測與微調切換機制於不同軌跡之效能比較 .....	99

# 第一章 緒論

## 1.1 研究動機與目的

早在遠古時代，遠端操作的概念已被應用於日常生活中，例如使用樹枝來探測未知的環境，測量其軟硬深淺，或是利用具有一定長度的工具讓人類遠離難以承受的環境並且完成工作。遠端操作的概念使人類的生活增加了延展性與提高了安全性。在這個網路發達的年代，遠端操作系統透過網路資料交換大大地延長了操作的距離，網路就像無限延長的媒介，讓人類可將雙手延伸到世界的任何一個角落。舉凡從軍事任務、太空計畫、救災行動、核能電廠的燃料棒更換、高危險化學物質封裝、深海探測，到遠端外科手術，都是遠端操作系統的極致應用。

雖然網際網路延長了遠端操作系統的範圍，卻也是遠端操作系統最需要克服的難題。對於一個遠端操作系統而言，最重要的就是穩定性(stability)，其次是透明性(transparency)或同步性(synchronization)，而這幾項特性往往因為網路的特性造成嚴重的不良影響。由於網路具有時間延遲、封包遺失與頻寬的考量，在這樣的限制之下，操作系統也會同樣受到限制，例如時間的延遲與封包遺失可能導致系統不穩定，頻寬的不足可能導致應用程式或控制器的效能受限而無法讓操作者能夠準確地操作或判斷當前情況。所以基於種種限制因素的考量，如何克服網際網對系統帶來的負面影響，一直是學者們努力的目標，也是本論文的重點。



## 1.2 研究背景與發展概況

當網際網路逐漸成為人們日常生活不可或缺的糧食時，科技也隨著網路的蓬勃發展迅速的進步。雖然人類在機器人領域已投注了不少時間與心力，但是在許多嚴苛或充滿不定因素的環境，例如危險的軍事武器拆卸、核能電廠的燃料棒更換、太空任務、深海任務、無人駕駛飛機與汽車…等等，機器人的自主性還是略顯不足，因此遠端操作的技術便因此誕生。

1940 年代晚期，由 R.Goertz 於 Argonne National Laboratory 提出第一套主從式遠端操作系統機構[1]，到了 1960 年代早期，Ferrell 與 Sheridan 等人作了一系列的實驗，探討網路延遲對於遠端操作系統的影響，他們提出的監督式控制系統(supervisory Control)刺激了許多人對於這個領域的研究興趣[2]，所以相關的研究也開始蓬勃發展，進而發展出遠端操作系統的軟體語言與視覺強化預測顯示機制，目的為將主從端資料交換量降低。直到 1980 年代晚期至 1990 年代早期，網路理論(network Theory)逐漸成熟，Anderson 與 Spong 於 1989 年提出的分散理論(scattering Theory)也在此時間世[3]。而關於透明度(transparency)的相關探討於 1990 年代初期由 Lawrence[4]、Yokokohiji 與 Yoshikawa 提出[5]，其中闡述了雙向力回饋與速度回饋的重要性。直至 90 年代中期，網際網路開始被人們廣泛使用於通訊應用，封包與網路中的傳送，導致了新的資料延遲或遺失問題，因此，相關嶄新的研究也相繼產生。

而近五十年來，遠端操作系統亦廣泛的被運用在眾多科技中。

### 1. 遠端手術系統

頗具名氣的達文西(da Vinci)系統[6]，在現今微創手術的時代，同時利用了腹腔鏡手術，傷口小、出血量少、恢復快的優點與傳統開腹式手術，視野自由、手眼協調的優點，將外科手術技術推向了另一個革命點。透過遠端外科手術系統，可以讓有能力的外科醫生將醫術傳達到全世界的各個角落，

造福更多需要幫助的病人。

## 2. 深海交通工具

由於深海的高壓與低含氧量環境，使人類探索世界的觸角難以進入，但近年來透過深海水底探測器，科學家得以對海底深層的生物採樣研究，另外也可應用於軍事與石油探測等任務。

## 3. 太空機器人

顯而易見的，太空的環境非常不適合人類生存，在太空任務中，重複性高的任務就需要藉由操作機器人手臂達成，如此一來可以減少太空人穿戴裝備的麻煩，更重要的是可以降低危險發生的機率。2004 年 Imaida 與 Yoon 等人發表了在 Engineering Test Satellite 7(ETS-VII)衛星上進行的 slope-tracing 與 peg-in-hole 實驗，證明了即使在延遲時間平均為七秒之久的情況下，系統依然能夠維持穩定並達到可接受的任務完成度[7]。

## 4. 危險物質處理

最早的遠端操作系統就是為了處理核子原料而誕生的，由於核子分裂的過程中所產生的輻射會對生物造成細胞的破壞，所以透過遠端操作可讓工作人員避免受到輻射的汙染。而近期 Wang 與 Yuan 亦有對於針對以封裝的放射性物質，防止外洩之輻射線偵測的相關研究[8]。

## 5. 搜救機器人

在執行救災任務的時候，災難的第一現場往往是人類無法進入之處，以地震的例子來說，當房屋倒塌時，搜救人員進入瓦礫堆中尋找生還者的動作具有極高的危險性，為避免危險發生，有線或無線的搜救機器人因此誕生，對於機器人自主性無法掌握的環境中，即需要人為操作已達成任務。

### 1.3 問題陳述

#### 1. 網路延遲對於底層閉迴路系統造成的影響

由於遠端網路操作系統藉由網際網路(Internet)做為操作端與被控體間的媒介，而網際網路本身存在著不可避免的時間延遲效應，如此的時間延遲，在閉迴路系統中會造成非常嚴重的影響。

以一個簡單的例子來說，圖 1-1 閉迴路控制系統為一個閉迴路控制系統：

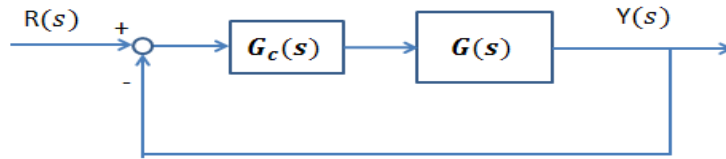


圖 1-1 閉迴路控制系統

$$\text{其系統響應為：} \frac{Y(s)}{R(s)} = \frac{G_c(s)G(s)}{1+G_c(s)G(s)}$$

其中  $G_c(s), G(s)$  為控制器與被控系統的 Laplace Transformation

接著加入網際網路為通訊媒介：

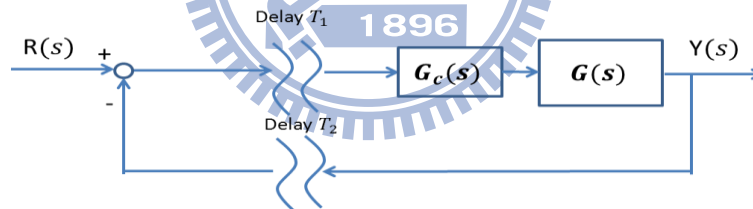


圖 1-2 加入網路的閉迴路控制系統

$$\text{系統響應則變成：} \frac{Y(s)}{R(s)} = \frac{G_c(s)G(s)e^{-T_1s}}{1+G_c(s)G(s)e^{-(T_1+T_2)s}}$$

其中  $T_1$  為 forward-delay time， $T_2$  為 backward-delay time。

值得注意的是，加入網路後的系統極點，會因為網路的延遲效應而改變，即在轉移函數中的  $e^{-(T_1+T_2)s}$  項造成之影響。除此之外，實際的網路延遲時間往往會隨時間、環境、使用者數量甚至天氣變動，如此一來系統的極點就會飄動不定，使系統進入 unstable 的機率增加。這也是為什麼遠端操作系統注重的第一個條件就是穩定度，原因就在於網路延遲非常容易造成系統趨於不穩定。

2. 網路影像監控系統由於網路延遲與工作空間差異造成之操作困難

在本論文架構的操作系統中，操作者藉由遠端網路攝影機傳送回來的影像，進行確認與判斷此時操作點的移動狀態與確切的位置。但是由於網路延遲的關係，操作者看到的網路影像與實際正在發生的情況其實相差了一個網路延遲的時間，在這樣的情況下，操作者會因為無法確認實際發生的狀況而無法給予受控體正確的指令以達到期望的結果。舉例來說，操作者目前看到操作點在  $x_1$  的位置，希望操作點到達  $Y$  點，在影像訊號傳送到操作端時，實際上遠端的操作點已經移動到  $x_2$  的位置，操作者此時會給予  $Y - x_1$  的位置命令，在訊號傳送到被控體後，執行接收到的位置命令  $Y - x_1$ ，就會造成最終操作點超過預期位置的情況發生，如圖 1-3 所示。而由於操作端與受控端的工作空間差異，難以提供精準的操做命令，降低操作上的透明度，也亦為造成循軌誤差的主要原因之一。

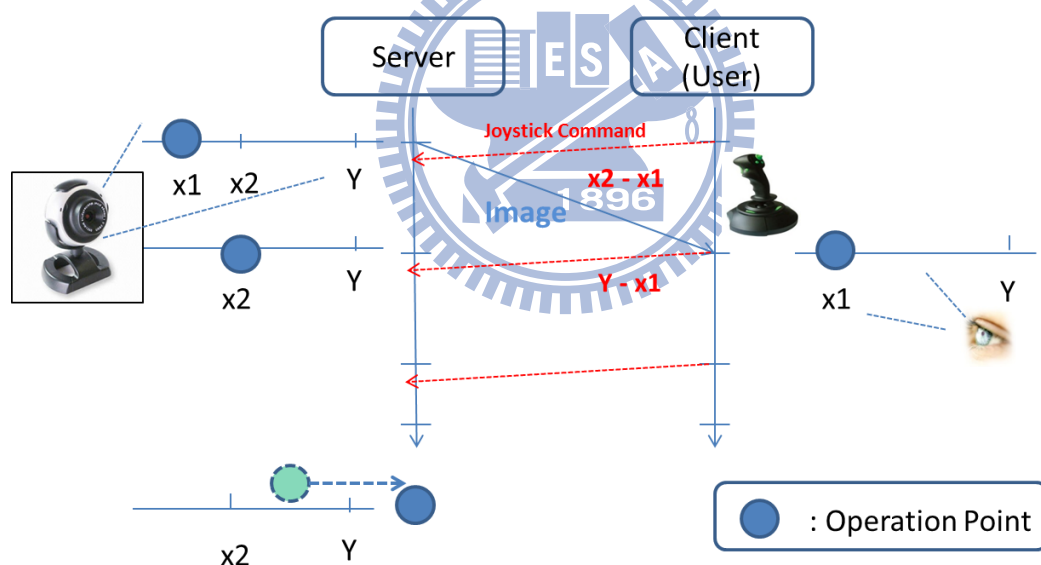


圖 1-3 網路監控系統操作示意圖

## 1.4 研究方法

### 1. 網路延遲對底層系統響應造成之影響：

#### I. 通訊擾動估測器 (communication disturbance observer, CDOB)

利用近年由 Kenji Natori 提出之通訊擾動估測器[9]，將網路延遲的效應視為系統的擾動量，利用估測器將延遲造成的影響一併與回授訊號相抵消，期望最後留下未經過網路延遲影響的響應訊號。但是由於 CDOB 實現於離散系統時會因為取樣頻率不足，造成架構內之積分項的取樣誤差，進而使估測值產生穩態誤差，此穩態誤差與回授訊號結合後容易造成系統的不穩定。所以在此提出了另一簡化架構，使估測器在系統取樣頻率不足的情況下依然可以正常運作。

#### II. 模型預測控制器 (model predictive controller, MPC)

模型預測控制器利用已知受控體模型，透過預測未來響應、最佳化指標與實際回授修正策略，即時計算出最符合預期響應的控制訊號。如此可改善由於其他不確定因素例如訊號延遲造成的影響，所以比一般控制器更能夠克服複雜且不定的受控體對期望響應造成的影響。運用於網路控制系統中，可基於受控體模型估測系統理想情況之輸出響應，在資料遺失情況發生時提供估測迴授值。

#### III. 估測迴授訊號之權重分配機制 (feedback weighting mechanism)

由於 MPC 可基於系統模型估測出系統理想的輸出響應，我們可以利用此估測響應，與 CDOB 估測之系統響應進行權重分配，當作最後未經過網路延遲影響的回授訊號，並且在不同延遲時間下分析各參數之響應特性，驗證所選之參數可使 NSC 有最佳響應。此外在 data dropout 的情況發生時，將權重調整至只使用 MPC 估測迴授，可彌補 CDOB 之估測誤差，進而提升系統穩定度。

## 2. 網路影像延遲對上層造成之操作困難：

### I. 虛擬操作點影像輔助機制

遠端操作系統透過網路影像之命令操作端，與受控端所處的實際工作空間(work space)存在的差異，造成操作者操作上的不真實與不適感，亦為造成循軌誤差的主要原因。本文提出了位置命令在兩工作空間的轉換機制：網路攝影機擷取畫面的受控端所處的工作空間為三維的空間座標，而操作端接收畫面與命令決定的工作空間為二維的空間座標。由於本平台可以透過操作介面確認攝影機照射範圍，所以此時便可將三維轉二維簡化為二維轉二維座標的轉換矩陣，將操作端的位置命令轉換為相對應的受控端位置命令，如此一來操作者可透過虛擬操作點，將遠方受控端等效為近端開迴路操作，且不用完全仰賴網路回傳影像，做為操作判斷依據，由實驗結果可看出此機制大大地提昇循軌準確度與效率。



圖 1-4 虛擬操作點：近端，open loop 操作(無影像定位誤差)

### II. 操作點預測輔助機制

造成循軌誤差的因素除了網路延遲之外，操作者從看到網路攝影機畫面到下達下個操作命令的反應時間也因人而異。雖然虛擬操作點輔助機制可有效降低循軌誤差，但在影像定位誤差的情況下，就會使整個機制失效造成嚴重誤差，所以在此設計另一輔助機制，克服影像定位誤差之影響。首先操作者在等速操作下看到一特定點時停止操作或折回，此時系統會記錄超過預期位置後的操作點移動次數，此資訊即包含了網路延遲所造成的操作誤差與操作者的反應時間。接著利用這些資料與即時估測操作點未來的平均速度，預測如果此時停止操作，操作點最後將抵



達的位置，顯示於畫面給予操作者輔助。由實驗結果可以得知此輔助策略亦可大幅改善循軌誤差並提升效率。

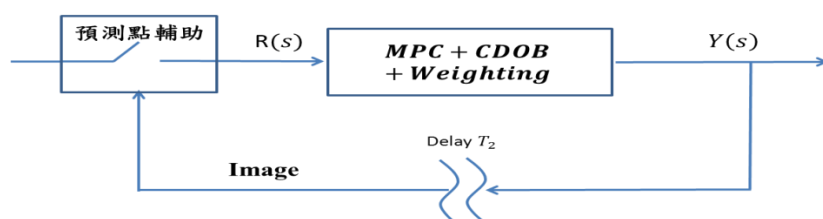


圖 1-5 操作點預測：遠端影像迴授，closed loop 操作(影像定位誤差)

### III. 微調切換機制

由於 perfect delay compensation(PDC)控制架構具有低rise time 與低 settling time 之快速響應，但 overshoot 較大。所以在可容忍震盪之短距離操作情況下，利用影像迴授進行微調時，切換至 PDC 控制架構，可達到快速追跡之效果，並進一步降低循軌誤差並提升效率。

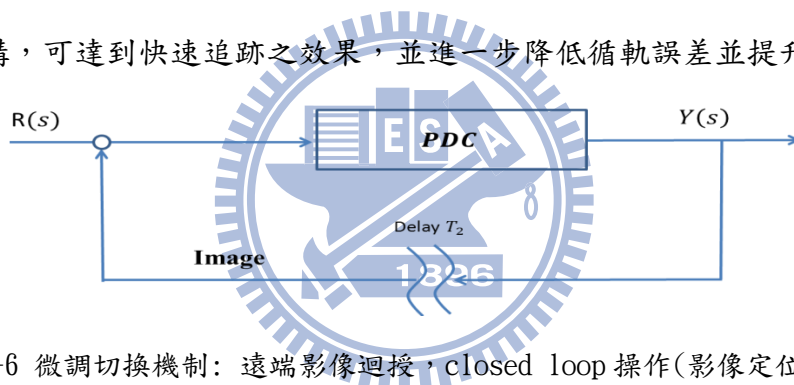


圖 1-6 微調切換機制：遠端影像迴授，closed loop 操作(影像定位誤差)

## 1.5 預期達成目標

### 1. 底層系統響應：

由於本研究實驗環境之平均網路往返延遲時間(round-trip delay, RTT)為 450 ms，加上人類視覺反應時間約為 150 ms [10]，所以在此期望系統響應之 rise time 在 600 ms 以內。如此一來，系統響應在下次操作命令改變之前能夠達到期望響應，避免發生響應永遠追不上操作命令之情況。並在此前提之下，維持系統最大穩定度，降低由於網路延遲對 overshoot 與 settling time 造成之劇烈變動。



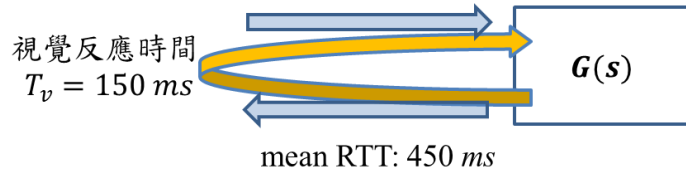


圖 1-7 期望響應速度示意圖

## 2. 上層影像迴授操作系統：

在網路延遲時間變動之環境下，克服網路影像延遲對操作者造成判斷困難之低透明度，並且考慮操作者反應時間等動態特性因素與不同軌跡特性，降低循軌誤差並縮短完成時間。

### 1.6 論文架構

本論文共分六章，第一章簡述研究的動機與目的，並陳述研究的問題和方法。第二章介紹遠端操作系統的相關網路以及實驗平台的開發環境與架構。第三章介紹通訊擾動估測器的架構與理論推導，並且提出 CDOB 實現於離散系統中，取樣頻率不足時的改善架構，接著與 adaptive Smith predictor 比較，驗證在網路延遲時間劇烈變動下，具有較佳性能。第四章介紹模型預測控制器之基本概念、理論推導與於網路控制架構中之實現方法，並與一般在網路控制架構使用的 PI 控制器作比較，並同樣地驗證其效能。由於 MPC 與 CDOB 皆可對迴授訊號進行估測，所以在第五章提出估測迴授之權重分配機制，選擇最佳的迴授訊號權重分配，進一步提高系統穩定度。三到五章介紹的內容主要目的為克服網路延遲對系統響應造成的影響，但即使系統維持快速且穩定之響應，網路延遲效應造成的操作低透明度並不會因此消失。所以接下來在第六章提出的輔助策略機制，便是為了要幫助操作者克服這個網路延遲造成的透明度不足問題，藉由虛擬操作點影像輔助機制、操作點預測與微調切換機制，達到準確且快速循軌的目標。最後，第七章對於本研究成果做分析探討與結論。

## 第二章 遠端操作系統

本章將對本論文中使用的遠端操作系統環境作介紹。首先為各種網路的通訊協定與特性，由於在不同的網路中，系統受到因為延遲的影響也不盡相同，因此對於每種網路的了解有其必要性。接著介紹利用 Visual studio 2008 下 Visual C++ 架設的遠端網路操作系統，包括利用網路攝影機的影像傳輸、DirectX 搖桿驅動物件應用、與追循物件的判定、循軌誤差之定義、遠近端封包格式規劃與網路程式的通訊技術—Windows Sockets。

### 2.1 相關網路協定與介面介紹

#### 2.1.1 實體與資料連結層：乙太網路(Ethernet)

乙太網路(Ethernet)源自於 1973 年 Xerox 公司 Palo Alto Research Center 帕羅奧托研究中心研發設計，起初速度僅 2.49Mbps 並只運用於 Xerox 公司內部。直到 1982 年, DEC, Intel 和 Xerox 等三家公司共同發表了 Ethernet Version 2 規格 (EV2)，此時頻寬已可達 10Mbps，與一般台灣現今家用網路相去不遠，Ethernet 也正式被投入市場，之後 IEEE 也根據 EV2 的內容略加修改後在 1983 年正式通過了 802.3 CSMA/CD 規格。

乙太網路使用 CSMA/CD(載波感應多重存取及衝突偵測)方式來存取介質上資料，在此方式中，每個節點在傳送資料前會先確認是否有其他節點正在傳送資料，如果沒有的話，此節點即可將資料傳送到網路上，如果遇到網路上有許多節點同時要傳送資料的情況，就會造成資料的碰撞，因此所有節點必須對網路狀態進行“監聽”的動作，確認安全後再進行資料的傳送，但是意外也有可能發生，例如兩個節點同時決定要開始傳送資料，此時資料碰撞發生，所有節點即會放棄

接收到的資料，而原先欲傳送資料的節點會等待一隨機決定的時間(避免再次資料碰撞)後再進行傳送資料動作。

而乙太網路的連結方式可以匯流排或是星狀方式連接，所有乙太網路的阻態在邏輯上是一種匯流排。每個訊息框架(Frame)可由網路中任一節點傳送到網路的另一節點中，特定的節點在確認標頭框架後(header)進行資料接受動作。而目前使用最多的乙太網路實現方式為 10BASE-T(雙絞線乙太網路)，10BASE-T 使用星狀網路連接，速度為 10Mbps，最長傳輸距離為 100 公尺。除此之外另外有 10BASE5(粗纜乙太網路)與 10BASE2(細纜乙太網路)，皆使用匯流排傳輸方式連接。

目前使用最廣泛的為快速乙太網路(Fast Ethernet)，於 1995 年由 IEEE 提出，由於提高了最高速率與碰撞區間的大小，碰撞區間也比傳統 Ethernet 降低十倍。Fast Ethernet 中定義了三種 100Mbps Ethernet 規格：100BASE-T4、100BASE-XT 與 100BASE-FX，其差別在於使用傳輸介質的不同與所需要傳輸線的數目。

### 2.1.2 網路與傳輸層：TCP/IP 通訊協定[11]

TCP/IP 為連結網路上電腦主機之通訊協定，此協定最早發源於美國國防部 ARPA 網項目，TCP 全名為傳輸控制協定(transmission control protocol)，而 IP 代表網際網路協定(internet protocol)。TCP 在 OSI 的分類模型中屬於傳輸層，提供了完整的傳輸服務給所需的應用程式；IP 則屬於網路層，對於每個在網路中的主機給予一特定位址。以下針對細節部分做相關簡介：

#### (1)網際網路協定(IP)

在 TCP/IP 模型中，IP 屬於非可靠性、非預接式的資料封包協定，將資料分割成許多較小的封包後進行傳送。封包傳送可以透過不同分割封包、不同路徑，也可以不按順序到達甚至重複到達，當封包到達時，也沒有提供方法將資料封包正確還原，這些問題都由上層協定負責，也就是 TCP 或 UDP，

而由於 UP 提供骨幹傳輸的功能，並且讓使用者自由加入所需要的功能，所以這些限制並不應被視為缺點。

## (2)傳輸控制協定(TCP)

TCP 為一種連結導向的、可靠的、基於位元組串流的傳送層通訊協定，提供完整的傳輸服務給所需的應用程式，並且在應用層與網路層之間提供應用程式與網路運作的中介服務。與 IP 最大的不同之處為，TCP 在傳送資料時會要求對方在接收資料後回傳確認訊號，如果在某個時限內 TCP 沒有接收到此訊號，即會開始重新進行封包的傳送，並且接收端如果接收到排序錯誤的資料封包，TCP 會將其重新排序。而 TCP 也會使用滑動窗口(sliding window)的協定來做流量控制，TCP 總是告知對方它能夠接收資料的位元組數。建立一個 TCP 連線，需要經過以下步驟：

1. 伺服器準備接收連線
2. 客戶端發起連線請求並傳送一 SYN 訊號給伺服器端
3. 伺服器確認 SYN 訊號後，同時回傳一個 SYN 訊號，並進行確認動作
4. 客戶端確認伺服器送來的 SYN 訊號

而中止連線時一般需要四個步驟：

1. 其中一方執行關閉，並由 TCP 向另一方傳送 FIN(結束)訊號
2. 接收方接收到結束訊號，並且由 TCP 回傳確認訊號
3. 接收方也執行關閉，此時同樣的也由 TCP 傳送一結束訊號
4. 原傳送方接收到此結束訊號後，也由 TCP 回傳確認訊號

### 2.1.3 傳輸層到應用層之通訊界面：Windows Sockets

Windows Sockets 是由 Microsoft 公司聯合 Intel、Sun、Informix、Novell 等公司共同制定的一套開放的、支援多種協定並於 Windows 作業系統下的網路編程介面。Windows Sockets 在 OSI 模型中，介於網路通訊協定(TCP/IP)與應用層之

間，應用程式透過呼叫 Windows Sockets API(application programming interface)來實作資料通訊，利用下層的網路通訊協定、作業系統與實體介質完成了資料的傳輸。Socket 即為應用層到傳輸層的介面，由於 TCP 或 UDP 可同時對多個應用程式提供服務，而有時候可能會使用同一個 TCP 協定埠傳輸資料，為了區別不同的連線和應用程式的行程，Socket 則為應用程式與 TCP/IP 協定交互的介面。使用 TCP Socket 編程的流程圖如圖 2-1 所示：

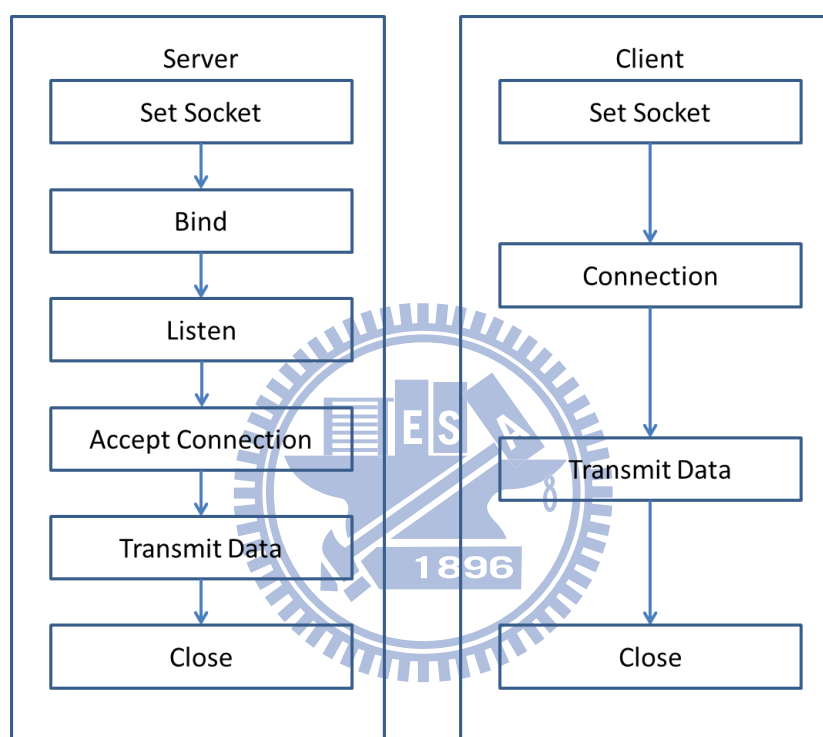


圖 2-1 TCP Socket 客戶端/伺服器

首先客戶端在得知伺服器 IP 位址的前提下，伺服器首先進行 Windows Socket 的初始化，建立 Socket 後將伺服器綁訂(Bind)到一個已知的名字上，接下來進入監聽(Listen)狀態，等待客戶端連線請求，收到客戶端連線請求後，伺服器透過 accept() 函式接受此連線請求，即可開始進行資料通訊。對客戶端而言，首先發出連線請求後，如果伺服器端接受了該請求並回傳確認訊號，就可以進行資料通訊。

## 2.2 遠端操作系統設計

### 2.2.1 開發環境

本論文的實驗平台軟體使用 Microsoft 公司開發的 Visual Studio 2008 Professional 版，使用 Visual C++ 語言編寫程式。Visual Studio 2008 提供了強大的整合程式開發環境，利用程式資源的建立工具，讓視窗程式的開發不再是程式設計師的夢靨。MFC(Microsoft Foundation Class)即為針對視窗程式開發的 AF(Application Frameworks)，其中可分為應用程式類別、繪圖類別、網際網路類別、容器類別、資料庫類別與其他類別。應用程式類別中定義了建立應用程式可能要用到的類別，如 CWinApp (視窗應用程式類別)、CDocTemplate(文件範本類別)、CDocument(文件類別)...等等。繪圖類別包含了 CDC(裝置內文類別)、Cpen(畫筆類別)、CBrush(筆刷類別)...等等。網際網路類別用於建立網路連結，有 Ftp、Gopher 與 Http 三種。資料庫類別用於資料庫建立連結，主要有 ODBC 資料庫類別、DAO 資料庫類別與 CRecordset 類別(紀錄集類別)。容器類別包括 Clist、CArray、CMap...等容器類別提供使用者記錄不同資料。除此資外 MFC 還有例外處理類別、檔案操作...等其他類別。硬體部分，本研究之操作平台於操作端使用無力回饋的一般市售遊戲搖桿，成本低廉且取得容易，利用 DirectX8 提供之 API 與軟體進行連接與類比資料讀取。受控端則使用一般市售網路攝影機，對顯示於螢幕之虛擬操作畫面進行監控，利用 Microsoft Video for Windows(VFW) 提供之函數操作連接硬體，VFW 之優點為操作容易，對於只需要用到基本需求的使用者而言，實現速度與軟體複雜程度比 DirectrX 來的快速且簡單。

硬體架構如圖 2-2 所示，操作者藉由搖桿透過位置命令，移動遠端的虛擬操作點，並對特定軌跡進行循軌動作，期望迅速且精確地完成任務，其控制架構如

圖 2-3 所示，並且在本實驗中設定系統模型  $G(s) = \frac{6}{s+6}$ 。



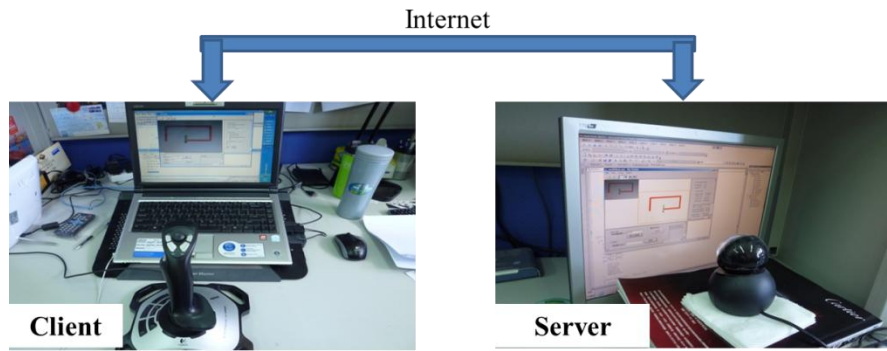


圖 2-2 遠端操作系統實體照片

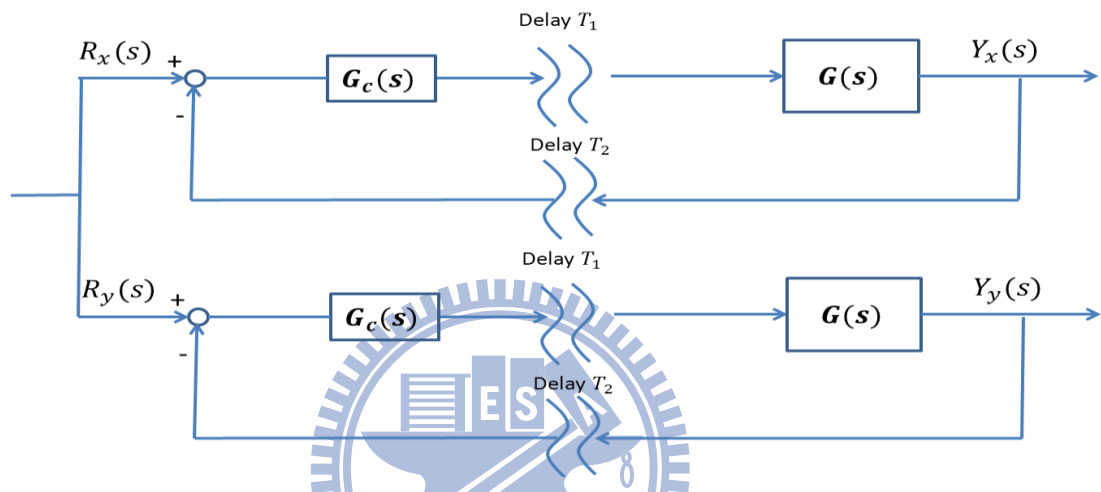


圖 2-3 遠端網路系統控制架構

## 2.2.2 操作介面與軟體架構

由於遠端操作系統中分為操作端(Client)與受控體端(Server)，所以需要設計不同的操作平台介面。首先介紹操作端之操作平台介面，如圖 2-4 所示：



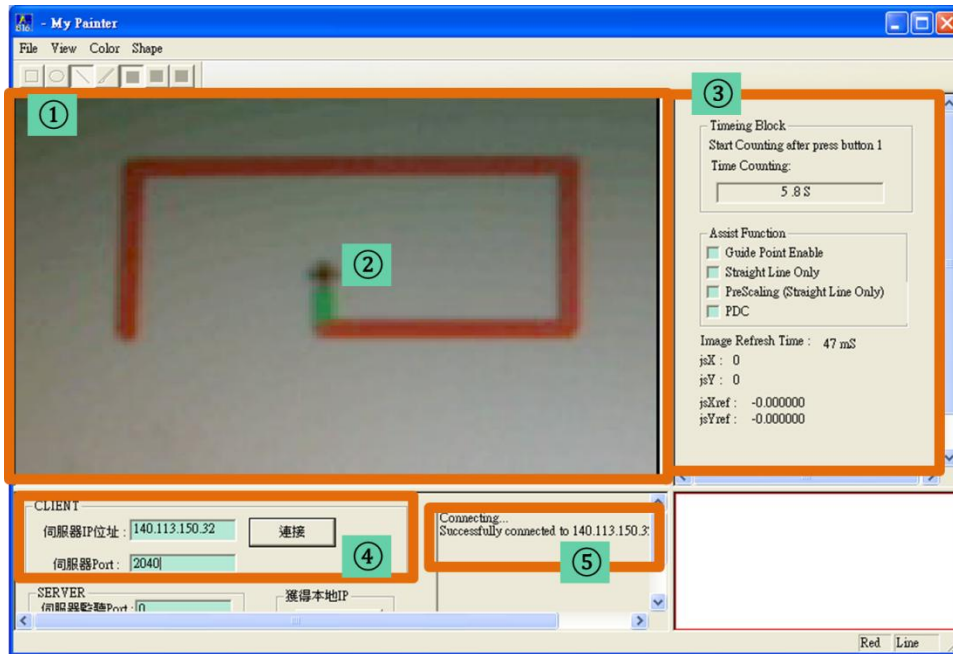


圖 2-4 操作端(Client)操作介面

- ①：網路攝影機回傳畫面，圖中粗線段為循軌目標，在此以五條 XY 方向、長度不一之直線為循軌目標，操作者透過搖桿操縱圖中②之操作點，一一對既定軌跡進行循軌操作
- ②：操作點，顏色不同之粗線為目前追循軌跡，進行完第一段軌跡追蹤後，第二段軌跡會改變顏色以提醒操作者目前欲追循的目標
- ③：紀錄經過時間、輔助機制選取方塊、畫面更新時間、目前傳送之位置命令與提供操作者監控之回授訊號
- ④：TCP/IP 連線之 IP 位址與伺服器埠設定
- ⑤：網路連線狀態

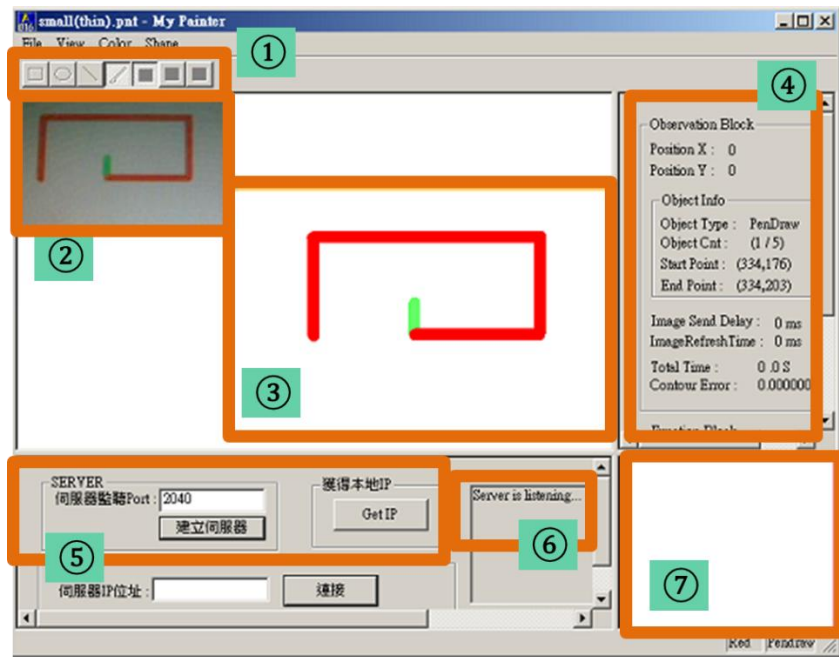


圖 2-5 受控端(Server)操作介面

- ①：繪圖工具列，用來畫出特定圖形，即操作點將追循之軌跡
- ②：網路攝影機畫面。確認網路攝影機正常運作，在虛擬操作點輔助機制中可由此畫面得知網路攝影機之畫面照射定位範圍
- ③：繪圖區與操作點操作範圍，預定追循之軌跡顯示於此，軌跡之各部分會依據循軌進度更換顏色提供操作者進行確認
- ④：操作點目前座標位置、軌跡物件座標位置、畫面更新時間、操作者使用時間與循軌誤差
- ⑤：網路連接控制方塊，由 Get IP 按鈕可得知本地端 IP 位址，以提供給操作端進行連接
- ⑥：網路連線狀態
- ⑦：繪圖區之局部放大方塊

### 2.2.3 遠近端傳送封包格式

最後，如果操作端(Client)與受控端(Server)要進行資料交換，資料封裝格式必須固定，如此才能進行正確的資料解譯，在此我們定義兩端傳送之訊號封包，如圖 2-6 所示：

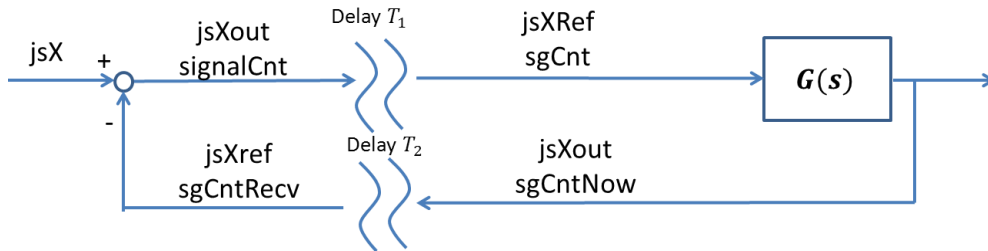


圖 2-6 訊號名稱定義

1. 操作端傳送至受控端封包格式(Client  $\Rightarrow$  Server)：

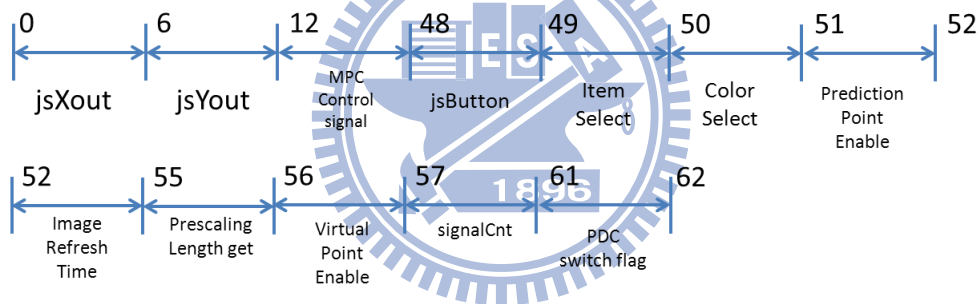


圖 2-7 操作端傳送封包

MPC Control Signal 為 MPC 算出之控制訊號列，本實驗中未使用，此為預留備用。jsButton 為搖桿按鍵編號，Item Select、Color Select 為繪圖工具切換，Prediction Point Enable 為致能操作點預測機制訊號，Virtual Point Enable 為致能虛擬操作點輔助機制訊號，signalCnt 為訊號標籤，用以確認回授訊號編號，用以量測 RTT 與偵測資料是否遺失。

## 2. 受控端傳送至操作端封包格式(Server⇒Client)

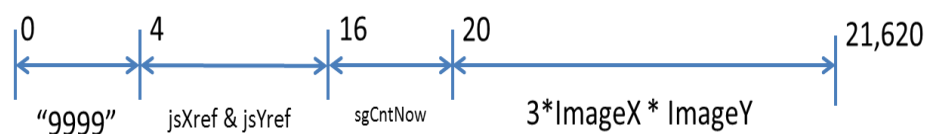


圖 2-8 受控端傳送封包

封包最前端"9999"為提供操作端進行封包切割用途，jsX/Yref 為響應之回授訊號，sgCntNow 為訊號編號，最後為網路攝影機之影像陣列。

### 2.2.4 循軌誤差(Contouring error)定義

本論文在此實驗平台中分別設計直線與斜線進行循軌操作。

#### 一、方形直線循軌：

在進行方形直線循軌時只提供操作搖桿兩個自由度，即操作者只能沿著 X 軸方向與 Y 軸方向操作，原因在於期望排除操作者之操作熟練度考量，只考慮網路延遲對操作者在操作上的影響，所以操作方法應越簡單越好。而在此設計之追循軌跡皆為 XY 方向的直線，操作者利用搖桿操作遠端操作點分別對每段直線進行循軌動作，唯當操作點準確地從一段直線之起點往終點方向移動時，系統即判斷操作者已開始對此線段進行循軌動作，並且將其線段之顏色改變以告知操作者(圖 2-10)，如果操作點超過或提前改變方向，系統則會判斷循軌度做尚未完成，所以線段顏色不會改變(圖 2-9)。而操作點超過線段終點之距離與操作點超過線段終點後，往反方向操作若再次超過線段終點之距離，即為定義之循軌誤差(contour error)，如圖 2-11 所示。



圖 2-9 目前線段之循軌未完成

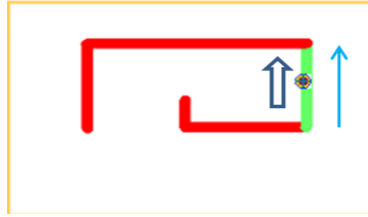


圖 2-10 前一線段循軌完成

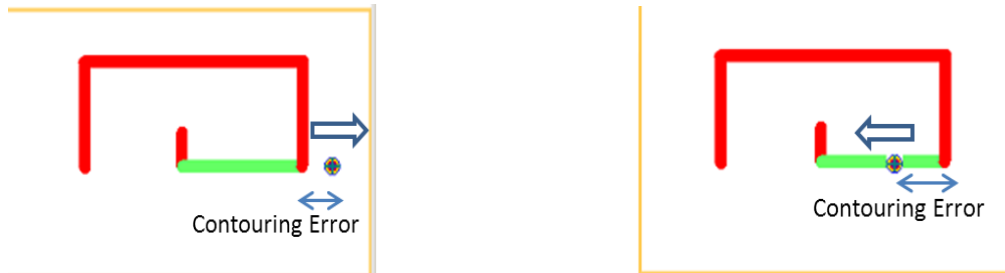


圖 2-11 方形直線循軌誤差定義

## 二、菱形斜線循軌：

在菱形斜線循軌的實驗中，亦為分別對各斜線段進行循軌動作。當操作點位於目前循軌物件之終點時(圖 2-12 (b))，系統即會判斷此線段循軌完成，進行下一個線段之循軌。

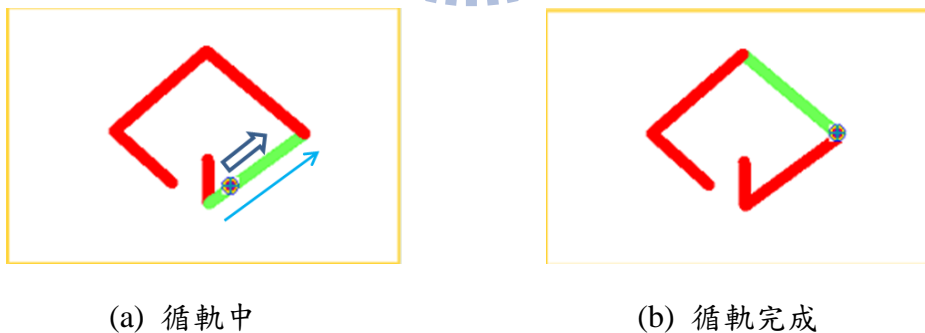
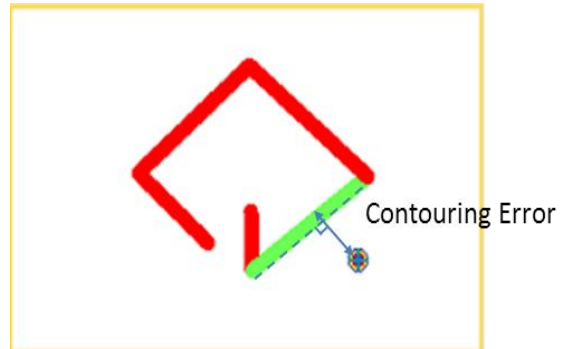
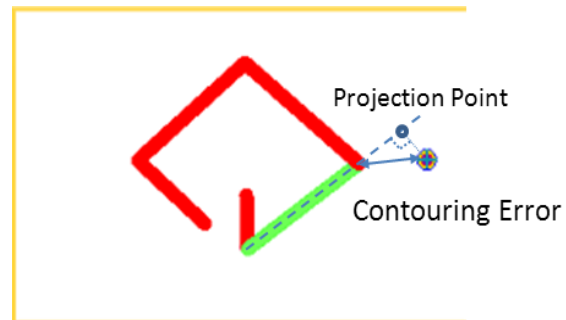


圖 2-12 循軌狀態判斷

在斜線之循軌實驗中，循軌誤差之定義為操作點與欲追循線段之最近距離(圖 2-13 (a))，當操作點對於線段之投影點超過線段終點時，循軌誤差則為操作點至目前循軌線段終點之距離(圖 2-13 (b))。



(a) 操作點之投影點位於追循線段中



(b) 操作點之投影點位於欲追循線段外

圖 2-13 菱形斜線循軌誤差定義

### 第三章 通訊擾動估測器之實現與改進技術

#### 3.1 通訊擾動估測器(communication disturbance observer, CDOB)

這一小節將對通訊擾動估測器之概念與原理做詳細的介紹，首先必須先了解的是將網路延遲視為系統的擾動之概念，接著藉由通訊擾動估測器，估測並消除由於網路延遲對回授訊號造成的影響。

#### 3.2 網路擾動之概念

圖 3-1 為一個網路系統示意圖，如果從控制端送出的控制訊號與回授訊號角度來看，控制訊號 $U(s)$ 在進入受控系統之前會經過網路延遲時間 $T_1$ ，受控系統產生的輸出訊號再經過回授的網路時間延遲 $T_2$ ，最後回到控制端，響應訊號為 $Y(s)$ ，等效系統如圖 3-2。此系統之狀態空間表示如下：

$$\begin{aligned} \dot{x}(t) &= Ax(t) + bu(t - T_1) \\ y(t) &= cx(t - T_2) \end{aligned} \quad (3-1)$$

$x(t)$ 為 $n \times 1$ 階的系統狀態向量， $u(t)$ 為輸入訊號， $y(t)$ 為輸出訊號， $A$ 為 $n \times n$ 矩陣， $b$ 為 $n \times 1$ 階向量， $c$ 為 $1 \times n$ 階向量， $A, b, c$ 為系統特性矩陣。

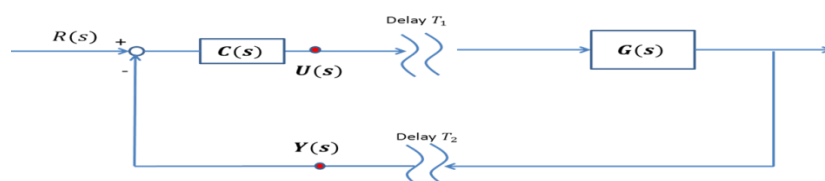


圖 3-1 網路控制系統示意圖

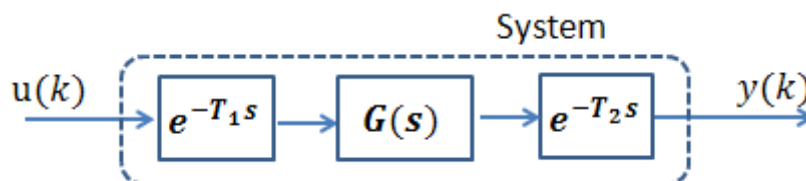


圖 3-2 網路控制系統等效圖



Laplace 轉換如下：

$$\begin{aligned} sX(s) &= AX(s) + bU(s)e^{-T_1s} \\ Y(s) &= cX(s)e^{-T_2s} \end{aligned} \quad (3-2)$$

由上式(3-2)可得系統輸出與輸入訊號關係如下：

$$\begin{aligned} \frac{Y(s)}{U(s)} &= c(sI - A)^{-1}be^{-T_1s}e^{-T_2s} \\ &= G(s)e^{-Ts}, \quad (T = T_1 + T_2) \end{aligned} \quad (3-3)$$

由輸入與輸出的關係可看出，系統的響應由系統特性 $G(s)$ 與一延遲因素 $e^{-Ts}$ 構成。同樣的觀點下，將系統示意如圖 3-3 所示，此時系統已無延遲的控制方塊，取而代之的是一擾動量，如下式表示：

$$d_{net}(t) = u(t) - u(t - T) \quad (3-4)$$

Laplace 表示式：

$$D(s) = U(s) - U(s)e^{-Ts} \quad (3-5)$$

此擾動量定義為”網路擾動”(network disturbance)，換言之，我們將網路延遲的因素視為網路擾動效應。由圖 3-3 中可得到系統的輸出與輸入關係式如下：

$$\begin{aligned} \dot{x}(t) &= Ax(t) + bu(t) - bd_{net}(t) \\ y(t) &= cx(t) \end{aligned}$$

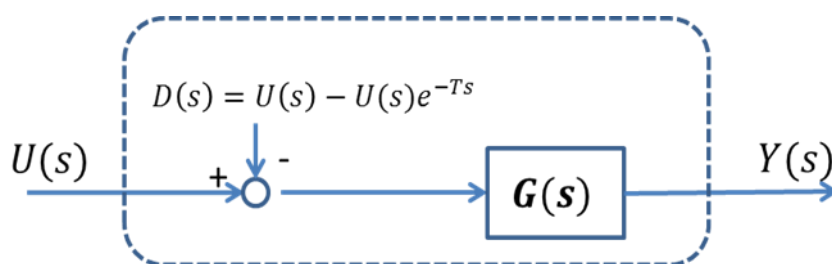


圖 3-3 將網路延遲視為擾動量之示意圖

經 Laplace 轉換，可得系統之輸出與輸入關係式如下：

$$\begin{aligned} \frac{Y(s)}{U(s)} &= c(sI - A)^{-1}be^{-T_1s}e^{-T_2s} \\ &= G(s)e^{-Ts} \end{aligned} \quad (3-6)$$

由式(3-3)與式(3-6)的比較後可發現系統為等效，所以驗證了將網路延遲效應視為網路擾動的概念，且此架構在只有單向網路延遲時( $T_1 = 0$  或  $T_2 = 0$ )亦成立。

### 3.3 通訊擾動估測器之設計

基於任何擾動估測器的目的皆為對擾動量進行估測，所以此時我們也必須將網路擾動(network disturbance)加入系統狀態空間(state space)之狀態變數(state variable)。此時設定之通訊擾動估測器為一階系統，故其能估測之網路擾動  $d(t) = constant$ 。系統狀態空間表示式如下：

$$\begin{bmatrix} \dot{x}(t) \\ \dot{d}_{net}(t) \end{bmatrix} = \begin{bmatrix} A & B \\ 0_{2 \times n} & 0_{2 \times 2} \end{bmatrix} \begin{bmatrix} x(t) \\ d_{net}(t) \end{bmatrix} + \begin{bmatrix} b \\ 0_{2 \times 1} \end{bmatrix} u(t)$$

$$y(t) = [c \quad 0_{1 \times 2}] \begin{bmatrix} x(t) \\ d_{net}(t) \end{bmatrix} \quad (3-7)$$

其中， $0_{N \times N}$  為  $N \times N$  之零矩陣， $B = [-b \quad 0_{n \times 1}]$ 。

所以通訊擾動估測器(CDOB)之狀態空間表示式如下：

$$\begin{bmatrix} \dot{\hat{x}}(t) \\ \dot{\hat{d}}_{net}(t) \end{bmatrix} = \begin{bmatrix} A & B \\ 0_{2 \times n} & 0_{2 \times 2} \end{bmatrix} \begin{bmatrix} \hat{x}(t) \\ \hat{d}_{net}(t) \end{bmatrix} + \begin{bmatrix} b \\ 0_{2 \times 1} \end{bmatrix} u(t) + k(y(t) - \hat{y}(t))$$

$$\hat{y}(t) = [c \quad 0_{1 \times 2}] \begin{bmatrix} \hat{x}(t) \\ \hat{d}_{net}(t) \end{bmatrix} \quad (3-8)$$

其中， $k$  為估測器之增益(observer gain)。

估測的網路擾動量為：

$$\hat{D}(s) = \frac{g_{net}}{s + g_{net}} D(s) \quad (3-9)$$

其中  $g_{net}$  為 CDOB 之極點。

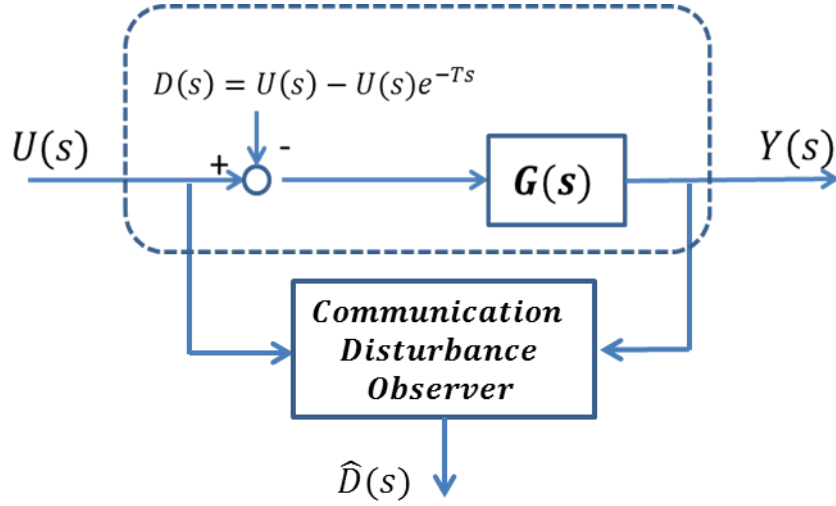


圖 3-4 網路擾動估測示意圖

由於我們期望利用 CDOB 估測網路擾動(ND)，所以必須對網路擾動的動態特性進行分析，進而設計 CDOB。藉由 Pade approximation，可將網路擾動量利用下列式子近似：

$$e^{-Ts} = \frac{1 + \sum_{i=1}^m \frac{m(m-1)\cdots(m-i+1)}{(m+n)(m+n-1)\cdots(m+n-i+1)} \frac{1}{i!} (-Ts)^i}{1 + \sum_{i=1}^n \frac{n(n-1)\cdots(n-i+1)}{(m+n)(m+n-1)\cdots(m+n-i+1)} \frac{1}{i!} (Ts)^i}$$

$$= P_{mn}(s) \quad (3-10)$$

此時  $D(s) = U(s)(1 - P_{mn}(s))$

$$= U(s) \frac{n(s)}{d(s)} \quad (3-11)$$

令  $d(s) = 0$ ，可得網路擾動量之極點。

然而 CDOB 因為之速度必須比網路擾動之動態特性快速，才能夠進行正確的估測，亦即 CDOB 之極點的實部絕對值必須大於網路擾動特性之極點的實部絕對值，如圖 3-5 所示：

$$\text{Re}[p_{cdob}] < \text{Re}[p_{nd}]$$

$$\Leftrightarrow g_{net} > -\text{Re}[p_{nd}]$$

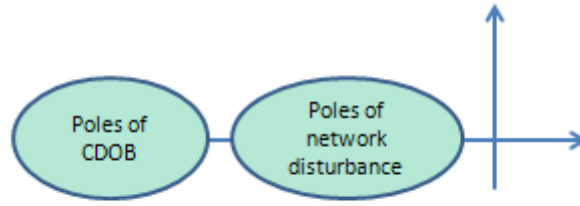


圖 3-5 CDOB 與網路擾動模型極點關係

將網路擾動量估測出來後，利用時間延遲補償機制(time-delay-compensation method)，將經過網路延遲之回授訊號進行補償，其架構如下圖 3-6 所示。

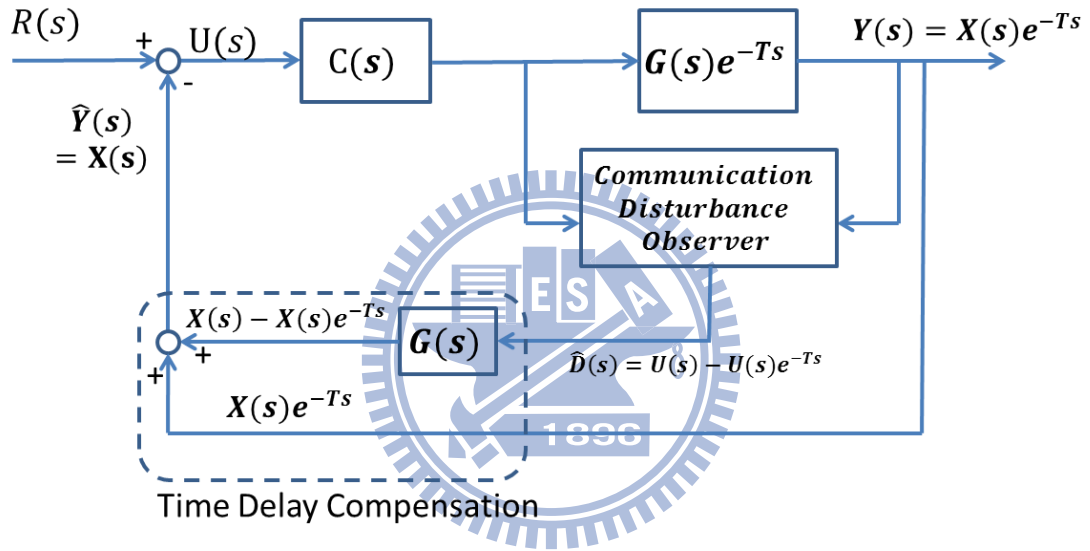


圖 3-6 CDOB 中網路延遲補償架構示意圖

### 3.4 CDOB 設計頻寬限制與系統取樣頻率不足時之估測誤差

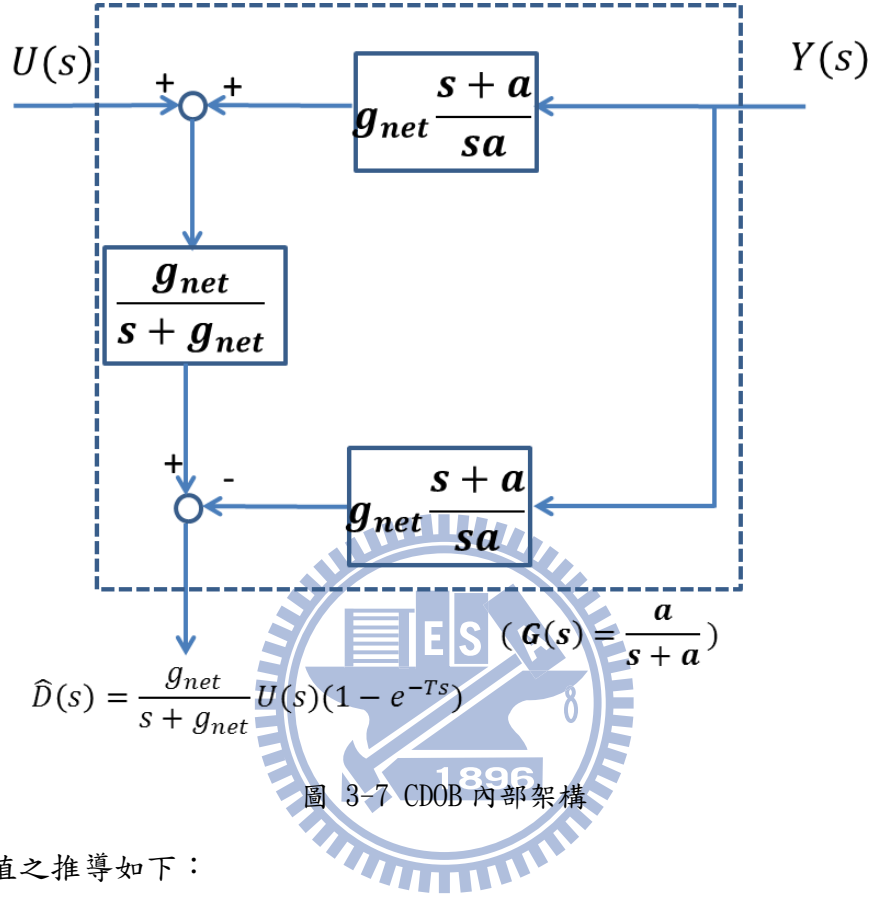
理想狀況下，如果網路擾動能夠精確地被估測，則  $\hat{Y}(s) = X(s)$ ， $X(s)$  為未經過網路延遲效應之系統響應，即  $U(s)C(s)G(s) = X(s)$ ； $X(s)e^{-Ts}$  則為操作端接收到的回授訊號。系統閉迴路之轉移方程式如下：

$$\frac{Y(s)}{R(s)} = \frac{C(s)G(s)e^{-Ts}}{1 + C(s)G(s)}$$

此時系統之 characteristic equation： $\Delta(s) = 1 + C(s)G(s)$ ，將不會受到延遲項  $e^{-Ts}$

影響，系統也因此不會因為網路延遲而造成 unstable 之情況。

最後我們可由提出此架構之作者 Kenji Natori [9] 的文章中可得知，CDOB 實現的內部架構如下圖 3-7 所示：



其估測值之推導如下：

$$\begin{aligned}
 \hat{D}(s) &= \frac{g_{net}}{s + g_{net}} U(s) + g_{net} \frac{s+a}{sa} Y(s) e^{-Ts} \frac{g_{net}}{s + g_{net}} - g_{net} \frac{s+a}{sa} Y(s) e^{-Ts} \\
 &= \frac{g_{net}}{s + g_{net}} U(s) + g_{net} \frac{s+a}{sa} \frac{-s}{s + g_{net}} Y(s) e^{-Ts} \\
 &= \frac{g_{net}}{s + g_{net}} U(s) (1 - e^{-Ts})
 \end{aligned}
 \tag{3-12}$$

由圖 3-7 可觀察出，回授訊號  $Y(s)$  經過一積分項  $(g_{net} \frac{s+a}{sa})$  後，與控制訊號  $U(s)$  相加，並通過一低通濾波器  $\frac{g_{net}}{s+g_{net}}$ ，最後與訊號  $(g_{net} \frac{s+a}{sa}) Y(s)$  相減後可得到

CDOB 對網路擾動的估測值  $\hat{D}(s) = \frac{g_{net}}{s+g_{net}} U(s) (1 - e^{-Ts})$ 。

但由於本研究實驗平台使用的網路攝影機的影像陣列龐大，所以系統之取樣時間被限制於  $0.05 \text{ sec}$  以上，如此一來通訊擾動估測器的頻寬就相對地被限制，理想的情況中， $g_{net}$  越大代表 CDOB 頻寬越大，估測的效果越佳，但在離散系統中實現  $\frac{g_{net}}{s+g_{net}}$  的前提為取樣頻率要夠快，使  $\frac{g_{net}}{s+g_{net}}$  在離散系統中與在連續時間內的響應不失真。當訊號進入一取樣時間  $T_s = 0.05 \text{ s}$  之系統中，訊號不失真的部分為頻率小於  $\frac{\omega_N}{2} = \frac{\pi}{T_s} = 62.8 \text{ (rad/s)}$  的部分，這也是圖 3-8 波德圖橫座標終點只到  $62.8 \text{ (rad/s)}$  的原因。而當我們用固定的取樣頻率對一個連續系統  $\frac{g_{net}}{s+g_{net}}$  取樣時，無論  $g_{net}$  多大，訊號進入此系統不失真的部分皆在頻率小於  $62.8 \text{ (rad/s)}$  處，所以由模擬結果圖 3-8 可看到系統  $-3\text{dB}$  大約位於  $62.8 \text{ (rad/s)}$  附近，由此可得知而此時之  $g_{net} = 15$  值即為通訊擾動估測器於  $T_s = 0.05 \text{ s}$  時之頻寬極限。

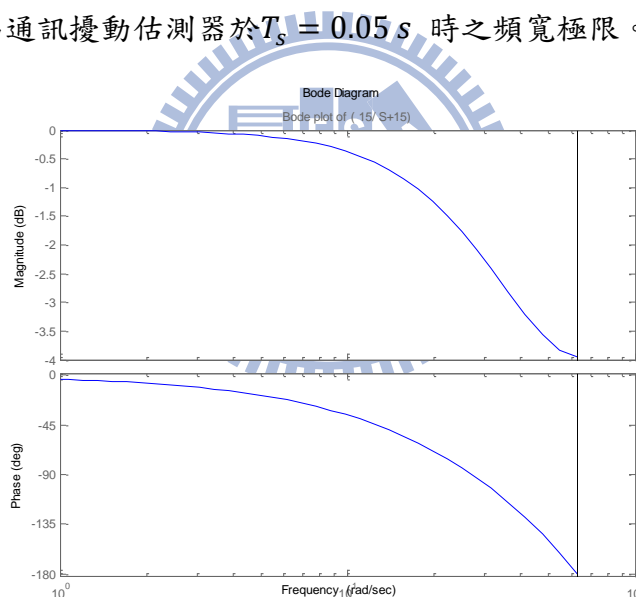


圖 3-8 離散系統中  $g_{net}/(s+g_{net})$  之波德圖 ( $g_{net} = 15$ )

而在取樣頻率不足的情況下，積分項  $(g_{net} \frac{s+a}{sa})$  於離散系統中之實現就會造成誤差，由(3-12)式中可發現有一於原點處之極零點對消效應，在連續系統中原本可達到的效應，在取樣頻率不足時會無法達成，進而造成穩態誤差，並影響 CDOB 估測性能。觀察(3-12)式可得知，系統之終值會收斂於零，如圖 3-9 所示：

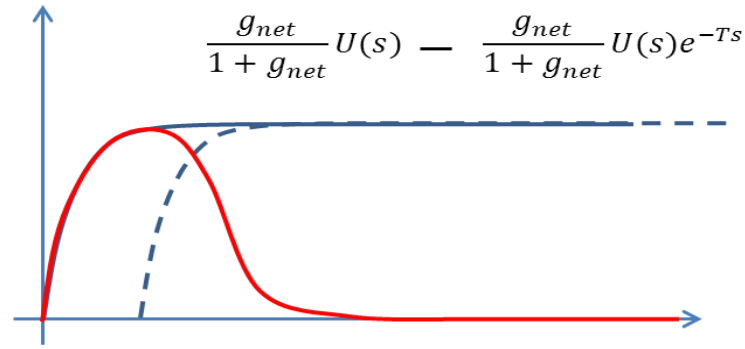


圖 3-9 CDOB 估測值

但是系統取樣頻率不足時，積分效應將無法理想地被微分效應抵消，所以在離散系統的實現中，會產生一穩態誤差。為驗證此觀點，在此分別設計兩等效系統架構，如圖 3-10 所示，並且在不同取樣頻率下轉換為離散系統實現之。模擬結果如圖 3-11 所示，在取樣時間與本研究相同的  $T_s = 0.05 \text{ s}$  情況下，會存在一穩態誤差。而在取樣時間為  $0.005 \text{ s}$  較快之系統，可將積分效應與微分效應幾乎相抵消，此等效系統之響應相同。

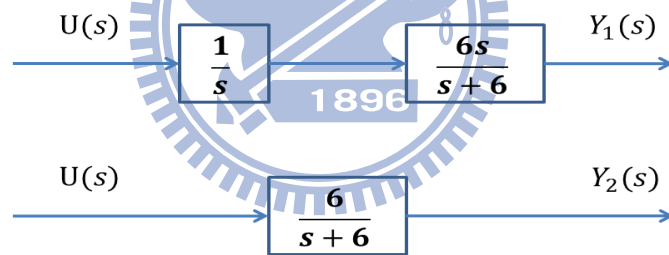
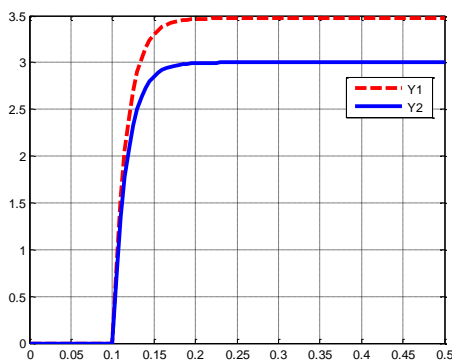
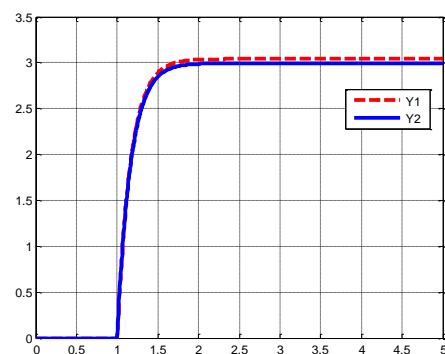


圖 3-10 兩等效控制系統架構



$T_s = 0.05 \text{ s}$



$T_s = 0.005 \text{ s}$

圖 3-11 相同系統於不同取樣時間下之模擬結果



接著將原 CDOB 架構實現於離散系統中，如圖 3-12 所示，其中網路往返延遲時間為 500 ms、取樣頻率為 50 ms。由(3-14)式終值定理計算可得，穩態誤差為輸入訊號增益值  $k$  之-0.421 倍。

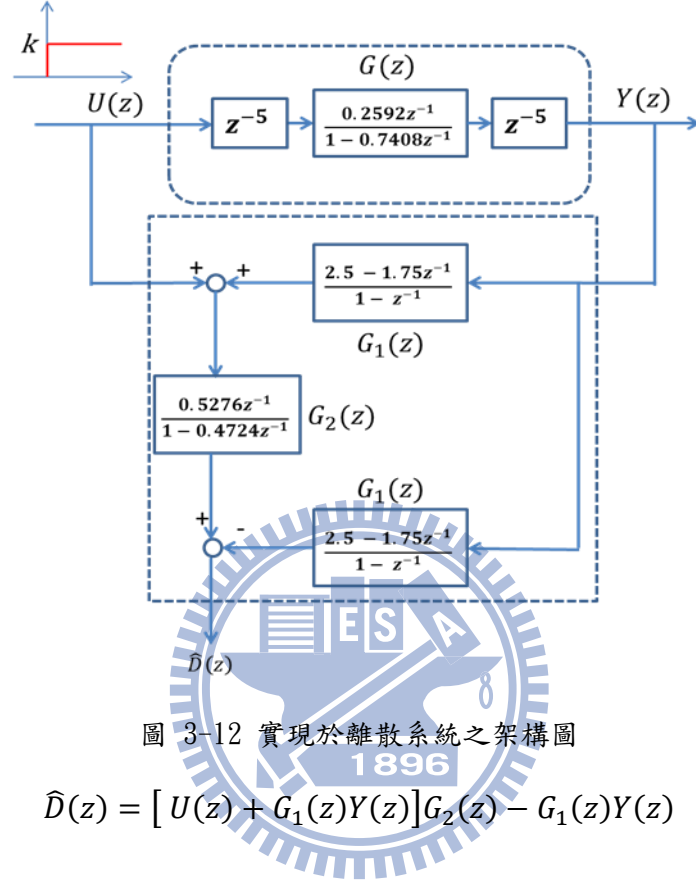


圖 3-12 實現於離散系統之架構圖

$$\hat{D}(z) = [U(z) + G_1(z)Y(z)]G_2(z) - G_1(z)Y(z) \quad (3-13)$$

$$\begin{aligned} \lim_{z \rightarrow 1} (z-1)\hat{D}(z) &= k \left( \frac{0.5276z^{-1}}{z^{-1}(1-0.4724z^{-1})} + \frac{(2.5-1.75z^{-1})0.2592z^{-10}}{(1-z^{-1})(1-0.7408z^{-1})} \frac{-(1-z^{-1})}{1-0.4724z^{-1}} \right) \\ &= k \left( 1 - \frac{0.2592(2.5-1.75)}{(1-0.7408)(1-0.4724)} \right) = -0.421k \end{aligned} \quad (3-14)$$

由模擬結果圖 3-13 可驗證此結果。

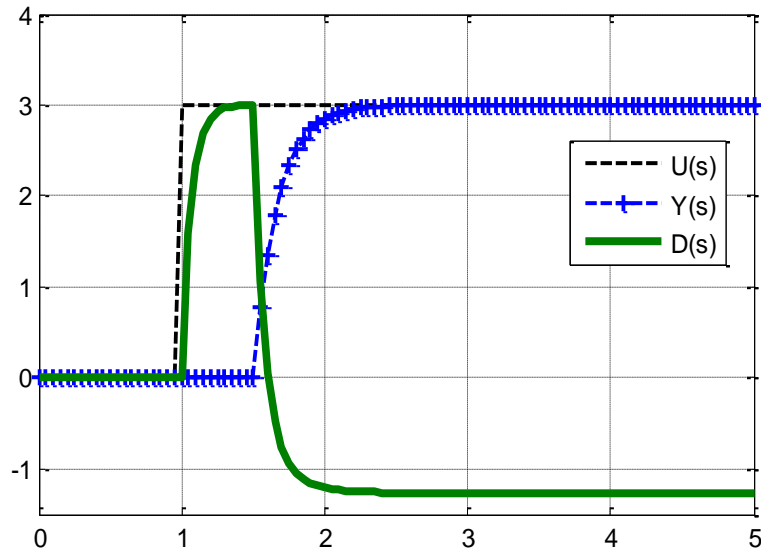


圖 3-13 CDOB 極點略小於網路擾動模型極點時之網路擾動估測值

### 3.5 系統取樣頻率不足時之 CDOB 改良架構

本論文提出一改善架構，將迴授訊號與控制訊號分開處理，並且將積分項與微分項相消除後再對系統進行取樣，轉換為離散系統後實現之，連續系統之系統架構如圖 3-14 所示。

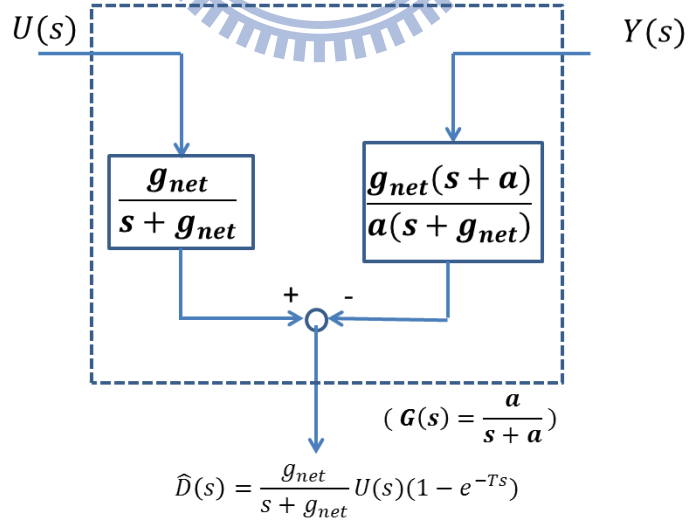


圖 3-14 CDOB 之改善後內部架構

在離散系統的實現上，架構如圖 3-15 所示。由(3-16)式終值定理可得知，此時之穩態誤差已降為輸入訊號增益之-0.0007 倍，可視為趨近於零。

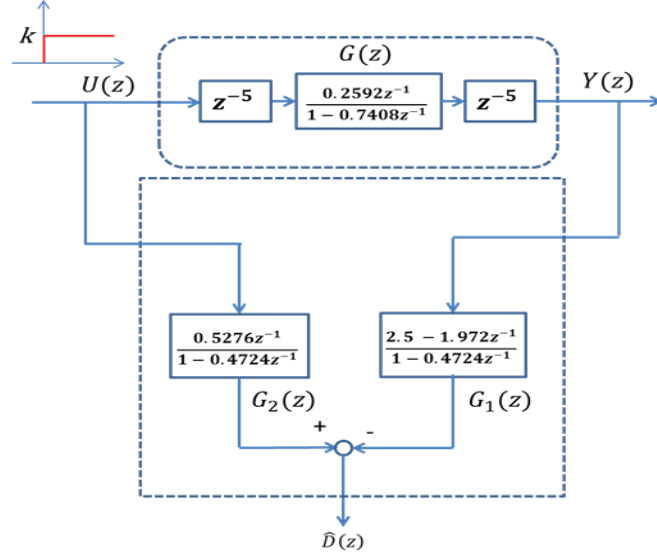


圖 3-15 改善架構於離散系統實現

$$\hat{D}(s) = U(s)G_1(z) - Y(s)G_2(z) \quad (3-15)$$

$$\begin{aligned} \lim_{z \rightarrow 1} (z-1)\hat{D}(z) &= \frac{k}{z^{-1}} \frac{0.5276z^{-1}}{1-0.4724z^{-1}} - \frac{kz^{-10}}{z^{-1}} \frac{0.2592z^{-1}}{1-0.7408z^{-1}} \frac{2.5-1.972z^{-1}}{1-0.4724z^{-1}} \\ &= k \left( 1 - \frac{0.528}{0.5276} \frac{0.2592}{1-0.7408} \right) = -0.0007k \rightarrow 0 \end{aligned} \quad (3-16)$$

由圖 3-16 之模擬結果可看到，CDOB 估測的訊號收斂至零點，如此一來可避免系統因為估測誤差而導致閉迴路後的穩態誤差甚只可能進入不穩定狀態。

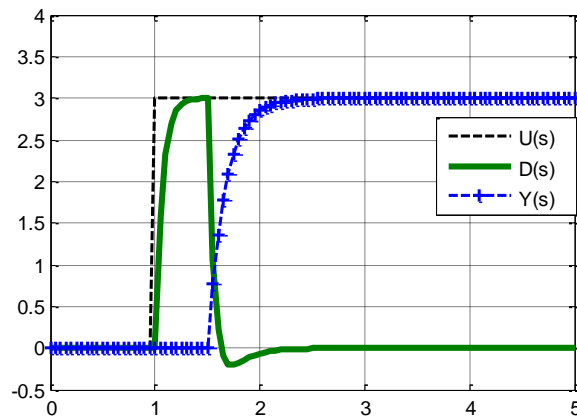


圖 3-16 CDOB 極點略小於網路擾動模型極點時之網路擾動估測值(簡化後架構)

CDOB 架構改善前後對於估測之系統輸出迴授訊號影響，可由架構圖 3-17 中分析，模擬結果如圖 3-18 與圖 3-19 所示。由模擬結果可得知，在系統取樣頻率不足的情況下，使用原本 CDOB 架構估測並迴授補償後的迴授訊號  $\hat{Y}(s)$  會存在一穩態誤差，而使用改善之 CDOB 架構後可消除此穩態誤差。然而迴授訊號有震盪情況之原因為 CDOB 之頻寬  $g_{net}$  受限於系統取樣頻率，對網路擾動之估測速度不足造成。

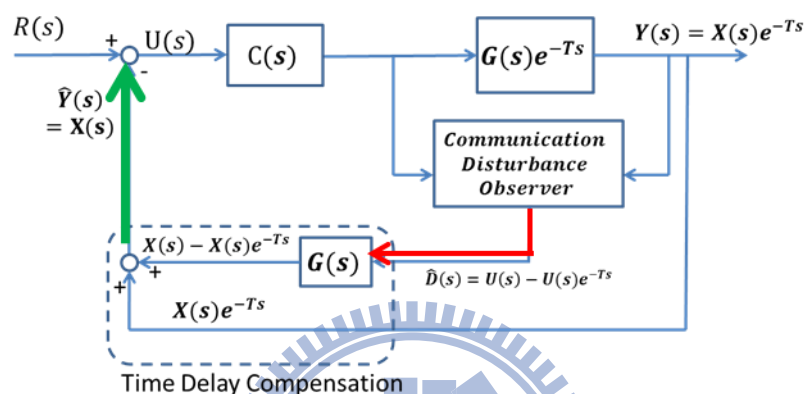


圖 3-17 CDOB 回授補償架構

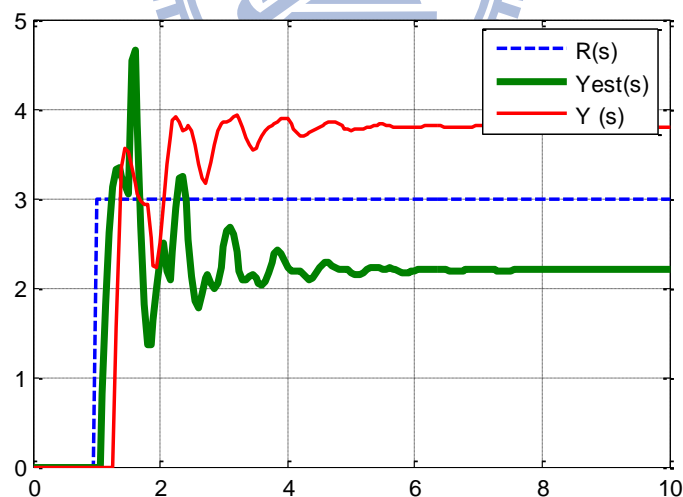


圖 3-18 CDOB 原始架構之估測迴授

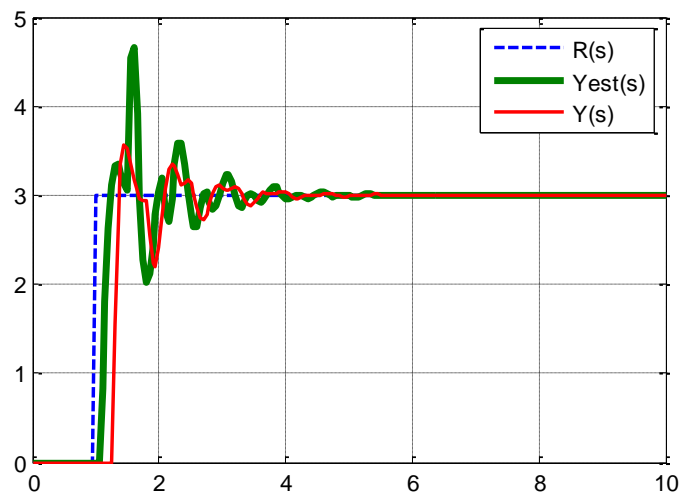


圖 3-19 CDOB 改善架構之估測迴授

### 3.6 實驗結果

本論文的實驗環境為透過 Ethernet，於交通大學內部網路系統做兩部電腦的連接，但由於交通大學的內部網路速度非常快，所以另外在兩個端點之間加入一緩衝區藉以凸顯網路對控制系統之影響，網路往返時間(RTT)之實際量測值如圖 3-20 所示。

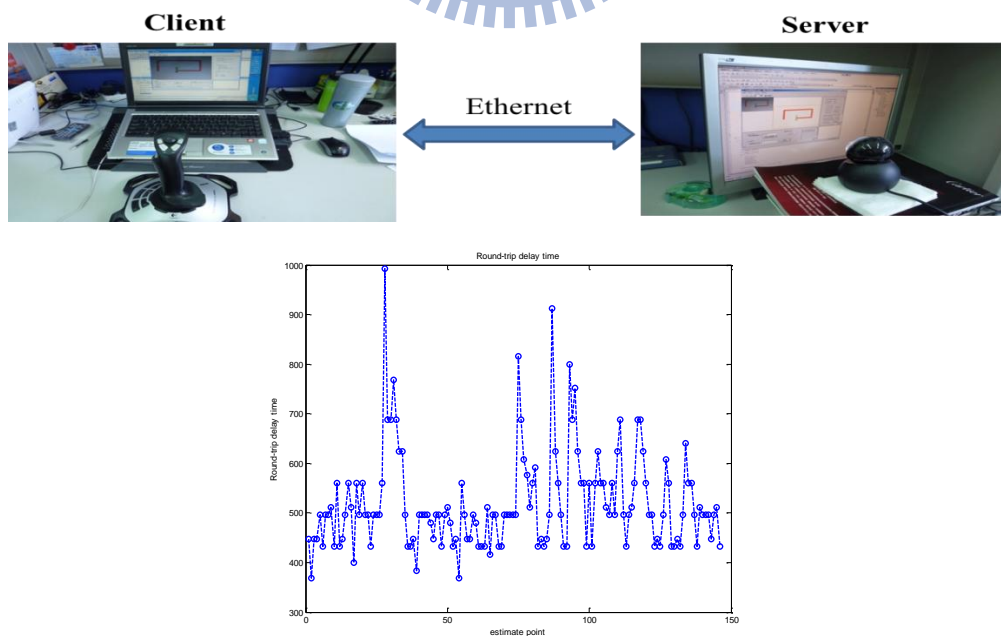


圖 3-20 Round-trip delay estimation

閉迴路控制系統如圖 3-21 所示

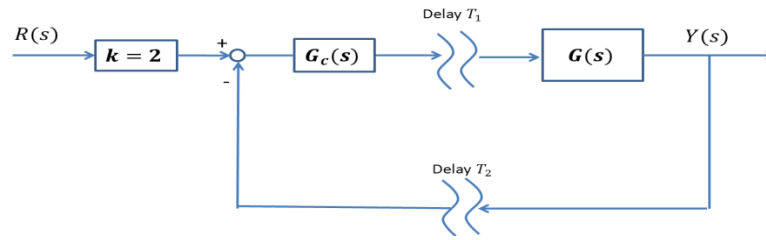


圖 3-21 閉迴路網路控制系統架構圖

其中受控體為一虛擬系統，其轉移函數  $G(s) = \frac{6}{s+6}$ 。前饋增益  $k = 2$  之目的為使響應收斂值與輸入訊號一致。

在未使用網際網路為傳輸媒介時，系統響應如圖 3-22 所示：

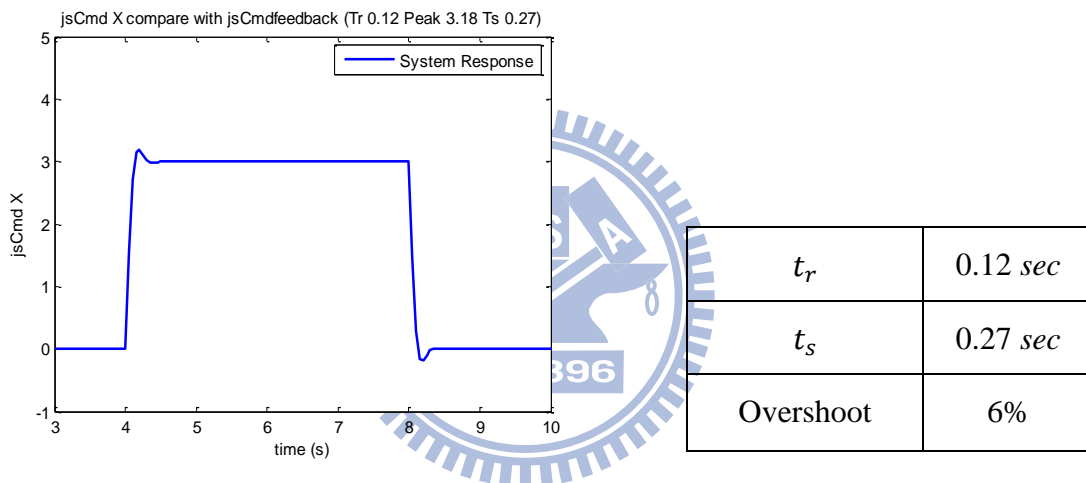


圖 3-22 未加入網路之系統響應

加入網際網路後，其響應如下圖 3-23 所示：

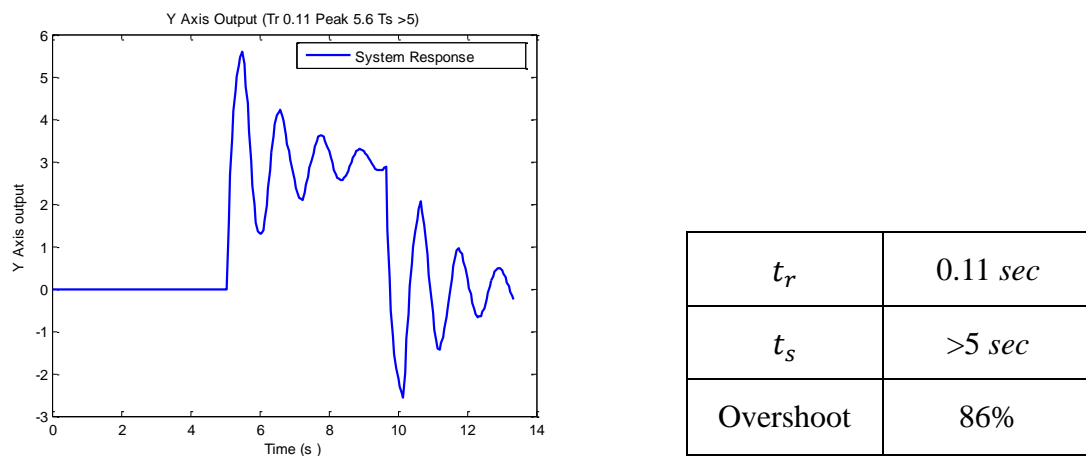


圖 3-23 加入網路後之系統響應

我們可以看到系統響應在加入網路後與預期結果差異甚大，主要原因為在閉迴路系統中的回授訊號誤差導致，所以在此加入通訊擾動估測器(圖 3-24)，發現振幅可大幅改善 41.8%，但收斂時間從約為五秒降為 4.66 秒，改善效果有限，並有-1.26 之穩態誤差，所以使用改善簡化後之 CDOB 架構進行實驗。

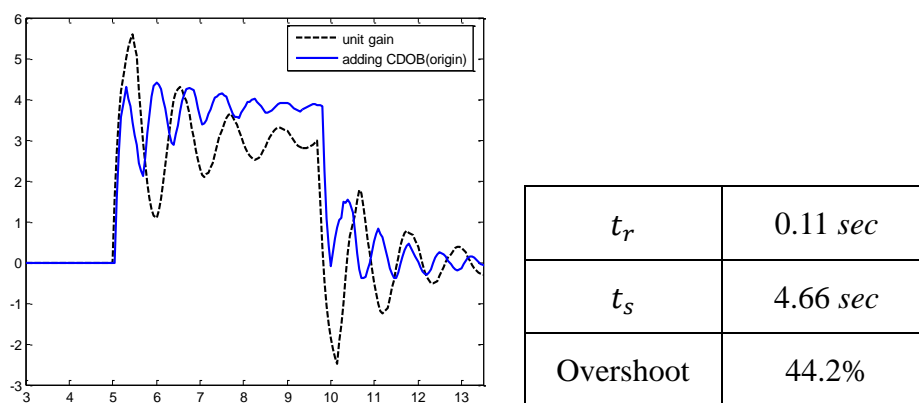


圖 3-24 加入 CDOB 之原始架構

下圖 3-25 為使用原始的 CDOB 架構與 CDOB 改善架構的實驗結果：

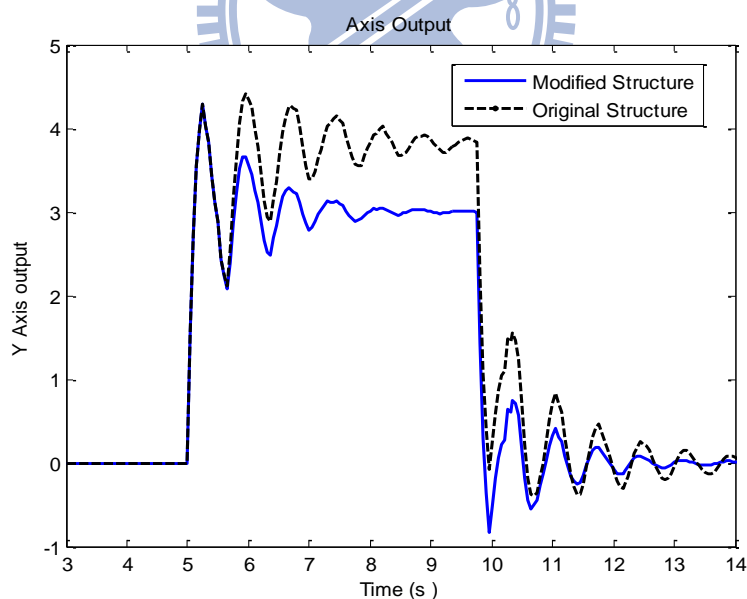


圖 3-25 使用原始 CDOB 架構與改善後架構之系統響應比較

圖 3-25 虛線部分為使用原始 CDOB 結構，與改良簡化後的結構比較可發現：使用改良簡化架構響應比使用原始架構收斂速度較快，並且沒有穩態誤差，誠如前敘，在系統取樣頻率不足時，原始架構內的積分項會由於無法與微分項相



消除而會造成一 offset，也就是穩態誤差發生的原因。

加入 CDOB 之效果由圖 3-26 可看出，CDOB 可在保持 rise time 的前提下大幅降低 settling time 與 overshoot。

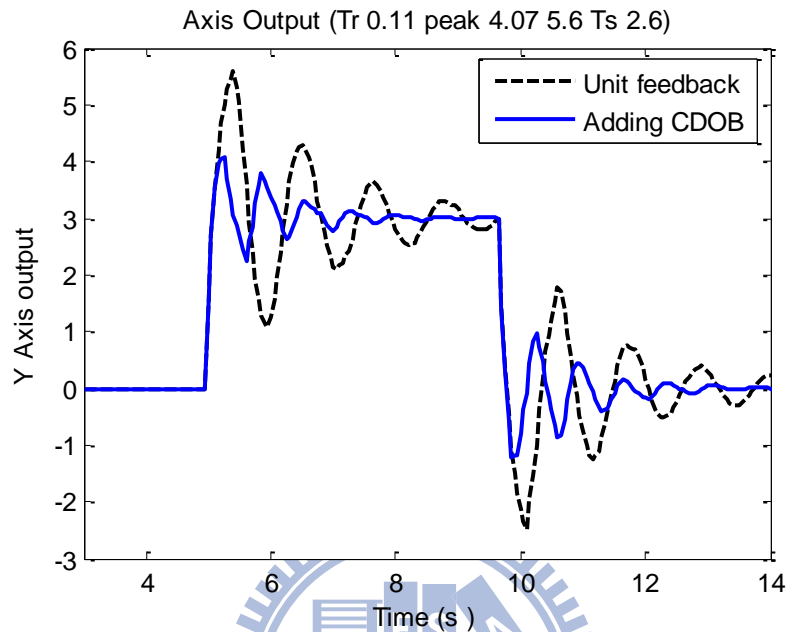


圖 3-26 網路控制系統 unit gain 與加入 CDOB 後響應比較

	Original system	Adding CDOB
$t_r$	0.11 sec	0.11 sec
$t_s$	>5 sec	2.6 sec
Overshoot	86%	35.6%

### 3.7 CDOB 與 adaptive Smith predictor 之比較

在此我們將通訊擾動估測器與 Smith predictor 與 adaptive Smith predictor[14]在不同的環境中作比較分析。圖 3-27 為 Smith predictor 架構，其概念主要為藉由網路延遲時間模型與受控體模型，結合控制訊號後與回授訊號，期望將網路延遲造成的影響抵消，CDOB 的延遲時間補償概念其實與此概念類似。而 adaptive Smith predictor 藉由即時量測 round-trip delay time(RTT)並即時更新 $e^{-T_m t}$ 補償項，

其中 $T_m$ 即為量測之 RTT。

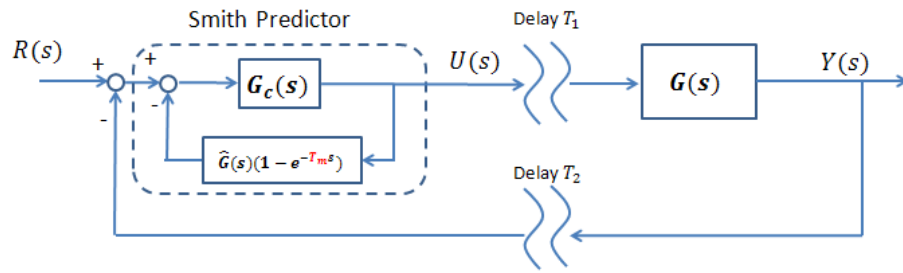


圖 3-27 Smith predictor 架構

考慮網路延遲變化劇烈的情況，設計情況之 RTT 測量值如圖 3-28 所示：

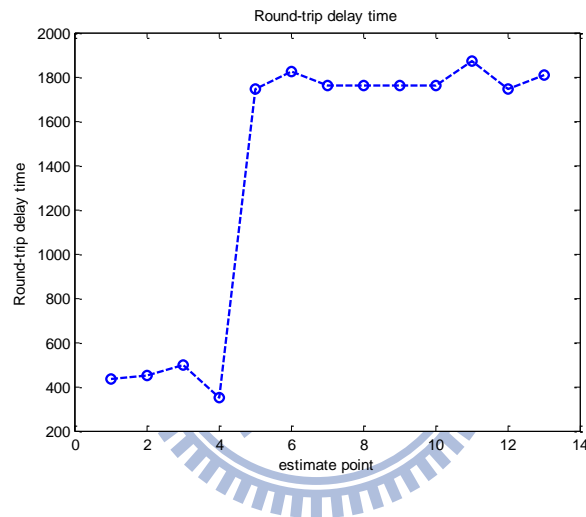


圖 3-28 網路延遲時間變動劇烈時之 RTT 量測值

在系統啟動後第十一秒時，延遲時間由大約 400 ms 劇增到 1800 ms，在此情況下，由於 Smith predictor 為基於網路延遲時間模型的控制架構，當此模型估測不準確時，響應結果就會有非常大的誤差，但由於 CDOB 為將網路延遲效應視為擾動，並且對其進行估測後回授補償，不需要網路延遲時間之模型，由實驗結果(圖 3-29)可看出使用 CDOB 之系統整體響應(實線)改變並不大：

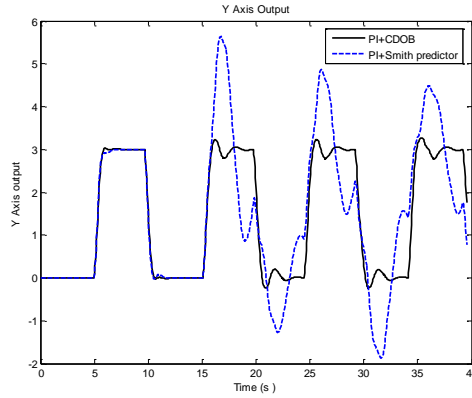


圖 3-29 CDOB 與 Smith predictor 於網路延遲變動劇烈時之響應比較

接下來考慮藉由 RTT 即時量測並即時更新 Smith predictor 時延遲間補償架構之 adaptive Smith predictor 與 CDOB 之實驗結果比較，由圖 3-30 可看出，adaptive Smith predictor 之補償效果依然不如 CDOB，其原因在於由於操作端在接收網路影像時，會切割擷取到的封包，因此會有資料遺失的情況，網路延遲時間越長，資料遺失機率越大，而 adaptive Smith predictor 的回授補償實現方式為，將控制訊號放進一緩衝區中，根據量測之 RTT 等待特定時間後，與迴授訊號相加以消除網路延遲對訊號之影響，但在資料遺失發生時，此方法即無法正確消除期望的網路延遲影響，所以造成迴授訊號誤差，進而影響系統響應；而由於 CDOB 是根據實際回授訊號與控制訊號進行估測，並不需要延遲時間的模型，所以依然能夠保持不錯的響應結果。

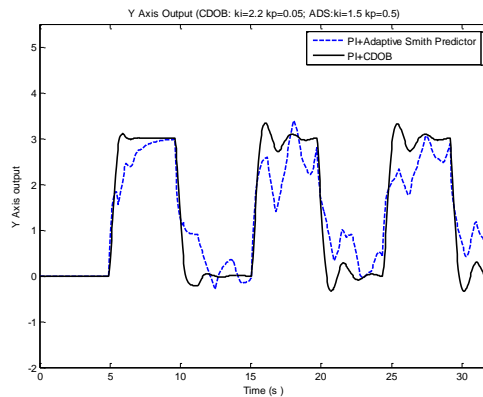


圖 3-30 CDOB 與 adaptive Smith predictor 於網路延遲變動劇烈時之響應比較

## 第四章 模型預測控制器

### 4.1 模型預測控制器(model predictive controller)基本概念

模型預測控制器(model predictive controller, MPC)之主要設計，其概念為首先將預定期望之系統響應與未來預測訊號長度，透過預測未來響應、最佳化指標與實際回授修正策略，即時計算出最符合預期響應的控制訊號。在此以一簡單的例子作 MPC 的比喻：

1. 我們可以想像有一工作團隊，目的為設計並實現出一個遠端操作控制系統，而我們的工作團隊的工作方法如下：在目標達成之前，在每個小時的開始時擬訂接下來四個小時的計畫藍圖，並執行第一個小時的工作內容。
2. 而訂定計畫的標準在於我們必須了解即將面臨何種問題，例如系統模型分析、設計、模擬與實現，這些都可以看成在訂定計畫時的操縱變數；另外，我們也必須將限制條件考慮進來，例如工作團隊對於系統模型的了解程度、設計的能力，與實現的能力，以上種種資訊在訂定計畫時皆為極重要的因素。
3. 掌握一切所需資訊後，工作團隊開始訂定接下來每個小時要做的工作，在考慮本身與系統條件限制的情況下，設計出最好的方法達成目標。計畫完成後即開始實行。
4. 第一個小時的工作結束後，工作團隊會審視這個小時的實際工作進度，由於計畫與實現往往存在著無法避免的誤差，種種因素猶如系統模型掌握不正確、工作成員的能力不足或低估，都會影響整體成果，所以在實作後蒐集這些資訊，加入接下來四個小時的計畫考慮因素中，重新設計後四個小時中每個小時的工作內容，如何以最大的效率完成計畫。接著進行下個小時的工作內容，一個小時後檢視成果並且設計接下來四個小時的策略，如此重複，

直到目標達成為止。

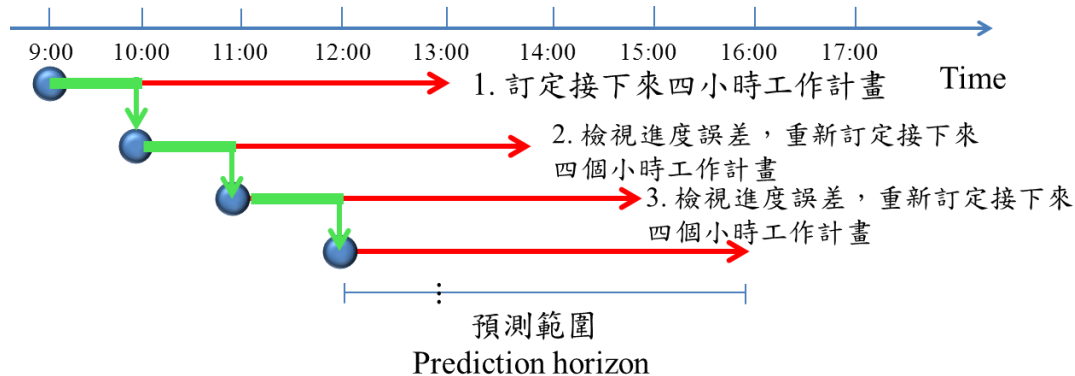


圖 4-1 MPC 設計概念示意圖

在整個計畫流程中有五項關鍵變數：

1. 預測範圍(prediction horizon)：決定將計畫未來幾個小時內的工作內容，在上述的例子中預測範圍為四小時。
2. 移動式預測窗口(moving horizon window)：由於在每個小時( $t_i$ )都會重新計畫接下來四個小時( $T_p=4$ )的工作內容，如此的動作猶如一個移動的窗口(從  $t_i$  到  $t_i + T_p$ )
3. 移動時域控制(receding horizon control)：雖然每次都會將四個小時的工作計畫訂定，但是工作團隊只會執行第一個小時的計畫內容，而其他三個小時的計畫不會接下去執行，此方法稱為移動時域控制。
4. 每個時間點的狀態(status)：在每個開始計畫的時間點都會有根據上個小時的工作成果獲得的資訊，這些資訊決定著接下來的設計限制與結果，所以工作團隊必須根據每個時間點的狀態決定接下來的工作計畫。
5. 設計出最佳工作計畫的準則：為了設計出最有效率的工作計畫，工作團隊必須根據目標訂定出評斷成果優劣的準則，此準則即為預期達到的目標與實際實行後結果之差距，而工作計畫的設計目標就是將此差距降為最低。

## 4.2 架構實現方法

由於 MPC 為 model-based 控制器，所以在此我們使用 state-space model 去表示一 single-input single-output 系統，如下式所示：

$$x_m(k+1) = A_m x_m(k) + B_m u(k) \quad (4-1)$$

$$y(k) = C_m x_m(k) \quad (4-2)$$

其中  $u$  為輸入， $y$  為輸出， $x_m$  為 state variable (dimension:  $n_1$ )。

對(4-1)式兩邊進行微分可得：

$$x_m(k+1) - x_m(k) = A_m[x_m(k) - x_m(k-1)] + B_m[u(k) - u(k-1)] \quad (4-3)$$

在此定義  $\Delta x_m(k) = x_m(k) - x_m(k-1)$ ； $\Delta u(k) = u(k) - u(k-1)$

上式(4-3)可以表示為：

$$\Delta x_m(k+1) = A_m \Delta x_m(k) + B_m \Delta u(k) \quad (4-4)$$

此時 state-space model 的表示式之輸入為  $\Delta u(k)$

由於 MPC 在決定新的 state 時必須將系統輸出納入考慮，所以定義一新的 state 變數  $x(k) = [\Delta x_m(k)^T \ y(k)]^T$ ，其中上標 T 代表矩陣的轉置(transpose)運算符號。

接著將式子(4-2)兩邊做相同的微分運算：

$$\begin{aligned} y(k+1) - y(k) &= C_m(x_m(k+1) - x_m(k)) = C_m \Delta x_m(k+1) \\ &= C_m A_m \Delta x_m(k) + C_m B_m \Delta u(k) \end{aligned} \quad (4-5)$$

將(4-4)與(4-5)整理在一起，可以得到系統新的 state-space model 表示式：

$$\begin{aligned} \begin{bmatrix} \Delta x_m(k+1) \\ y(k+1) \end{bmatrix} &= \begin{bmatrix} A_m & 0_m^T \\ C_m A_m & 1 \end{bmatrix} \begin{bmatrix} \Delta x_m(k) \\ y(k) \end{bmatrix} + \begin{bmatrix} B_m \\ C_m B_m \end{bmatrix} \Delta u(k) \\ &= \mathbf{A}x(k) + \mathbf{B}\Delta u(k) \end{aligned}$$

$$\begin{aligned}
y(k) &= [o_m \quad 1] \begin{bmatrix} \Delta x_m(k) \\ y(k) \end{bmatrix} \\
&= Cx(k)
\end{aligned} \tag{4-6}$$

其中  $o_m = \overbrace{[0 \ 0 \ 0 \ \dots \ 0]}^{n_1}$

將系統的 state-space model 轉換為符合 MPC 需求的表示方法後，現在要做的是預測出系統未來的輸出響應，要預測出未來的系統響應，首先必須先決定未來的控制訊號，這些控制訊號即為可調整的變數。由於 MPC 在每個時間點都會對未來控制訊號與系統響應進行估測，在離散系統中即為每個取樣時間發生的時刻。在此定義目前取樣的時間點為  $k_i$ ， $k_i > 0$ ，用來進行最佳化運算的響應預測窗口長度為  $N_p$ ，並假設在此刻系統的狀態變數  $x(k_i)$  是可以被量測到的。而我們期望得到的最佳控制訊號串：

$$\Delta u(k_i), \Delta u(k_i + 1), \dots, \Delta u(k_i + N_c - 1)$$

$N_c$  為期望預測的控制訊號長度(control horizon)。

而上述提到之  $N_p$ ，為由目前狀態  $x(k_i)$  預測未來狀態的訊號長度(prediction horizon)，如下式表示：

$$x(k_i + 1|k_i), x(k_i + 2|k_i), x(k_i + 3|k_i), \dots, x(k_i + m|k_i), \dots, x(k_i + N_p|k_i)$$

一般而言， $N_c$  會選擇小於或等於  $N_p$ 。

從系統的 state-space model (A,B,C)，可以將未來的狀態一一算出：

$$x(k_i + 1|k_i) = Ax(k_i) + B\Delta u(k_i) \tag{4-7}$$

$$x(k_i + 2|k_i) = Ax(k_i + 1|k_i) + B\Delta u(k_i + 1)$$

$$= A^2x(k_i) + AB\Delta u(k_i) + B\Delta u(k_i + 1)$$

$$x(k_i + 3|k_i) = Ax(k_i + 2|k_i) + B\Delta u(k_i + 2)$$

$$= A^3x(k_i) + A^2B\Delta u(k_i) + AB\Delta u(k_i + 1) + B\Delta u(k_i + 2)$$

⋮



$$\begin{aligned}
x(k_i + N_p | k_i) &= A^{N_p} x(k_i) + A^{N_p-1} B \Delta u(k_i) + A^{N_p-2} B \Delta u(k_i + 1) + \cdots + \\
&\quad A^{N_p-N_c} B \Delta u(k_i + N_c - 1)
\end{aligned} \tag{4-8}$$

有了預測的 state 後，我們就可以得到預測的系統輸出響應：

$$\begin{aligned}
y(k_i + 1 | k_i) &= CAx(k_i) + CB\Delta u(k_i) \\
y(k_i + 2 | k_i) &= CA^2x(k_i) + CAB\Delta u(k_i) + CB\Delta u(k_i + 1) \\
y(k_i + 3 | k_i) &= CA^3x(k_i) + CA^2B\Delta u(k_i) + CAB\Delta u(k_i + 1) + CB\Delta u(k_i + 2) \\
&\vdots \\
y(k_i + N_p | k_i) &= CA^{N_p}x(k_i) + CA^{N_p-1}B\Delta u(k_i) + CA^{N_p-2}B\Delta u(k_i + 1) + \cdots + \\
&\quad CA^{N_p-N_c}B\Delta u(k_i + N_c - 1)
\end{aligned} \tag{4-9}$$

值得注意的是，式(4-9)中，預測訊號的輸入訊號皆為 $x(k_i)$ 與 $\Delta u(k_i)$ ，所以在此我們再定義Y 與 $\Delta U$ ：

$$\begin{aligned}
Y &= [y(k_i + 1 | k_i) \ y(k_i + 2 | k_i) \ y(k_i + 3 | k_i) \ \cdots \ y(k_i + N_p | k_i)]^T \\
\Delta U &= [\Delta u(k_i) \ \Delta u(k_i + 1) \ \Delta u(k_i + 2) \ \cdots \ \Delta u(k_i + N_c - 1)]^T
\end{aligned}$$

此時便可將(4-9)整理如下：

$$Y = \mathbf{F} x(k_i) + \mathbf{\Phi} \Delta U \tag{4-10}$$

其中：

$$\begin{aligned}
\mathbf{F} &= \begin{bmatrix} CA \\ CA^2 \\ CA^3 \\ \vdots \\ CA^{N_p} \end{bmatrix} \\
\mathbf{\Phi} &= \begin{bmatrix} CB & 0 & 0 & \cdots & 0 \\ CAB & CB & 0 & \cdots & 0 \\ CA^2B & CAB & CB & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ CA^{N_p-1}B & CA^{N_p-2}B & CA^{N_p-3}B & \cdots & CA^{N_p-N_c}B \end{bmatrix}
\end{aligned}$$

接下來的目標為如何決定最佳控制訊號的方法。考慮一給定輸入訊號，並且在預測範圍中為保持不變的常數  $r(k_i)$ ，而此輸入訊號與系統預測的響應可構成一差值函數(error function)，藉由將此函式最小化(minimizing)，可計算出控制訊號列  $\Delta U$ ，此時計算出的結果即為在此定義的最佳控制訊號列。

令  $R_s^T$  為包含著輸入訊號資訊的資料向量：

$$R_s^T = \overbrace{[1 \ 1 \ \dots \ 1]}^{N_p} r(k_i)$$

定義系統之成本函數(cost function)  $J$ ：

$$J = (R_s - Y)^T (R_s - Y) + \Delta U^T \bar{R} \Delta U \quad (4-11)$$

其中  $\bar{R} = r_w I_{N_c \times N_c} (r_w \geq 0)$

顯而易見的，成本函數  $J$  的前半部  $(R_s - Y)^T (R_s - Y)$ ，為輸入訊號與輸出訊號的差值；後半部  $\Delta U^T \bar{R} \Delta U$ ，則反映了是否將  $\Delta U$  的大小納入考慮，藉由調整  $r_w$ ，可改變  $\Delta U$  在成本函數中佔的比重。舉例來說，如果  $r_w = 0$ ，則我們要考慮的就是如何算出控制訊號，讓系統輸出與輸入的差值在最短時間內降至最低；然而當  $r_w > 0$  時，代表著必須將控制訊號  $\Delta U$  的大小考慮進來，此時的狀況就變成不僅只要使輸出與輸入響應差值降低，還必須考慮到  $\Delta U$  的變化，感覺就像是我們必須謹慎地將許多因素考慮進來，以達到期望的目的。關於  $r_w$  的影響在接下來的模擬與實作中會再次驗證。

接著繼續介紹求出期望控制訊號列  $\Delta U$  之公式推導，將(4-10)代入(4-11)可得：

$$J = (R_s - Fx(k_i))^T (R_s - Fx(k_i)) - 2\Delta U^T \Phi^T (R_s - Fx(k_i)) + \Delta U^T (\Phi^T \Phi + \bar{R}) \Delta U \quad (4-12)$$

由成本函數  $J$  對  $\Delta U$  做一階微分：

$$\frac{\partial J}{\partial \Delta U} = -2\Phi^T (R_s - Fx(k_i)) + \Delta U^T (\Phi^T \Phi + \bar{R}) \Delta U \quad (4-13)$$

由於我們期望此成本函數降至最低，所以令：

$$\frac{\partial J}{\partial \Delta U} = 0$$

由此條件與(4-13)式，最後我們可以得到最佳的控制訊號列 $\Delta U$ 為：

$$\Delta U = (\Phi^T \Phi + \bar{R})^{-1} \Phi^T (\bar{R}_s r(k_i) - Fx(k_i)) \quad (4-14)$$

$$\text{其中 } \bar{R}_s = \overbrace{[1 \ 1 \ 1 \cdots 1]^T}^{N_p}$$

在離散系統中的每個取樣時間時，皆會算出一組 $\Delta U$ ，其中包含著 $N_c$ 個控制訊號的變化量 $(\Delta u(k_i) \ \Delta u(k_i + 1) \ \Delta u(k_i + 2) \ \cdots \ \Delta u(k_i + N_c - 1))$ ，但我們只將第一個 $\Delta u(k_i)$ 拿來使用，誠如上述計畫工作的案例所敘，每個時間點皆執行第一個小時的工作後，檢視成果與重新訂定計畫，此方法稱為移動時域控制(receding horizon control)。而由(3-14)式可觀察到， $(\Phi^T \Phi + \bar{R})^{-1} \Phi^T \bar{R}_s r(k_i)$  項中，因為 $\Phi$ 為只與系統 model 特性(C,A,B)有關之矩陣，所以如果輸入訊號為固定時，此項即為非時變矩陣； $-(\Phi^T \Phi + \bar{R})^{-1} Fx(k_i)$  項則因為 $F$ 為只與系統 model (C,A)有關之矩陣，所以 $-(\Phi^T \Phi + \bar{R})^{-1} F$ 亦為非時變矩陣，非時變矩陣的好處在於我們可以先將其結果求出，如此在實作時就不必在每個取樣時間都重新計算一次。

基於移動時域控制法則(RHC)，我們只執行第一個控制訊號，所以我們將 $\Delta U$ 的第一個元素取出：

$$\begin{aligned} \Delta u(k_i) &= \overbrace{[1 \ 0 \ 0 \cdots 0]}^{N_p} (\Phi^T \Phi + \bar{R})^{-1} \Phi^T (\bar{R}_s r(k_i) - Fx(k_i)) \\ &= K_y r(k_i) - K_{mpc} x(k_i) \end{aligned} \quad (4-15)$$

其中 $K_y$ 為 $(\Phi^T \Phi + \bar{R})^{-1} \Phi^T \bar{R}_s$ 的第一個元素， $K_{mpc}$ 為 $(\Phi^T \Phi + \bar{R})^{-1} F$ 矩陣的第一列。值得注意的是，式(3-15)其實為一閉迴路控制系統的標準表示式，新的控制訊號由輸入訊號減去回授訊號得到，而回授訊號之增益為 $K_{mpc}$ 。

將(4-15)代回(4-7)中，可得：

$$\begin{aligned} x(k_i + 1|k_i) &= Ax(k_i) - BK_{mpc}x(k_i) + BK_y r(k_i) \\ &= (A - BK_{mpc})x(k_i) + BK_y r(k_i) \end{aligned} \quad (4-16)$$



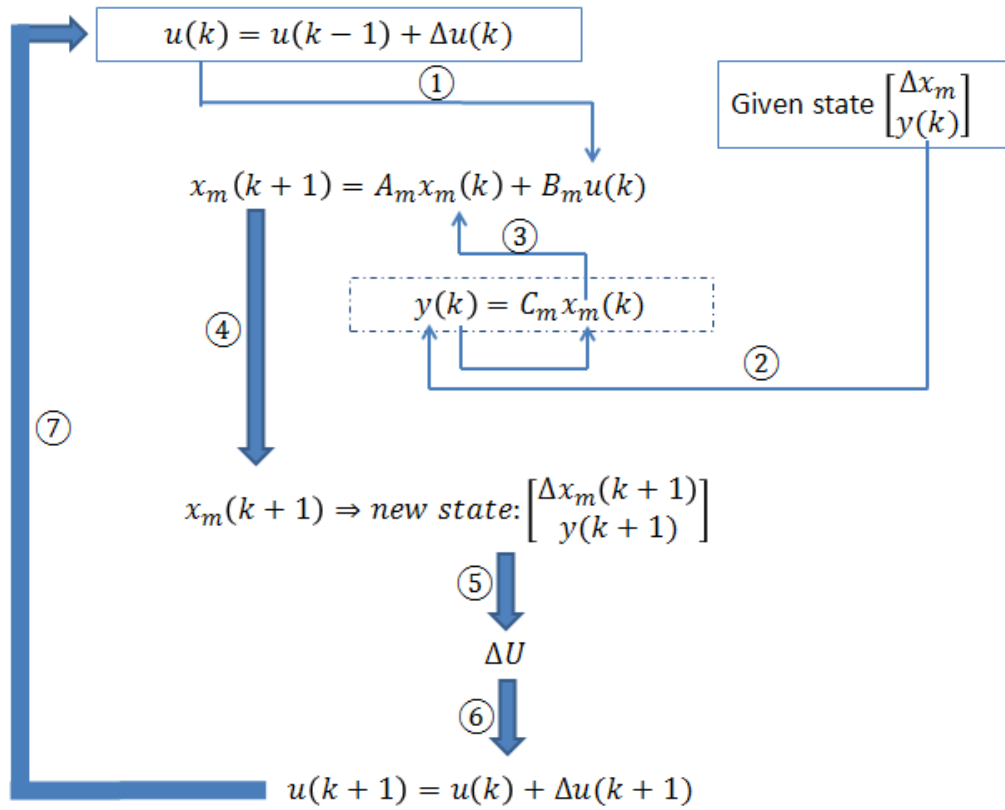


圖 4-3 MPC 控制訊號計算方法實現流程圖

- ①：首先我們必須得到系統初始之控制訊號  $u(k)$ ，與狀態量測值
- ②：有了目前的狀態後，我們可以得到  $y(k)$ ，由(3-2)反推可得到  $x_m(k)$
- ③：將  $x_m(k)$  與控制訊號  $u(k)$ ，由式(3-1)可算出新的系統狀態  $x_m(k+1)$
- ④：由前一個 state 與新算出的  $x_m(k+1)$ ，可得  $\Delta x_m(k+1)$  與  $y(k+1)$
- ⑤：由新的系統狀態，可對新的控制訊號  $\Delta U$  進行一定長度  $(N_p)$  的預估計算
- ⑥：但是我們只拿出第一個控制訊號變量  $\Delta u(k+1|k)$
- ⑦：計算出新的控制訊號後，重複此流程

### 4.3 整合 MPC 與 CDOB 估測輸出之實驗結果

首先由(4-11)：成本函數  $J = (R_s - Y)^T (R_s - Y) + \Delta U^T r_w I_{N_c \times N_c} \Delta U$  中得知， $r_w$  對  $\Delta U$  之計算結果佔有決定性的影響， $r_w$  越小，計算出來的  $\Delta U$  則變動越劇烈，

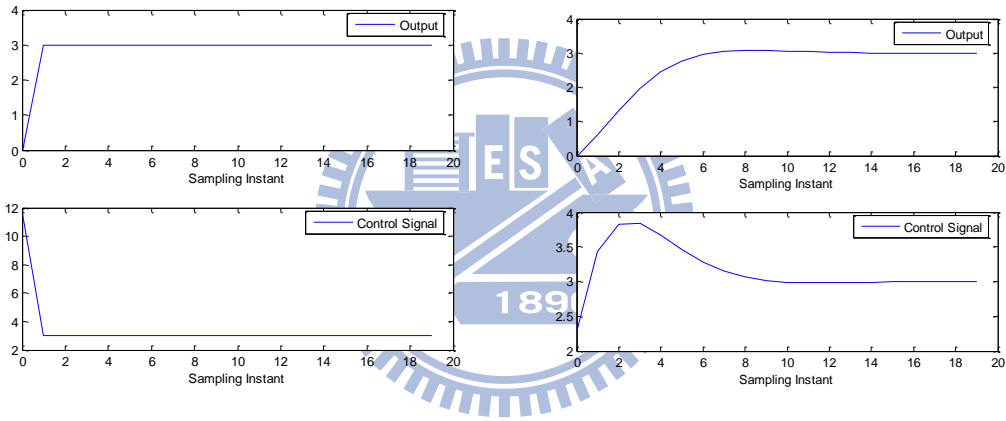
系統響應就會越快，反之亦然。由模擬結果(圖 4-4)可以看出當 $r_w = 0$ 時，系統響應可以在一個取樣時間就到達，隨著 $r_w$ 增加，系統響應的速度隨之減小，因為當 $r_w$ 增加時，代表成本函數中 $\Delta U$ 所佔比例增加，此時即不僅只考慮到響應與輸入訊號之間的誤差，還必須將計算出之 $\Delta U$ 的變化納入考量，如此一來算出的控制訊號就不會像在 $r_w = 0$ 時理想化，反而是小心地(cautiously)增加。

模擬條件：

$$\text{System transfer function : } G(s) = \frac{6}{s+6}$$

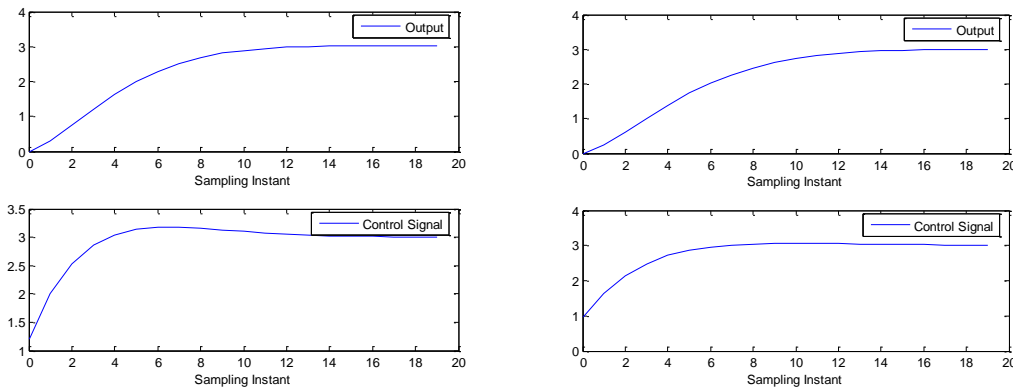
預測窗口長度： $N_p = 20$ ；預測控制訊號長度： $N_c = 4$

Reference input： $R(s) = 3$



(a)  $r_w = 0$

(b)  $r_w = 1$



(c)  $r_w = 5$

(d)  $r_w = 10$

圖 4-4 不同 $r_w$ 之模擬結果比較

接著，將 MPC 運用於網路控制系統中，由圖 4-5 與圖 4-6 中可發現，加入

網路後系統因為輸出迴授延遲造成系統劇烈震盪，因此 MPC 需要網路估測器以獲得未經過網路延遲之系統輸出迴授。

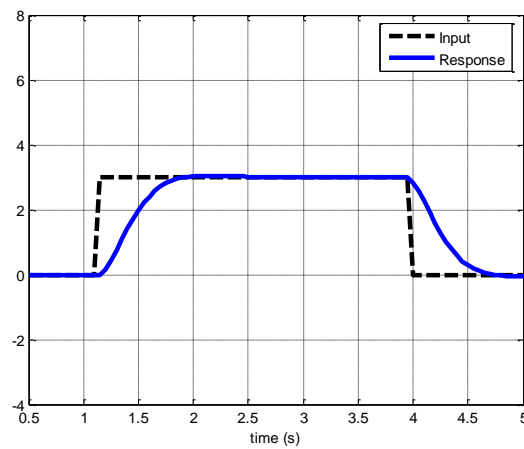


圖 4-5 未加入網路

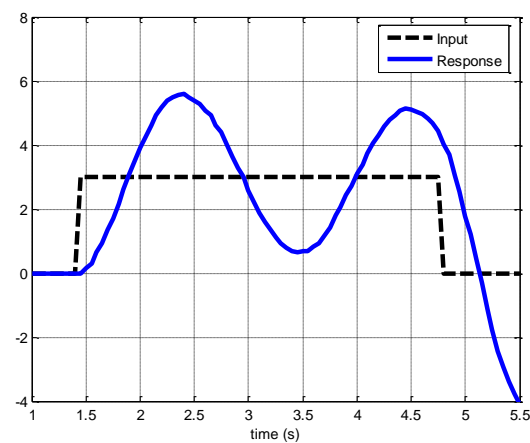


圖 4-6 加入網路，未使用估測器

所以在此分別使用 adaptive Smith predictor 與前章介紹之 CDOB 做為系統之估測器，並在如圖 4-7 所示網路延遲時間劇烈變動時進行實驗。

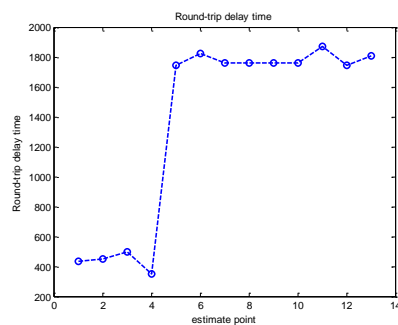


圖 4-7 網路延遲時間變動劇烈時之 RTT 量測值



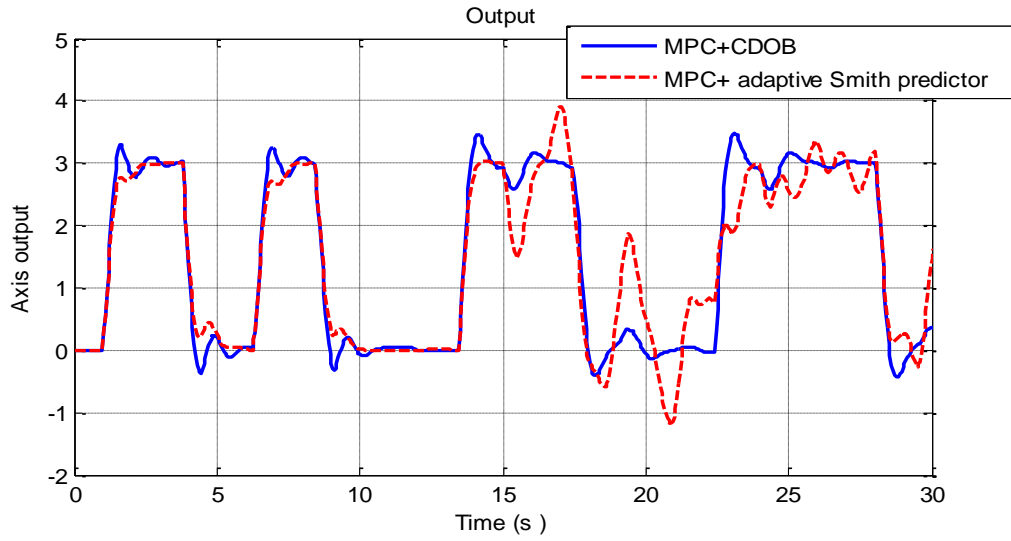


圖 4-8 分別使用 adaptive Smith predictor 與 CDOB 做為系統估測器於延遲時間劇烈變動下之系統響應

由實驗結果圖 4-8 可看出，在經過第十一秒，網路延遲時間變動至約 1800 ms 時，使用 adaptive Smith predictor 為估測器之系統之響應遠不如使用 CDOB 穩定，原因即為前述，由於 adaptive Smith predictor 在資料遺失的情況下無法正確地消除延遲造成的迴授訊號影響，所以迴授訊號之誤差依然存在，進而影響系統響應。所以本研究使用 MPC 與 CDOB 做為控制架構之控制器與估測器，控制架構如下

圖 4-9 所示：

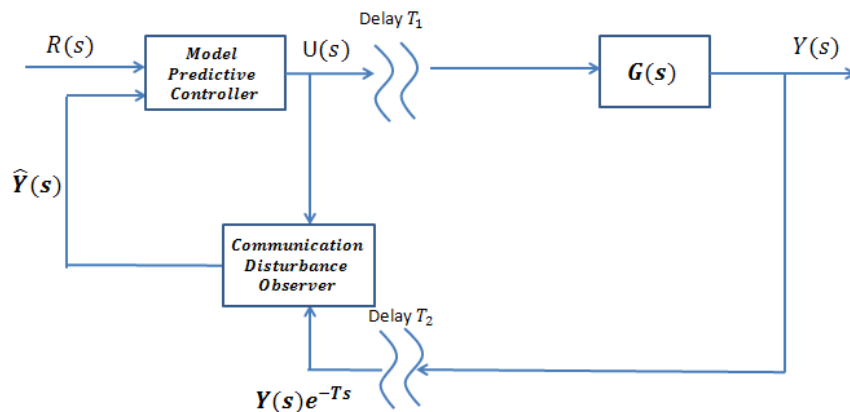
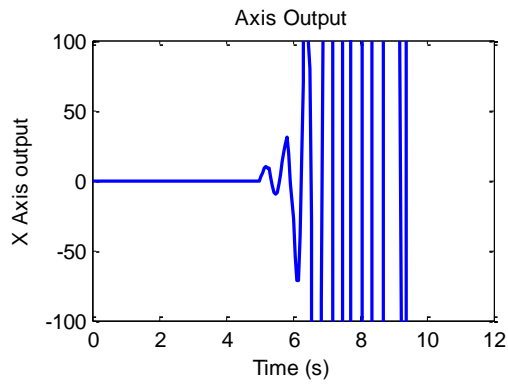


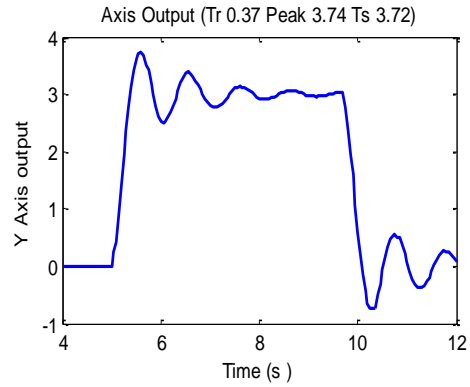
圖 4-9 MPC+CDOB 控制架構示意圖

再次，針對不同  $r_w$  進行實驗結果比較(實驗條件與 3.5 節相同)：



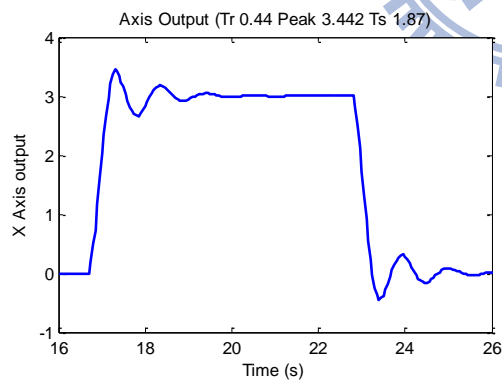
(a)  $r_w = 1$

$t_r$	unstable
$t_s$	unstable
Overshoot	unstable



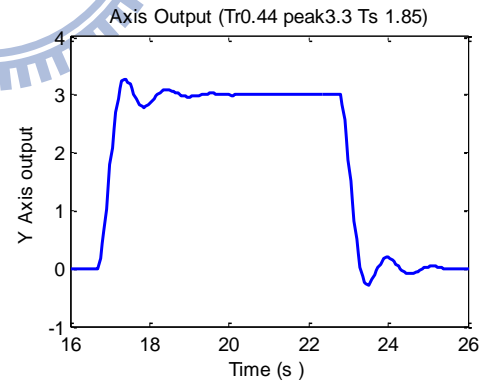
(b)  $r_w = 10$

$t_r$	0.37 sec
$t_s$	3.72 sec
Overshoot	24.6%



(c)  $r_w = 20$

$t_r$	0.44 sec
$t_s$	1.87 sec
Overshoot	14.7%



(d)  $r_w = 30$

$t_r$	0.44 sec
$t_s$	1.85 sec
Overshoot	10%

圖 4-10 不同 $r_w$ 於網路控制系統之實驗結果比較

由實驗結果(圖 4-10)可以觀察出， $r_w$  對響應的變化趨勢與模擬結果相同。 $r_w$  越小時 rise time 越小，但 overshoot 越大，系統也越趨近於不穩定。而當  $r_w = 1$  時系統不穩定的原因在於：對於每個系統而言，都有其極限所在，當我們將控制器設定在想要讓系統的響應極快，但實際上系統卻無法及時的追上命令時，就會造成系統震盪，加上在網路控制系統中，不確定的因素造成建模誤差，進而影響控制器的效能，很容易就會讓系統進入不穩定的狀態。由實驗結果可看到  $r_w = 20$  與  $r_w = 30$  之 rise time 相同，但又更穩定，所以在此我們設定  $r_w = 30$ ，並與未加入控制器之系統響應做比較：

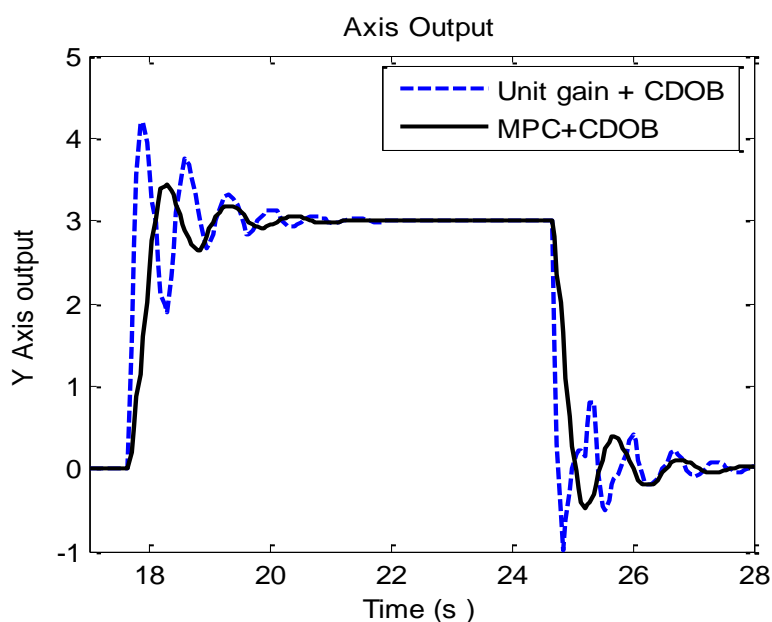


圖 4-11 CDOB 與 MPC+CDOB 響應比較

	Unit gain + CDOB	MPC + CDOB
$t_r$	0.11 sec	0.44 sec
$t_s$	2.6 sec	1.85 sec
Overshoot	35.6%	10%

由實驗結果(圖 4-11)可看出，加入 MPC 後可讓系統在 rise time 仍然在容忍範圍之前提下，提高系統之穩定度。

## 4.4 MPC 與 PI 控制器於網路控制系統之比較

由於 PI 控制器(proportional-integral controller)的設計與實現方法簡單並且具有一定的控制效能，所以在一般的網路控制系統中經常被使用。在接下來的實驗中，皆使用 CDOB 做為系統的估測器，針對不同控制器在網路控制系統中的效能進行比較與討論。

首先考慮未加入網路時，將 MPC 與 PI 控制器參數調整至響應一致：

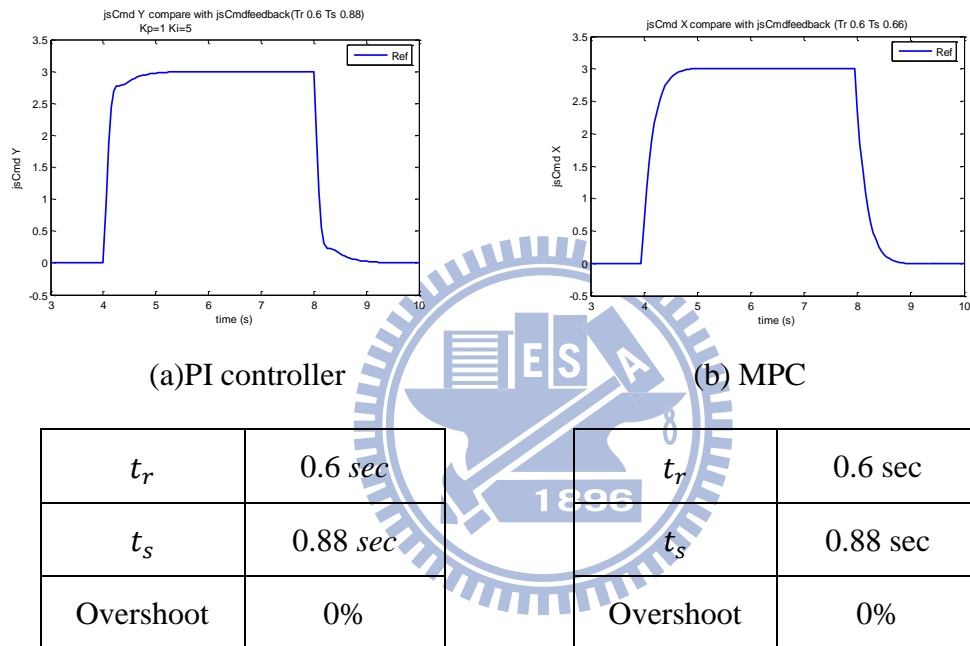
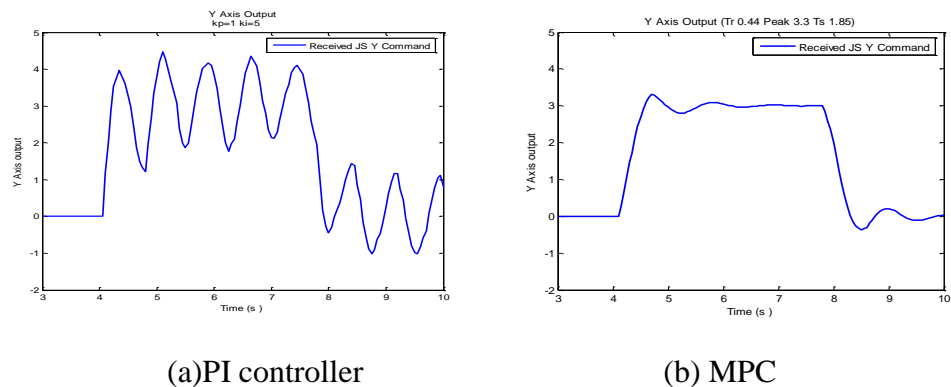


圖 4-12 未加入網路時之 MPC 與 PI 控制器響應比較

接著加入網路，並皆使用相同參數之 CDOB 做為系統估測器：



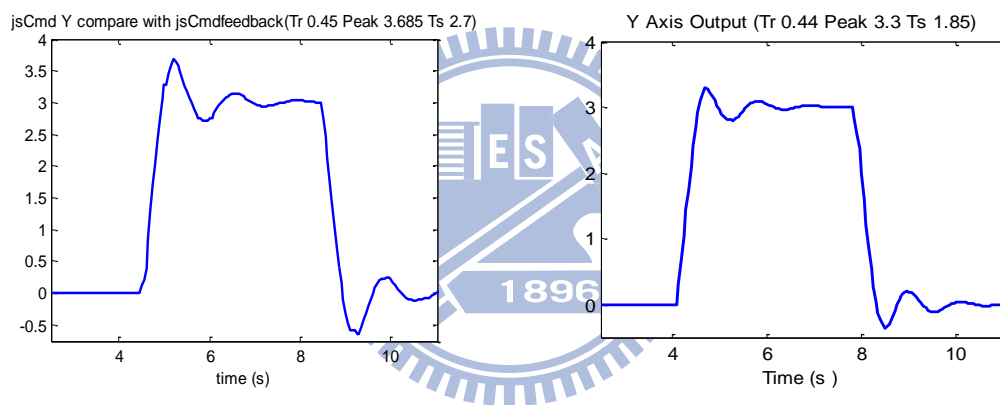
$t_r$	0.15 sec
$t_s$	>4 sec
Overshoot	45%

$t_r$	0.73 sec
$t_s$	1.25 sec
Overshoot	7%

圖 4-13 加入網路且參數不變時之系統響應比較

由實驗結果可以看出，在參數未改變的情況下，MPC 相對於 PI 控制器具有較佳的應變能力，此實驗也驗證了 1.4 節中提到 MPC 比一般控制器更能夠克服複雜且不定的受控體對期望響應造成的影響。

為了再進一步比較 MPC 與 PI 控制器的效能，此時調整 PI 控制器參數，使響應之 rise time 皆為 0.44 s 左右，以比較其他效能的表現：



(a) PI+CDOB

(b) MPC+CDOB

$t_r$	0.44 sec
$t_s$	2.7 sec
Overshoot	22.8%

$t_r$	0.44 sec
$t_s$	1.85 sec
Overshoot	10%

圖 4-14 Rise time 相同時之響應比較

實驗結果再次證明了 MPC 在網路控制系統中具有較佳的響應，原因為由於 MPC 利用已知受控體模型與預測未來響應、最佳化指標與實際回授修正策略，即時計算出最符合預期的控制訊號，所以在系統受到其他因素影響時更能夠比 PI 控制器克服網路延遲造成的影響。

## 第五章 估測迴授訊號之權重分配機制

### 5.1 MPC + CDOB + (Feedback Weighting Mechanism)

本論文提出一網路控制系統架構，目的為抑制因為網路延遲造成遠端操作系統不穩定之影響，並且期望藉由改良系統響應，提高遠端操作系統穩定性與透明性。此控制架構如圖 5-1 所示。

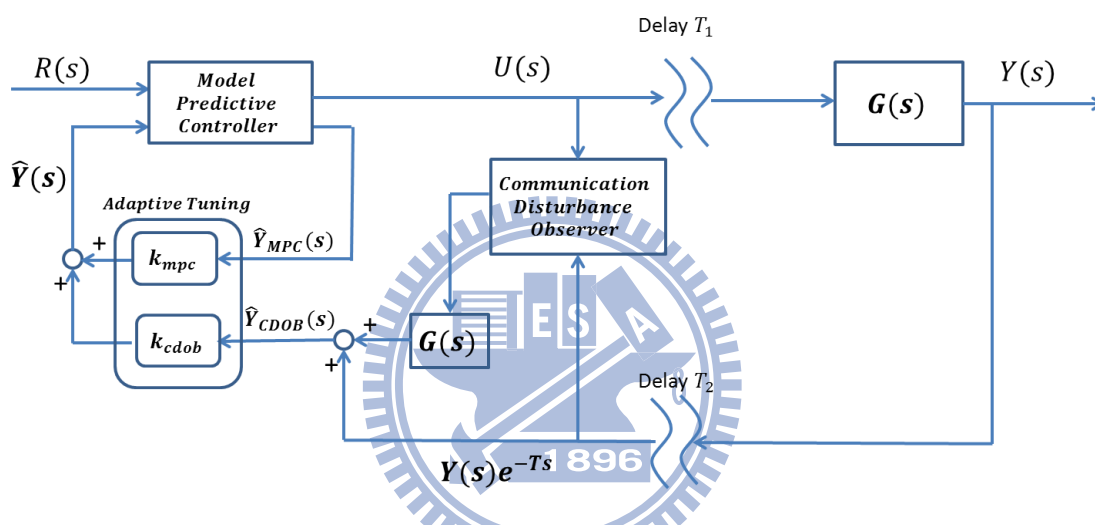


圖 5-1 MPC + CDOB + (feedback weighting mechanism) 之控制架構

本論文所提出之整合式網路控制系統架構可分為三個部分：

#### 1. 網路通訊擾動估測器 (CDOB)

此為一訊號估測器，藉由將網路延遲效應當作系統的擾動現象，並進行估測，最後將估測的擾動值與實際經過延遲的響應相消除，期望只留下未因為網路延遲影響的回授訊號。而由於本系統使用之影像傳輸陣列龐大，使 CDOB 於取樣頻率不足時發生估測值之穩態誤差，所以提出一改善架構，使 CDOB 於取樣頻率不足時亦能運作正常，消除其穩態誤差。

## 2. 模型預測控制器(MPC)

利用已知受控體模型、設計最佳化指標與實際回授修正策略，即時計算出最符合預期的控制訊號並對未來響應進行估測。應用於本操作系統中，可大幅提升系統穩定度，並且讓響應速度仍保持在容忍範圍內。而 MPC 同時亦會對系統理想之輸出迴授進行估測，在此可提供系統回授訊號權重機制之依據。

## 3. 估測迴授訊號之權重分配機制(feedback weighting mechanism)

由於系統之中斷速度受限於影像傳輸陣列大小，進而影響訊息估測器之速度，在此，將 MPC 與 CDOB 估測的訊息作權重之分配後，當作最後的回授訊號，並且在資料遺失的情況發生時，將權重調整為僅使用 MPC 估測之響應當作回授訊號。

對於權重分配參數之決定，在此採用手動調整，將實驗結果列出後比較，選擇具有最佳響應之參數，實驗數據如下表所示：

表格 5-1 權重分配參數調整

	Rise Time	Peak Value	Settling Time
$k_{cdob} = 0 \quad k_{mpc} = 1$	0.85 s	3.00	0.95 s
$k_{cdob} = 0.3 \quad k_{mpc} = 0.7$	0.58 s	3.05	0.85 s
$k_{cdob} = 0.5 \quad k_{mpc} = 0.5$	0.57 s	3.10	1.30 s
$k_{cdob} = 0.7 \quad k_{mpc} = 0.3$	0.48 s	3.17	1.37 s
$k_{cdob} = 1 \quad k_{mpc} = 0$	0.46 s	3.31	1.85 s

由此實驗數據表中可發現，不同參數下的 rise time 與 peak value 並無太大差距，但在 settling time 的表現上，不同參數之差異便較明顯，所以在此我們選擇



在 rise time 容忍範圍中，settling time 與 peak value 最小之參數：

$$k_{cdob} = 0.3 \quad k_{mpc} = 0.7$$

接著，考慮在延遲時間 450 ms 內不同的延遲時間下，不同參數之實驗結果比較：

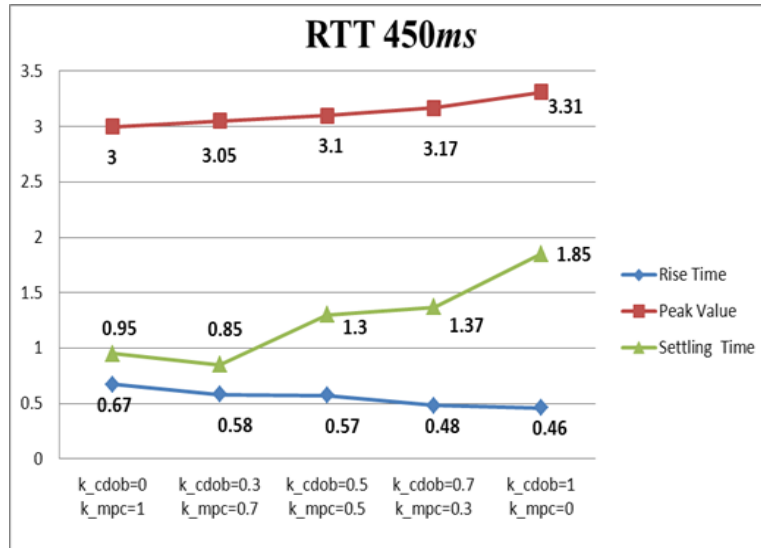


圖 5-2 RTT 為 450 ms 下之各參數響應比較

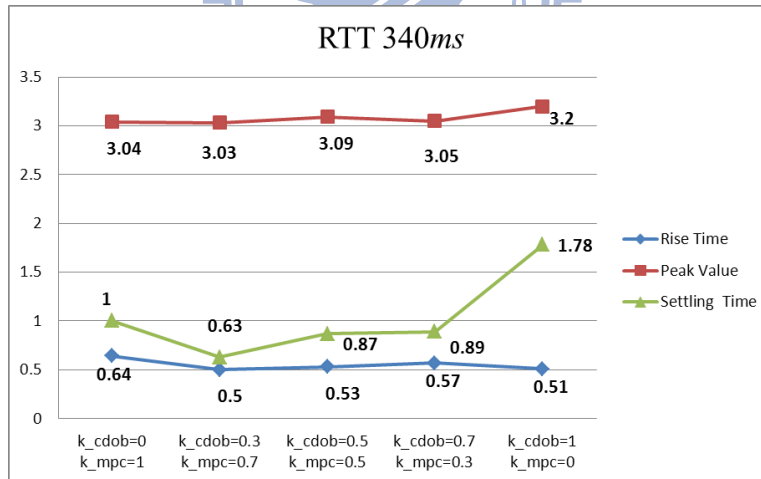


圖 5-3 RTT 為 340 ms 下之各參數響應比較

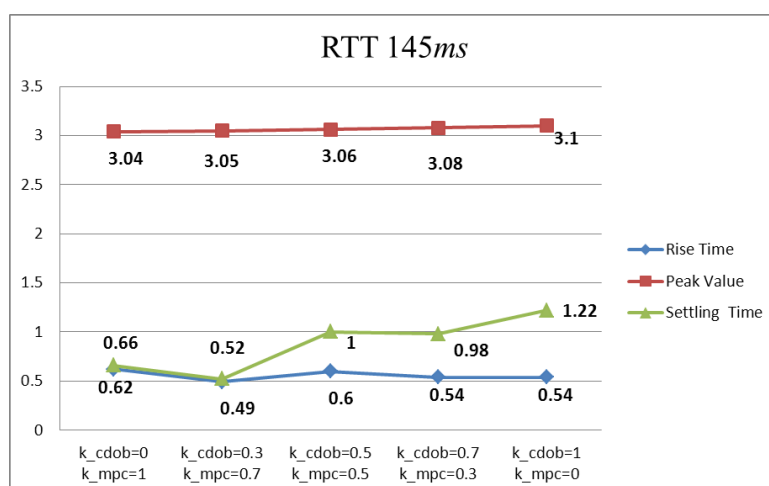


圖 5-4 RTT 為 145 ms 下之各參數響應比較

由實驗結果比較數據可歸納出幾個結論：

1. 使用 MPC 估測之迴授，此時為開迴路架構，所以系統響應較慢且穩定；使用 CDOB 估測之迴授，響應速度變快，但由於頻寬對其性能之限制，估測值存在其一定誤差，使系統較不穩定。
2. CDOB 迴後訊號比重增加時，系統之 settling time 與 peak value 隨之增加，系統越趨不穩定。
3. 由於在不同參數下之 peak value 與 rise time 之變化皆不如 settling time 之改變幅度，故在 rise time 之容忍範圍內，選擇 settling time 最小之參數，即  $k_{cdob} = 0.3$   $k_{mpc} = 0.7$ ，由此可驗證此組參數在延遲時間 450 ms 內皆具有最穩定之系統響應。

最後，加入在回授訊號資料遺失時之參數調整機制。由於在本實驗中操作端接收影像陣列時，會在接收時切割封包進而造成資料遺失，所以回授訊號之資料遺失率會遠大於操作端至受控端之資料遺失率。而在本論文中，MPC 之系統模型準確，所以在網路迴授資料遺失時，CDOB 估測之迴授誤差即相對較大，所以此時切換至只使用 MPC 所估測之迴授響應， $k_{cdob} = 0$   $k_{mpc} = 1$ ，由實驗結果圖 5-5 可看出，各參數下之 rise time 並無太大改變、peak value 小幅降低，而 settling time 相對而言降低幅度較大(圖 5-6)，所以由此可知加入此機制後，系統

在維持原本速度之情況下，穩定度可獲得進一步的提升。

表格 5-2 加入資料遺失時之權重切換機制(data dropout rate: 30.9%)

	Rise Time	Peak Value	Settling Time
$k_{cdob} = 0 \ k_{mpc} = 1$	0.67 s	3.00	0.95 s
$k_{cdob} = 0.3 \ k_{mpc} = 0.7$	0.58 s	3.03	0.61 s
$k_{cdob} = 0.5 \ k_{mpc} = 0.5$	0.50 s	3.13	0.94 s
$k_{cdob} = 0.7 \ k_{mpc} = 0.3$	0.53 s	3.09	0.80 s
$k_{cdob} = 1 \ k_{mpc} = 0$	0.46 s	3.21	1.32 s

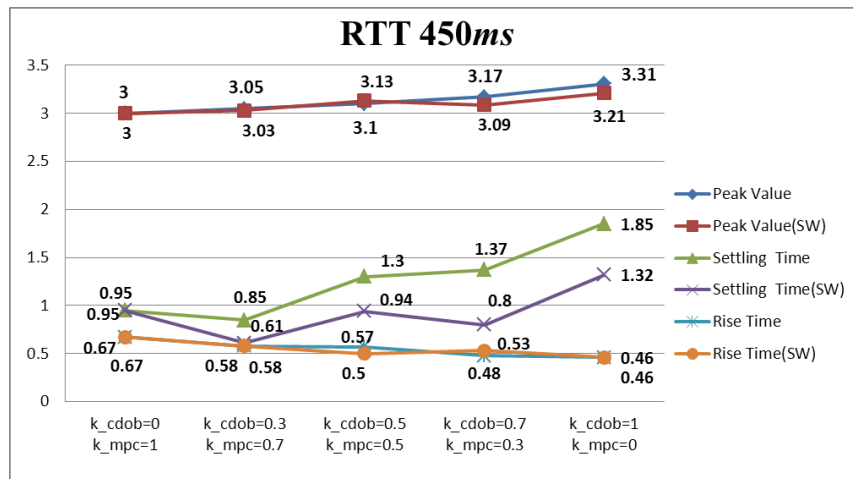


圖 5-5 加入資料遺失時之權重切換機制(data dropout rate: 30.9%)

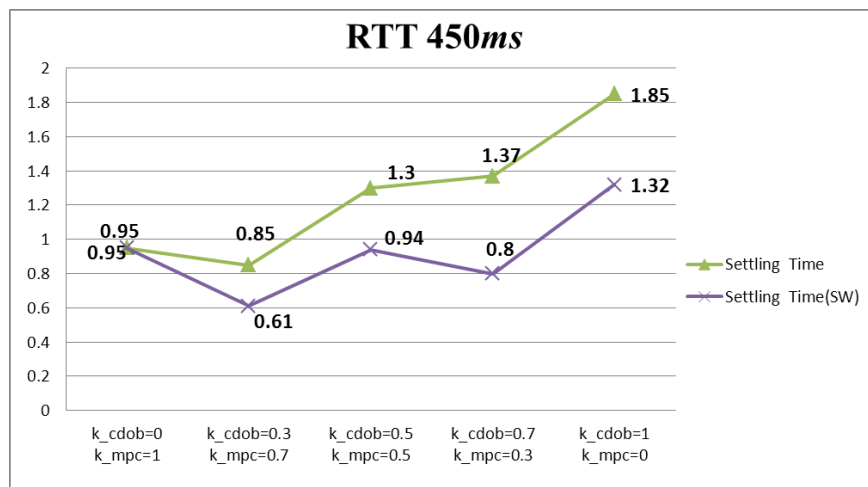


圖 5-6 加入權重切換機制之 settling time 變化

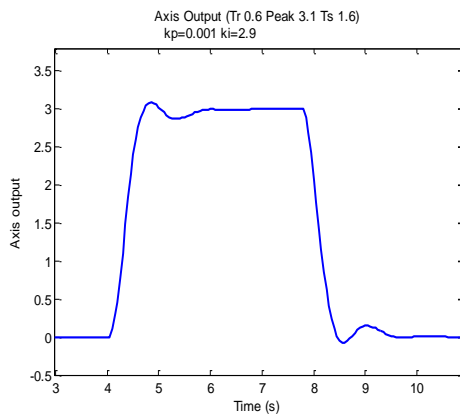
## 5.2 各種控制架構之實驗結果比較

本小節將針對幾個網路控制系統架構進行實驗結果比較：

	控制器	迴授訊號	$t_r$	$t_s$	overshoot
架構一	PI + adaptive Smith predictor	網路直接迴授	0.60 s	1.60 s	3.3%
架構二	PI Controller	CDOB	0.58 s	1.71 s	12.7%
架構三	MPC	CDOB	0.58 s	1.80 s	6.6%
架構四	MPC	MPC+CDOB +weighting	0.58 s	0.61 s	1.0%
架構五	Perfect delay compensation		0.12 s	0.30 s	4.0%

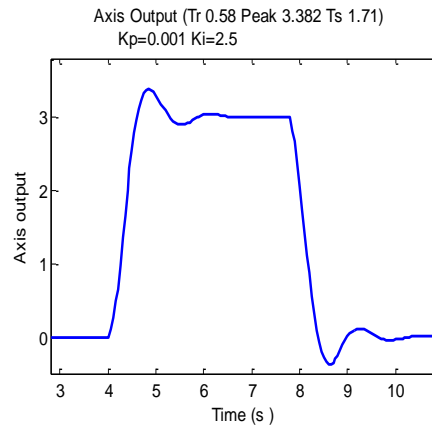
實驗中，將各個控制架構下，在響應之 rise time 大致相同為 0.6 sec 之情況下將其 settling time 與 overshoot 進行性能比較。

由架構一與二之實驗結果比較中，可看到 PI+ adaptive Smith predictor 之響應比 PI+CDOB 為佳，但在 3.6 節中提到，在網路延遲時間變動劇烈的情況下，adaptive Smith predictor 之響應性能會不如 CDOB。



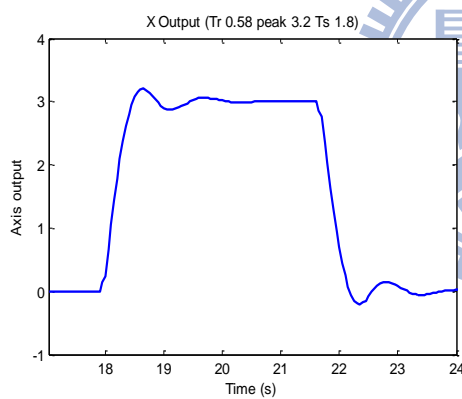
架構一、PI+ adaptive Smith predictor

$t_r$	0.60 sec
$t_s$	1.60 sec
Overshoot	3.3 %



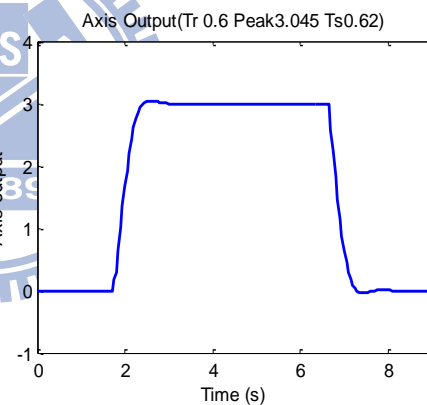
架構二、PI + CDOB

$t_r$	0.58 sec
$t_s$	1.71 sec
Overshoot	12.7 %



架構三、MPC + CDOB

$t_r$	0.58 sec
$t_s$	1.80 sec
Overshoot	6.6 %



架構四、MPC+CDOB

+(Feedback weighting mechanism)

$t_r$	0.58 sec
$t_s$	0.61 sec
Overshoot	1.0%

圖 5-7 各種控制架構之實驗結果比較

而架構二與三之差別為控制器分別為 PI Controller 與 MPC，在 rise time 相同的情

況下，可再次驗證 4.4 節中之結論，MPC 比 PI 控制器在網路控制系統中具有更加性能。從架構三與四中可得知，加入回授訊號權重分配機制後，可大幅縮短 settling time 與 overshoot，而架構四亦比架構一具有更加響應表現。由此四種不同網路控制架構之實驗結果可看出，本論文所提出之控制架構 MPC+CDOB+迴授訊號權重分配機制具有最佳響應表現。

接著，將架構四與 PDC 控制架構進行比較。首先，perfect delay compensation(PDC)控制架構為利用兩個變化型 butterfly 架構之 model-free 控制架構，其架構如圖 5-8 所示：

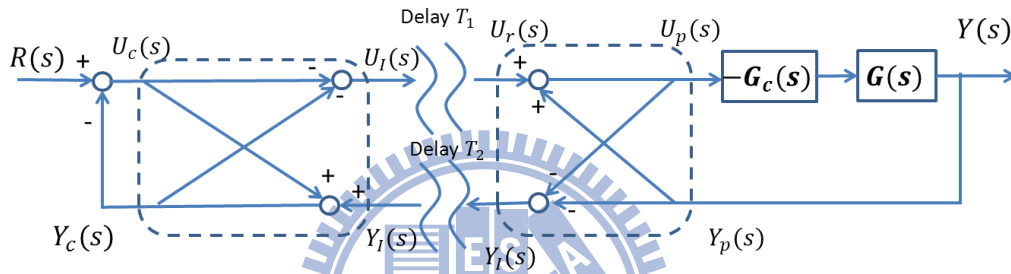


圖 5-8 PDC 控制架構示意圖

由上圖中可得系統訊號關係式：

$$\frac{Y_r(s)}{U_r(s)} = \frac{-U_p(s) - Y_p(s)}{U_p(s) - Y_p(s)} = \frac{G_c(s)G_p(s) - 1}{G_c(s)G_p(s) + 1} \quad (5-1)$$

其中  $\frac{Y_p(s)}{U_p(s)} = -G_c(s)G_p(s)$

在控制架構左邊的轉移函數表示如下：

$$\frac{Y_c(s)}{U_c(s)} = \frac{(1 + G_c(s)G_p(s)) - (G_c(s)G_p(s) - 1)e^{-(t_1+t_2)s}}{(1 + G_c(s)G_p(s)) + (G_c(s)G_p(s) - 1)e^{-(t_1+t_2)s}} \quad (5-2)$$

接著考慮回授訊號 $Y_c(s)$ 與命令 $R(s)$ 之轉移函數：

$$\frac{Y_c(s)}{R(s)} = \frac{(1 + G_c(s)G_p(s)) - (G_c(s)G_p(s) - 1)e^{-(t_1+t_2)s}}{2(1 + G_c(s)G_p(s))} \quad (5-3)$$

由控制架構得知：

$$U_c(s) = R(s) - Y_c(s) \quad (5-4)$$

$$(-U_p(s) - Y_p(s))e^{-t_2s} + U_c(s) = Y_c(s) \quad (5-5)$$

結合式(5-4)、(5-5)，可得到系統轉移函數：

$$G_{NCS}(s) = \frac{Y(s)}{R(s)} = \frac{G_c(s)G_p(s)e^{-t_1s}}{1 + G_c(s)G_p(s)} \quad (5-6)$$

由(5-6)中可以得出，系統轉移函數與未加入網路時之轉移函數只相差了一個 pure forward-delay 項，所以網路延遲的效應不會對系統特性造成影響。

最後將架構四：MPC + CDOB + (Feedback Weighting Mechanism)與 PDC 控制架構之實驗結果進行比較與探討：

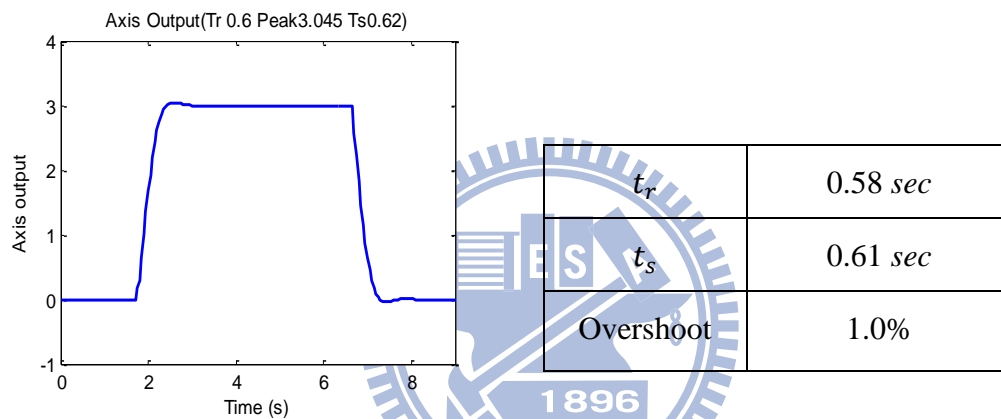


圖 5-9 MPC+CDOB+(Feedback Weighting Mechanism)

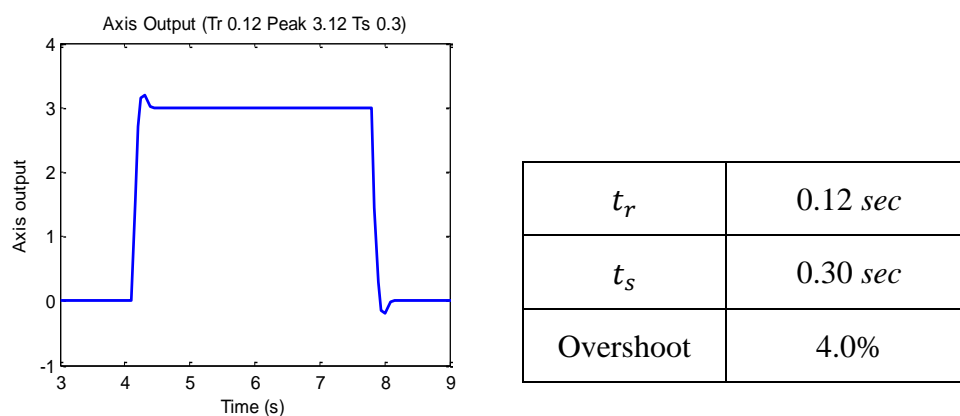


圖 5-10 PDC 控制架構響應



由實驗結果可看到，PDC 控制架構具有更快速之響應，但 overshoot 較大。在真實操作中，對於距離較長之軌跡，人所佔的因素重要性增加，rise time 與 settling time 越相近時，可提供較佳的操作舒適感，且此時操作者所給予的操作命令較大，overshoot 對循軌誤差造成之影響也較大。而對於軌跡較短時，人所佔的因素即相對較小，且操作命令也會較小；操作命令小，底層 overshoot 響應對循軌誤差量之影響也相對較小，所以我們將 PDC 控制架構應用於能夠容忍震盪之短距離操作中。



## 第六章 操作輔助策略

前面幾章提出了解決網路延遲對系統響應影響之方法，由實驗結果驗證，本論文提出之網路控制架構能有效地穩定系統並具有良好的響應結果。但是，網路延遲的效應依然存在，操作命令透過系統執行後的響應結果由網路攝影機擷取畫面後回傳到操作端，前後經過的延遲時間並不會因為控制器或估測器消失，針對此現象，在此提出操作輔助策略，目的於解決網路延遲造成操作困難之問題。

### 6.1 虛擬操作點影像輔助機制

本實驗平台中，受控端之虛擬工作區藉由電腦螢幕顯示，經過網路攝影機擷取影像後回傳至操作端，並顯示於操作端之監控畫面上。受控端之電腦螢幕所處的座標位置為三維座標系統，而操作端之電腦螢幕為二維座標系統，所以在此提出一個輔助機制的概念，即在操作端設計一虛擬操作點，讓操作者藉由此虛擬操作點對著監控螢幕進行操控，接下來的工作都交給系統處理，如果系統能夠準確地將操作端命令轉換至相對的受控端命令，操作者即可不必透過網路攝影機確認操作點移動狀態後進行反應動作，因為經過網路延遲的影像往往增加系統的不透明度(transparency)。本系統將操作端虛擬操作點的命令映射為受控體工作端的命令，架構如圖 6-1 所示，其主要概念為三維與二維之間的座標轉換機制，並且在影像定位機制下可將其簡化為二維與二維的映射機制。

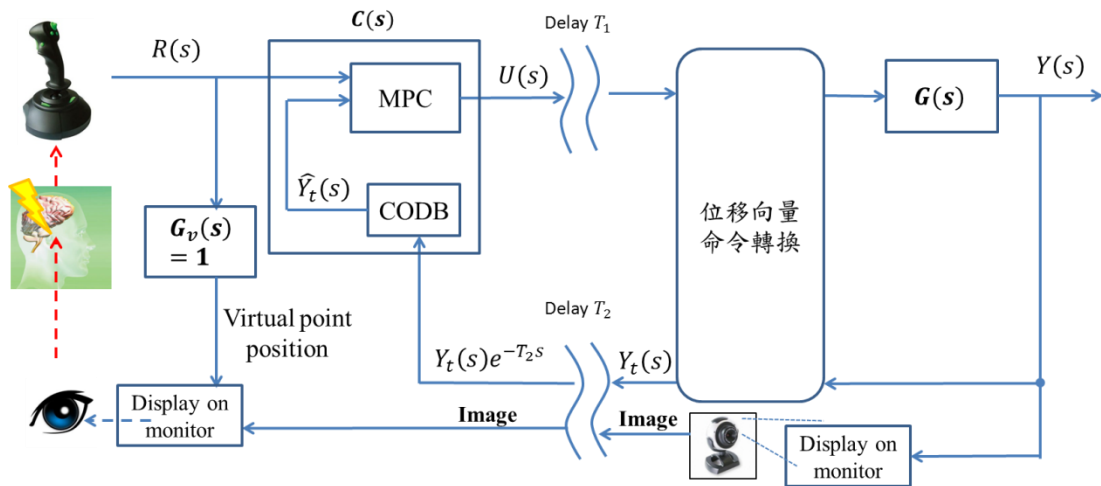


圖 6-1 虛擬操作與影響輔助機制架構圖

首先我們先考慮圖 6-2 中兩個工作空間，視覺空間(visual Space)為網路攝影機擷取畫面後於焦距  $f$  成像的空間，而受控端存在於工作空間(work space)中， $C$  為攝影機之鏡頭位置，亦是投影的中心點，圖中兩點  $P_I$  與  $P_S$  分別存在於視覺空間與工作空間中，其座標分別為  $(x_i, y_i, f)$ 、 $(x_s, y_s, z_s)$ ，皆用網路攝影機所處的座標軸  $(x, y, z)$  表示之。特別注意的是工作空間為三維空間，我們想要得到的是工作空間與成像平面間的座標轉換矩陣。

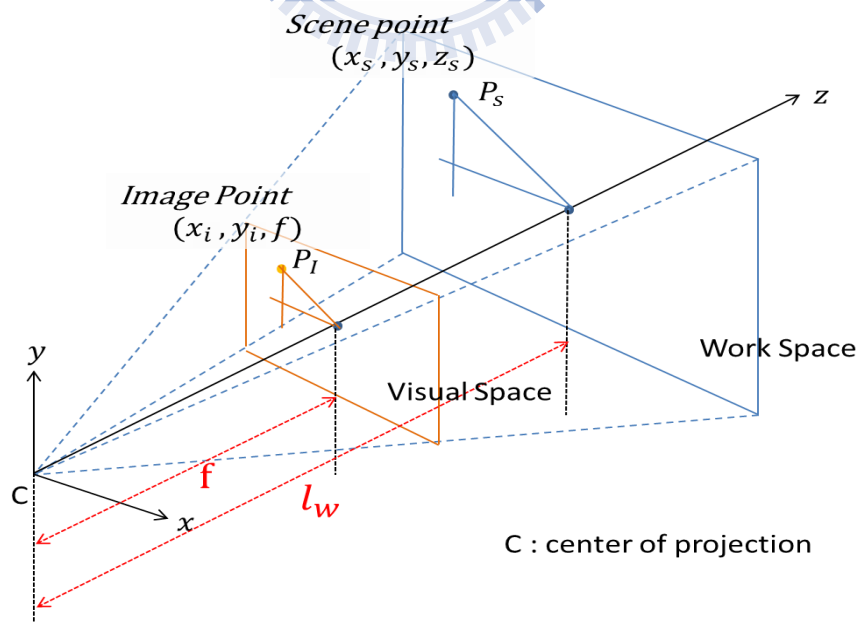


圖 6-2 工作空間於視覺空間點投影示意圖

在此我們的第一個目標即為找出將 $P_s$ 投影至 $P_l$ 之映射矩陣。由三角形的相似形理論可得：

$$x_i = f \frac{x_s}{z_s}$$

$$y_i = f \frac{y_s}{z_s}$$

令 $x_i = \frac{u}{w}$ 、 $y_i = \frac{v}{w}$ ，可得轉移矩陣：

$$\begin{bmatrix} u \\ v \\ w \end{bmatrix} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_s \\ y_s \\ z_s \\ 1 \end{bmatrix}$$

$$= [K | 0_3] \begin{bmatrix} x_s \\ y_s \\ z_s \\ 1 \end{bmatrix} \quad (6-1)$$

其中 $K = \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix}$ ； $0_3 = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$

接著考慮工作空間與網路攝影機所處座標之間的轉換關係。首先考慮圖 6-3 中的點 M，此點用工作空間座標 $(I, J, K)$ 表示為 $(X_s, Y_s, Z_s)$ ，同樣的點 M 用視覺空間座標 $(i, j, k)$ 則表示為 $(x_s, y_s, z_s)$ ，接下來考慮圖 6-3 中三項量之間的關係：

$$\overrightarrow{CM} = \overrightarrow{OM} - \overrightarrow{OC}$$

由此關係可得：

$$\begin{bmatrix} x_s & y_s & z_s \end{bmatrix} \begin{bmatrix} i \\ j \\ k \end{bmatrix} = \begin{bmatrix} X_s & Y_s & Z_s \end{bmatrix} \begin{bmatrix} I \\ J \\ K \end{bmatrix} - \tilde{C} \begin{bmatrix} I \\ J \\ K \end{bmatrix} \quad (6-2)$$

其中 $\tilde{C} = \begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix}$  (工作空間座標表示之 $\overrightarrow{OC}$ )

展開(6-2)式可得：

$$x_s \mathbf{i} + y_s \mathbf{j} + z_s \mathbf{k} = (X_s - X_c) \mathbf{i} + (Y_s - Y_c) \mathbf{j} + (Z_s - Z_c) \mathbf{k} \quad (6-2)$$

將上式左右同時對向量 $\mathbf{i}$ 內積可得：

$$x_s = (X_s - X_c)I.i + (Y_s - Y_c)J.i + (Z_s - Z_c)K.i$$

利用相同操作，將(6-2)式分別對 j, k 內積，即可得映射矩陣：

$$\begin{bmatrix} x_s \\ y_s \\ z_s \\ 1 \end{bmatrix} = \begin{bmatrix} R & -R\tilde{C} \\ 0_3^T & 1 \end{bmatrix} \begin{bmatrix} X_s \\ Y_s \\ Z_s \\ 1 \end{bmatrix} \quad (6-3)$$

$$\text{其中 } R = \begin{bmatrix} I.i & J.i & K.i \\ I.j & J.j & K.j \\ I.k & J.k & K.k \end{bmatrix}$$

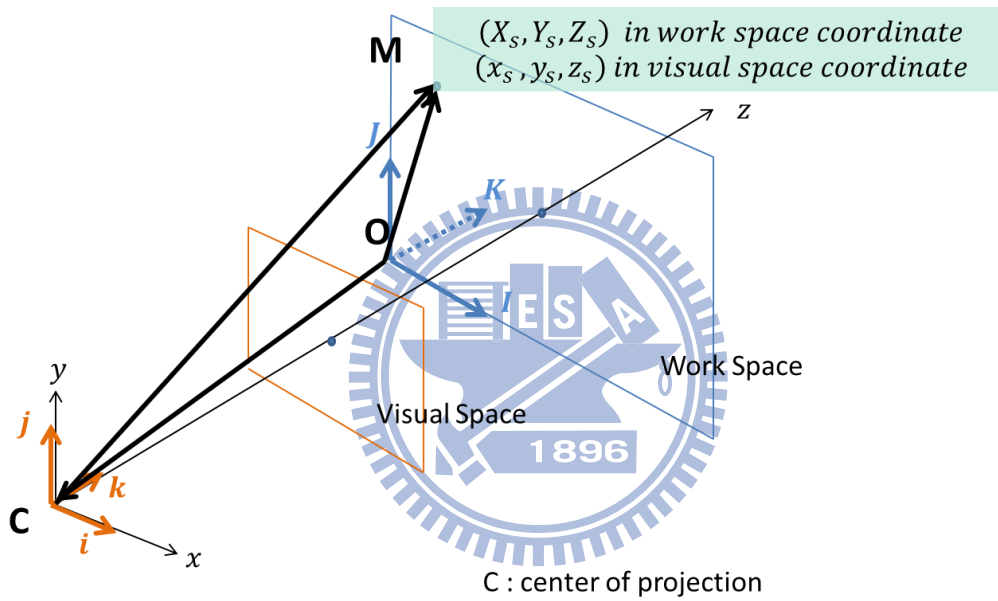


圖 6-3 工作空間與視覺空間座標系統轉換示意圖

有了工作空間對視覺空間之點投影映射矩陣(6-1)與工作空間與視覺空間中之座標映射矩陣(6-3)，可整理出工作空間任一點對攝影機成像平面投影的座標轉換矩陣：

$$\begin{bmatrix} u \\ v \\ w \end{bmatrix} = K[I_3 | 0_3] \begin{bmatrix} R & -R\tilde{C} \\ 0_3^T & 1 \end{bmatrix} \begin{bmatrix} X_s \\ Y_s \\ Z_s \\ 1 \end{bmatrix}$$

$$= KR \begin{bmatrix} I_3 & | & -\tilde{C} \end{bmatrix} \begin{bmatrix} X_s \\ Y_s \\ Z_s \\ 1 \end{bmatrix} \quad (6-4)$$

接下來我們要做的是找出向量  $\tilde{C}$  與攝影機焦距  $f$ ，始可獲得(6-4)所需要的參數。

我們可以從圖 6-4 中看到網路攝影機與虛擬工作影像相對位置，雖然電腦螢幕是在三維空間中，但是其顯示螢幕本身其實是二維平面，所以透過移動網路攝影機，可使攝影機成像平面與照射到的平面呈現平行狀態，並且兩平面的中心點與網路攝影機之攝像頭呈現一直線，從網路攝影機看過去之透視圖，如圖 6-5 所示。

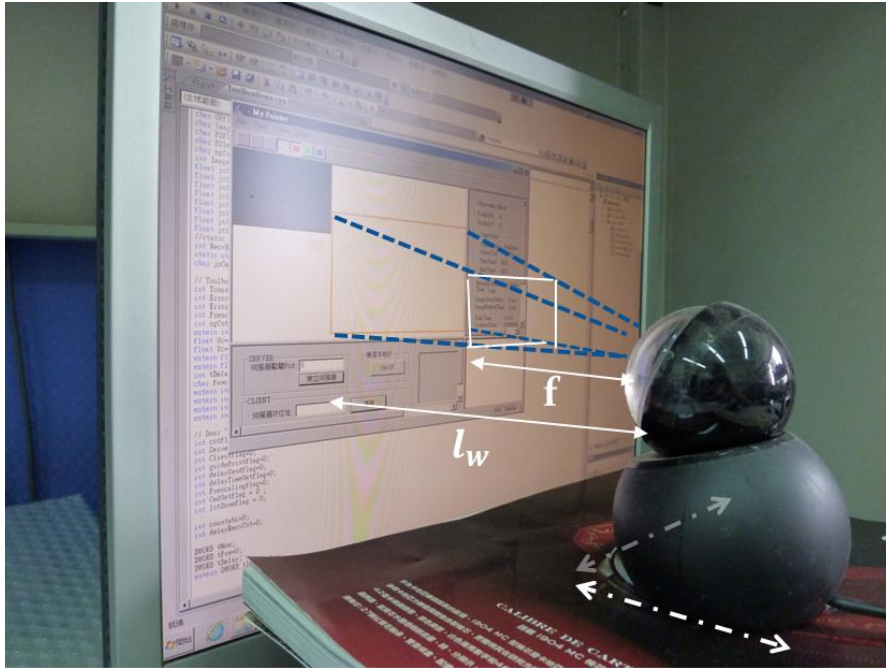


圖 6-4 Webcam 與虛擬工作影像相對位置圖

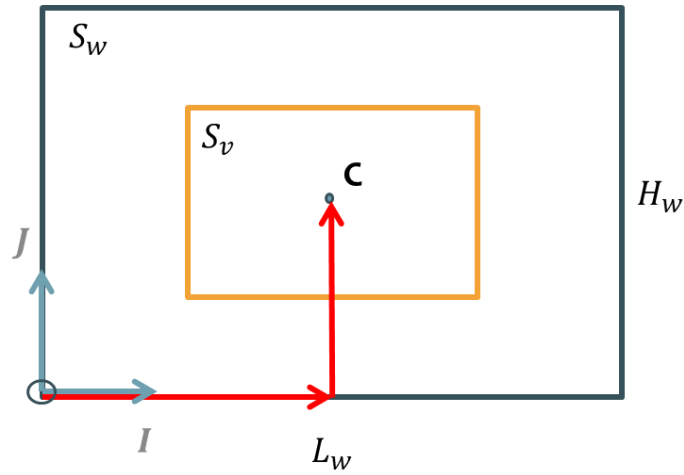


圖 6-5 由 Webcam 鏡頭朝工作區看之透視示意圖

如此一來即可找到向量  $\tilde{C}$ ：

$$\tilde{C} = \begin{bmatrix} \frac{L_w}{2} \\ \frac{H_w}{2} \\ -l_w \end{bmatrix} \quad (6-5)$$

接著由圖 6-6 所示，同樣的利用相似形原理，我們可以得到網路攝影機成像焦距  $f$ ：

$$f = \frac{a}{A} l_w \quad (6-6)$$

其中  $A$  為操作點於工作空間與網路攝影機照射畫面平面中心點之距離， $a$  為投影在成像平面之操作點與成像平面中心點距離。

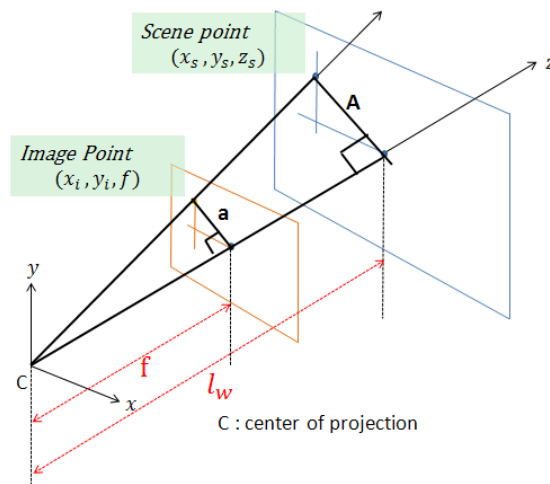


圖 6-6 Webcam 與工作區距離與成像焦距關係示意圖

最後，將(6-5)與(6-6)得到之條件帶回(6-4)之映射矩陣中：

$$\begin{bmatrix} u \\ v \\ w \end{bmatrix} = KR[I_3 \mid -\tilde{C}] \begin{bmatrix} X_s \\ Y_s \\ Z_s \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{a}{A} l_w & 0 & 0 \\ 0 & \frac{a}{A} l_w & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} I.i & J.i & K.i \\ I.j & J.j & K.j \\ I.k & J.k & K.k \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & \frac{L_w}{2} \\ 0 & 1 & 0 & \frac{H_w}{2} \\ 0 & 0 & 1 & -l_w \end{bmatrix} \quad (6-7)$$

展開後可得：

$$u = f(X_s - X_c) = \frac{x_i}{w}$$

$$v = f(Y_s - X_c) = \frac{y_i}{w}$$

$$w = Z_s - X_c = -X_c = l_w$$

經過整理後最後可得到於視覺空間之成像平面上之期望的點座標：

$$\begin{aligned} x_i &= \frac{l_w \frac{a}{A} (X_s - X_c)}{l_w} = \frac{a}{A} (X_s - X_c) \\ y_i &= \frac{l_w \frac{a}{A} (Y_s - X_c)}{l_w} = \frac{a}{A} (Y_s - Y_c) \end{aligned} \quad (6-8)$$

由於網路攝影機成像的畫面即為操作端看到的畫面，圖 6-7 中，網路攝影機前的小框即為其成像平面，我們可以想像成，將一虛擬操作點至於此平面上，將欲移動距離透過等比例放大後，投射在虛擬工作區的真實操作點中，如圖 6-7 所示。

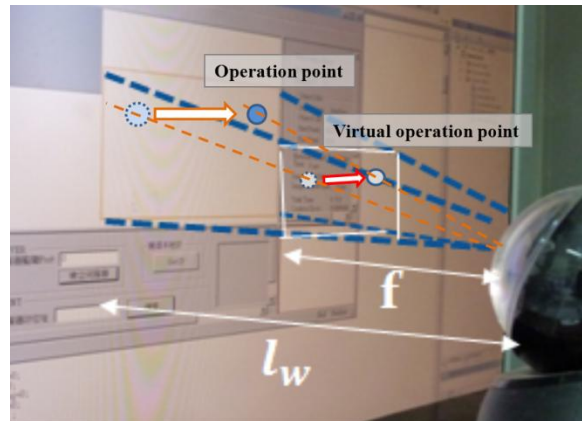


圖 6-7 虛擬操作點與實際操作點映射示意圖



## 6.2 操作點預測輔助機制

雖然虛擬操作點影像輔助機制能夠有效地降低循軌誤差，但是必須在影像定位正確之前提下才能有預期的效果，如果在影像定位產生誤差的情況下，就會造成嚴重誤差，所以在此提出另一較適用於一般環境中的操作點預測輔助策略機制，以克服網路影像定位對輔助機制之限制，架構如圖 6-8 所示。

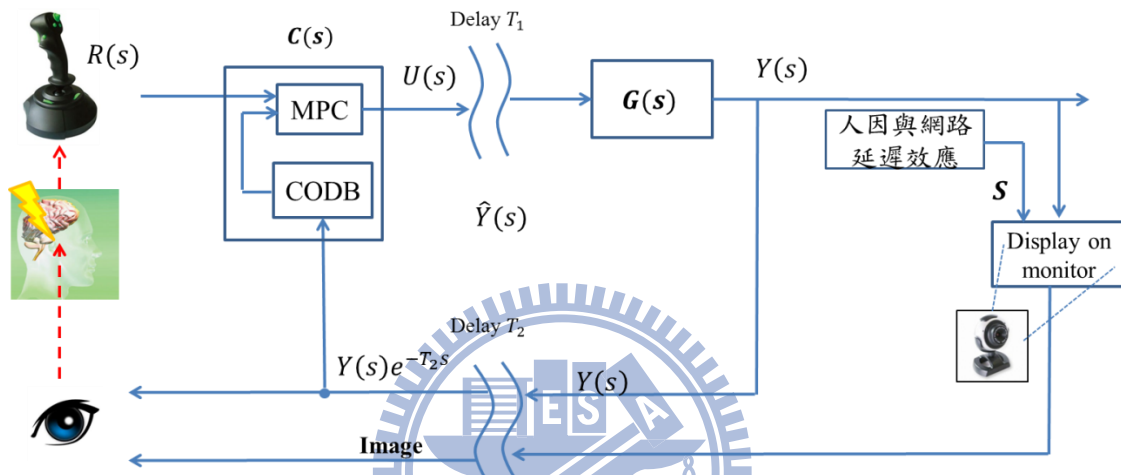


圖 6-8 操作點預測輔助機制架構圖

首先系統會根據操作者對第一個物件循軌的結果進行記錄，由於一般人在操作時如果預期會有網路延遲的影響，往往會有先給予些微命令，然後等待其結果再判斷接下來動作的情況產生，不過由於系統需要紀錄操作者的反應時間與網路對操作者造成的影響，所以此時操作者在進行第一個物件循軌時，必須盡量給予固定的位置命令，當操作者看到操作點正好到物件的終點時，將搖桿放開，如此就能提供系統需要的必要資訊即圖 6-8 中之 human factors information 部分，操作情形如圖 6-9 所示。

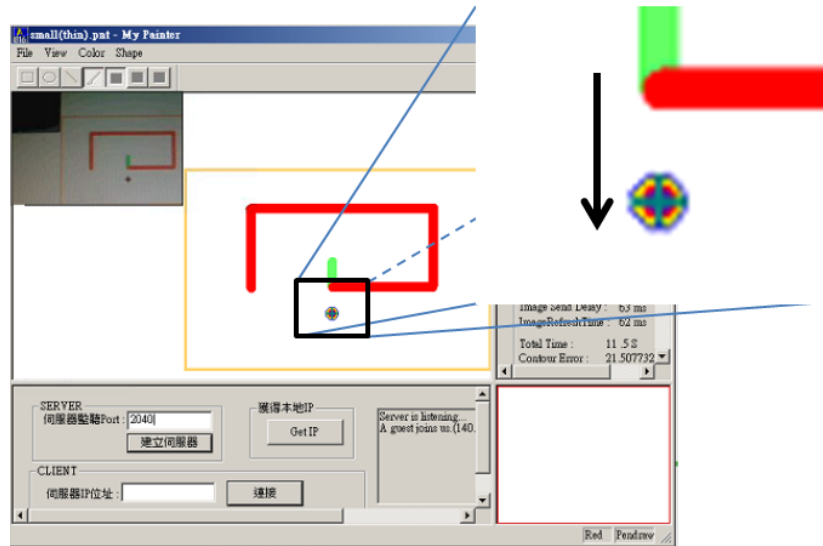


圖 6-9 操作點預測之學習機制

如此動作的意義在於，當操作者看到操作點到達目標時，將搖桿放開或往反方向下位置命令，可以預期的是，操作點會因為網路延遲對操作者造成的影響，超過預期的位置，然而在這樣前提下的循軌誤差，其實就代表著：如果操作者某個時刻希望操作點停止，此段距離即代表操作者當初要在多遠之前將搖桿放開。

因此，在此操作規則下，第一段循軌誤差包含了操作者的反應時間與網路延遲影響的資訊，另外要納入考量的就是操作點目前的移動速度，由於操作點的移動速度並不會永遠與第一次循軌時之速度相等，所以在預測操作點時，我們必須預測未來操作點之平均速度，以估測出操作前置量。預測操作點之概念如圖 6-10 所示：

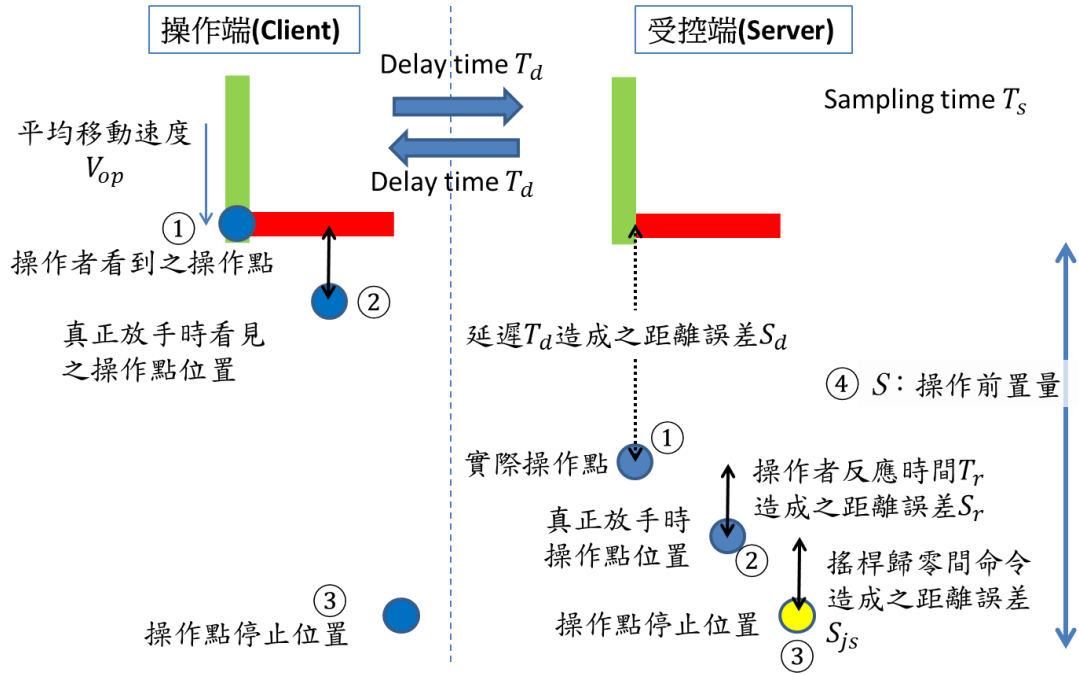


圖 6-10 預測操作點之概念示意圖

- ①：當操作者看到操作點以平均速度  $V_{op}$  移動至第一個線段之終點時，停止操作命令。可以預期的是，此時在受控端之操作點實際位置會因為網路延遲的關係距離第一個線段之終點  $S_d = V_{op}[T_d/T_s]$ 。 $T_s$  為系統中斷時間。
- ②：由於從實際看見到真正停止操作需要經過一段視覺反應時間  $T_r$ ，在這段時間內所行進之距離為操作者反應時間造成之距離誤差  $S_r = V_{op}[T_r/T_s]$ 。
- ③：當操作者真正放開搖桿時，搖桿之操作命令會以漸進的方式歸零，這段時間內造成之距離誤差為  $S_{js}$ 。
- ④：最後，可得到在等速循軌下，由網路延遲、操作者反應時間與搖桿歸零反應時間內造成的循軌誤差  $S = S_d + S_r + S_{js}$ 。此前置量之意義即代表，未來在操作時需要多少距離之前“提早”停止操作。

接著，對操作前置量  $S$  進行分析。由上述可得知：

$$S = V_{op} \left[ \frac{T_d + T_r}{T_s} \right] + S_{js} \quad (6-9)$$

而本實驗中搖桿操作命令範圍為 0~4 pixel，第一次操作時操作者會將操作速度維

持在 $V_{op} = 3$  或  $4$  之間；往返網路延遲 RTT 為  $450\text{ ms}$ ，所以單程延遲時間 $T_d$ 為  $225\text{ ms}$ ，而人類視覺反應時間 $T_r$ 約為  $150\text{ ms}$  [10]，系統之中斷時間 $T_s = 50\text{ ms}$ 。

所以由上述條件代回(6-9)式中可得：

$$S = \left[ (3 \text{ or } 4) \left\lceil \frac{225 + 150}{50} \right\rceil + S_{js} \right] = [(24 \sim 32 \text{ pixel}) + S_{js}] \quad (6-10)$$

由實驗結果(6.4.2 節)可得知，第一次操作時之循軌誤差一般亦約在  $24 \sim 32\text{ pixel}$  之間，所以可發現 $S_{js}$ 對 $S$ 之影響可忽略，最後可將操作前置量  $S$  表示如下：

$$S = V_{op} \left\lceil \frac{T_d + T_r}{T_s} \right\rceil = V_{op} N \quad (6-11)$$

$N$ ：操作點超過第一個線段終點後，在停止操作前操作點位移之次數

然而操作者之操作命令會隨時間改變，所以必須根據不同之操作點移動速度 $V_{op}$ 決定其前置量  $S$ 。由上述之條件可得知，一般  $N$  值為  $\left\lceil \frac{225+150}{50} \right\rceil = 8$ ，所以在操作點維持等速之情況下，取前八個操作點位移量之平均值，做為未來八個中斷時間內之平均速度，並以此估測速度 $V'_{op}$ 決定此時之操作前置量 $S = V'_{op} N$ 。

### 6.3 微調切換機制

以本實驗而言，在每段直線之間之轉折處，往往是造成循軌誤差的關鍵，如圖 6-11 所示之四個轉折點。

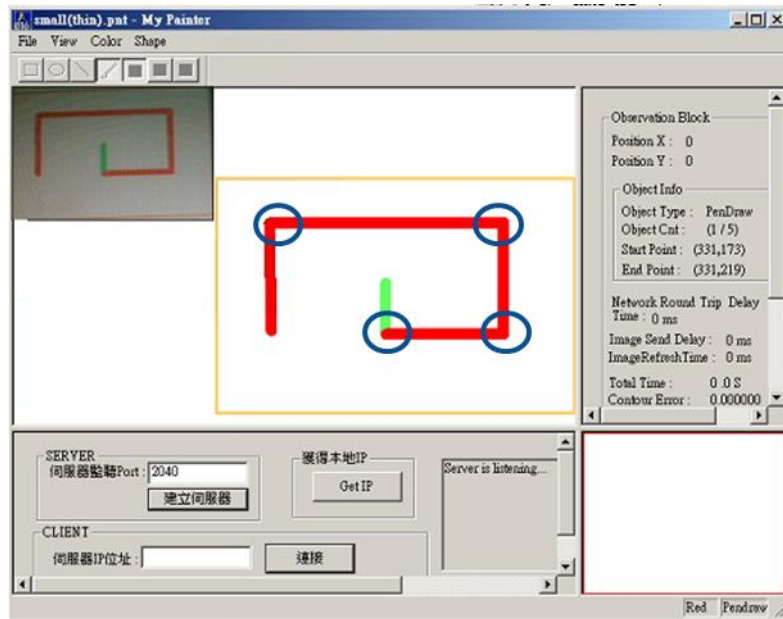


圖 6-11 循軌誤差發生處

操作點輔助策略可以讓操作者將操作點確實的停在預期位置附近，但往往不是百分之一百準確，所以此時必須進行微調動作。然而 rise time 越長時，操作者容易誤認為網路延遲效應而持續給予操作命令，進而造成循軌誤差，所以在進行微調時 rise time 越小越佳。

PDC 控制架構具有低 rise time、低 settling time 之良好響應之優點，雖然 overshoot 較大，但在短距離之操作中由於操作命令小，所以造成之誤差則相對於長距離較小。

因此，在轉折點必須進行微調的情況發生時，操作者可切換至 PDC 控制架構，利用其快速的響應對操作點進行微調。此切換機制架構如圖 6-12 所示：

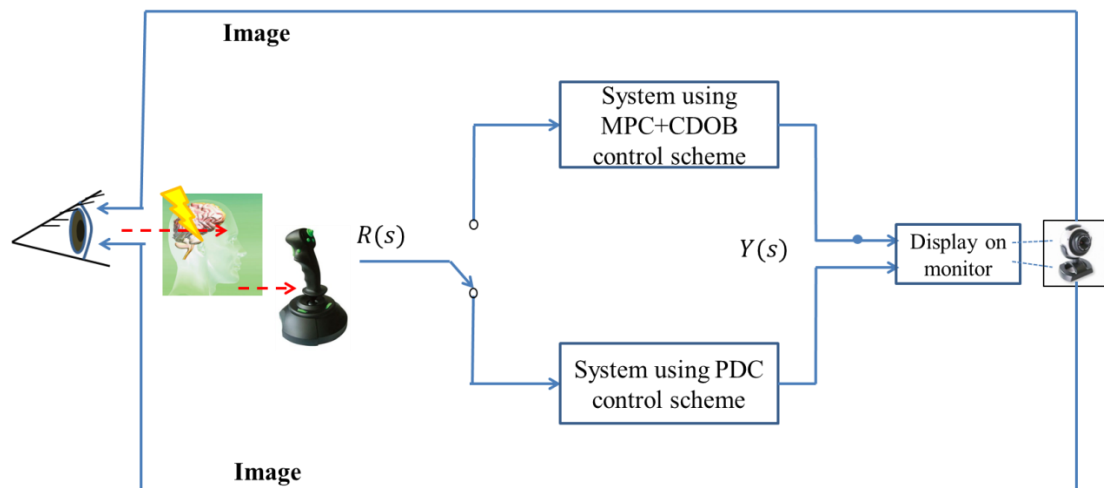
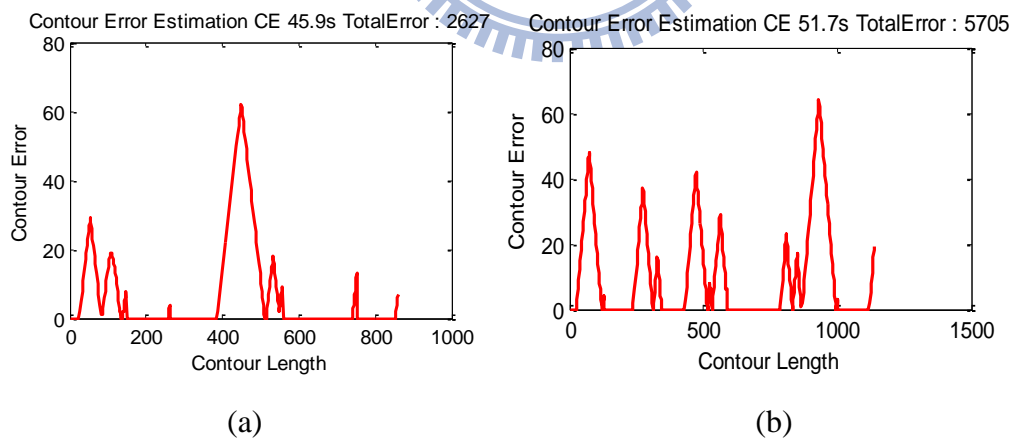


圖 6-12 微調切換機制架構圖

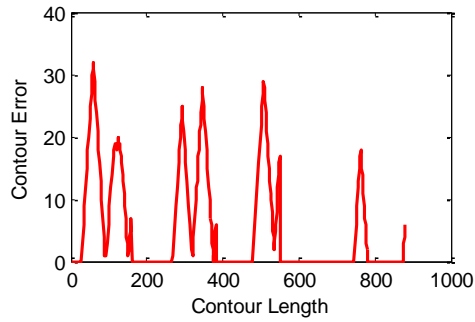
## 6.4 實驗結果

本實驗邀請五位受測者進行操作，每位受測者操作兩次，最後將實驗結果中循軌誤差最高與最低的兩組數據捨去，期望留下較客觀的實驗平均數據。首先考慮未加入任何輔助機制以當作實驗對照組：

### 6.4.1 方形直線循軌實驗

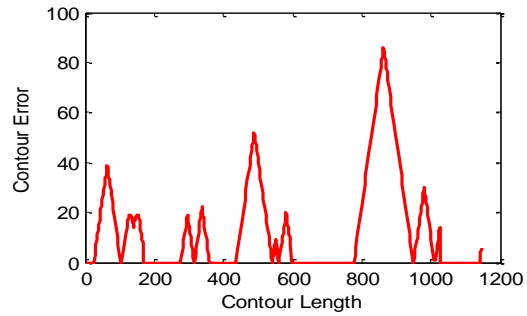


Contour Error Estimation CE 42.3s TotalError : 2639



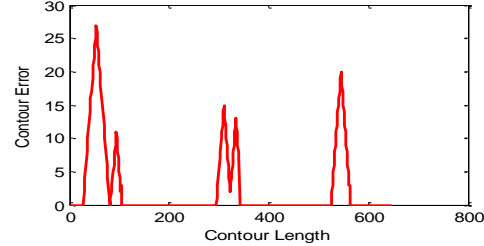
(c)

Contour Error Estimation CE 46.7s TotalError : 6618



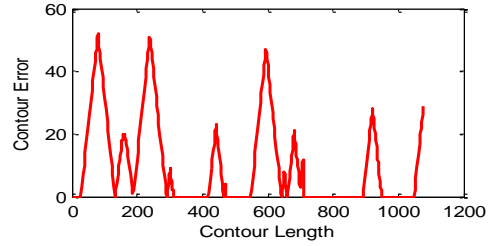
(d)

Contour Error Estimation CE 56.9s TotalError : 1356



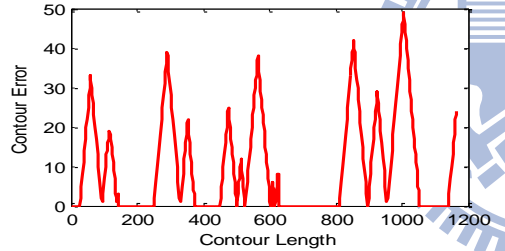
(e)

Contour Error Estimation CE 41.2s TotalError : 4948



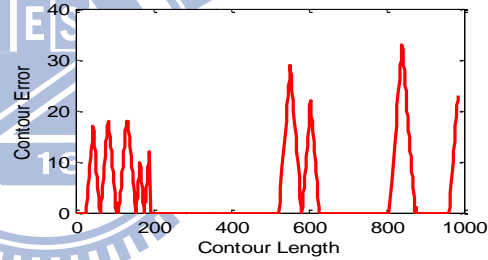
(f)

Contour Error Estimation CE 42.4s TotalError : 5450



(g)

Contour Error Estimation CE 74s TotalError : 2887



(h)

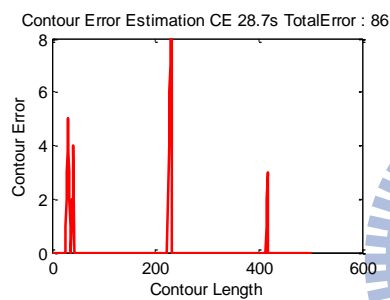
圖 6-13 未加入操作輔助機制之直線平均循軌誤差

表格 6-1 未加入操作輔助機制直線平均循軌誤差數據

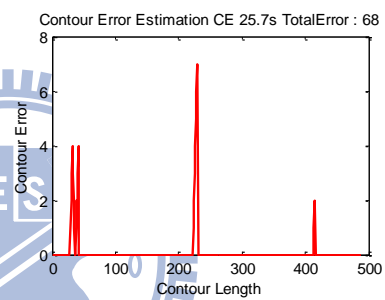
	Complete Time	Contour Error
Subject 1 exp1	45.9 sec	2627
Subject 1 exp2	51.7 sec	5705
Subject 2 exp1	42.3 sec	2639
Subject 2 exp2	46.7 sec	6618
Subject 3 exp1	56.9 sec	1356

Subject 3 exp2	41.2 <i>sec</i>	4948
Subject 4 exp1	42.4 <i>sec</i>	5450
<del>Subject 4 exp2(excluded)</del>	<del>23.3 <i>sec</i></del>	<del>549</del>
<del>Subject 5 exp1(excluded)</del>	<del>42.7 <i>sec</i></del>	<del>6634</del>
Subject 5 exp2	74 <i>sec</i>	2887
Average	50.1 <i>sec</i>	4028.7

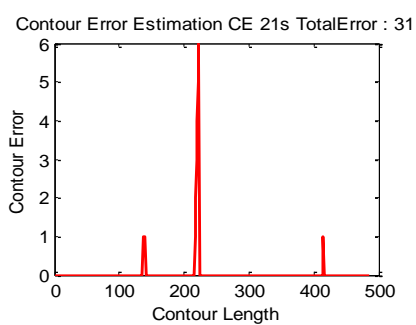
#### 6.4.1.1 虛擬操作點影像輔助機制



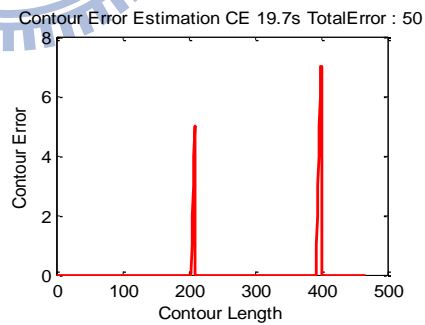
(a)



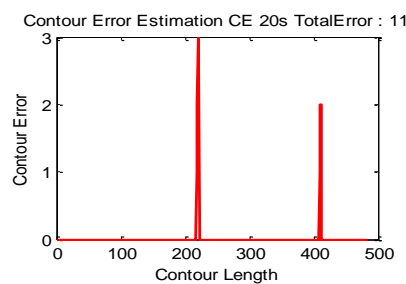
(b)



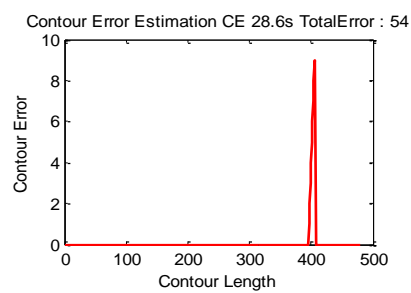
(c)



(d)

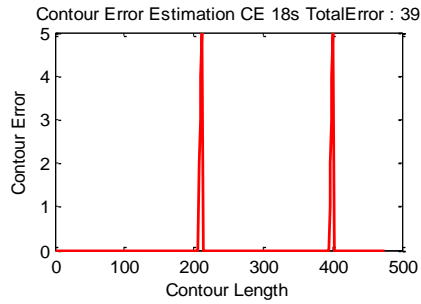


(e)

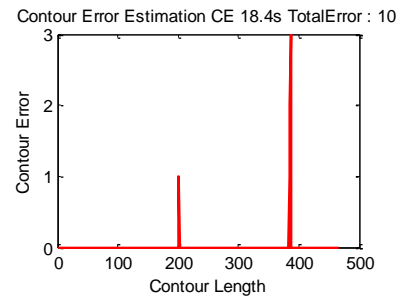


(f)





(g)



(h)

圖 6-14 加入操作點影像輔助機制之直線平均循軌誤差

表格 6-2 加入操作點影像輔助直線平均循軌誤差數據

	Complete Time	Contour Error
Subject 1 exp1	28.7 sec	86
Subject 1 exp2	25.7 sec	68
Subject 2 exp1	21 sec	31
Subject 2 exp2	19.7 sec	50
Subject 3 exp1(excluded)	21.3 sec	89
Subject 3 exp2	20 sec	11
Subject 4 exp1(excluded)	23.6 sec	7
Subject 4 exp2	28.6 sec	54
Subject 5 exp1	18.4 sec	10
Subject 5 exp2	18 sec	39
Average	22.5 sec	43.6

由實驗結果可證實，虛擬操作點輔助機制可大幅改善循軌誤差高達 98.9%，並且縮短 55% 完成時間。然而此輔助機制需要一定的條件限制，例如我們必須得知在工作區影像擷取範圍，所以一旦影像範圍定位產生誤差時，如圖 6-15 與圖 6-16 所示，此輔助機制便會失效，循軌結果甚至更加不理想，由圖 6-17 之實驗結果可以獲得驗證。

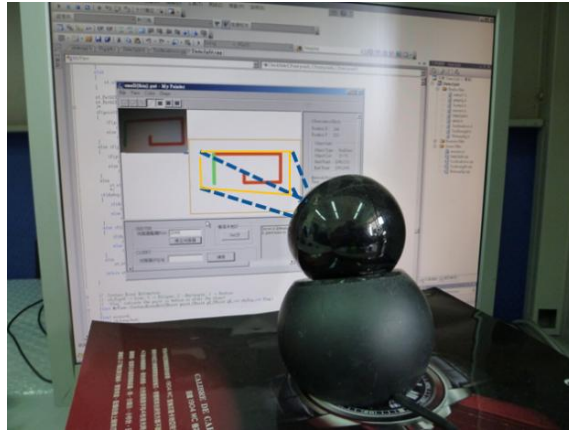


圖 6-15 影像擷取範圍確認誤差示意圖

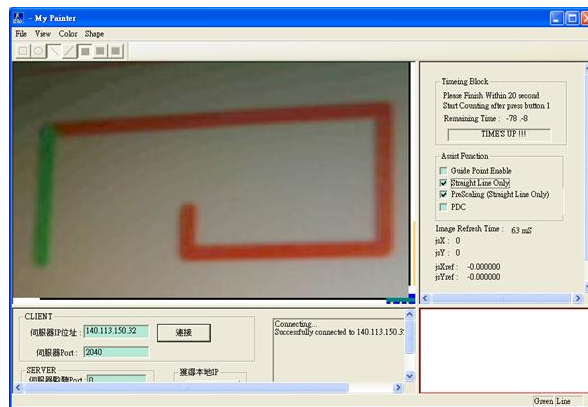


圖 6-16 影像定位誤差之操作端示意圖

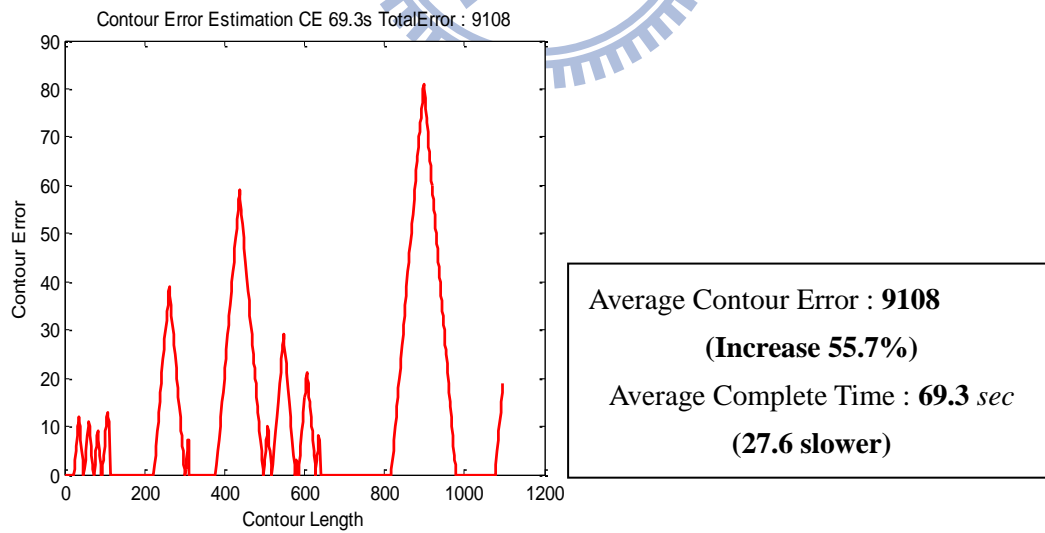
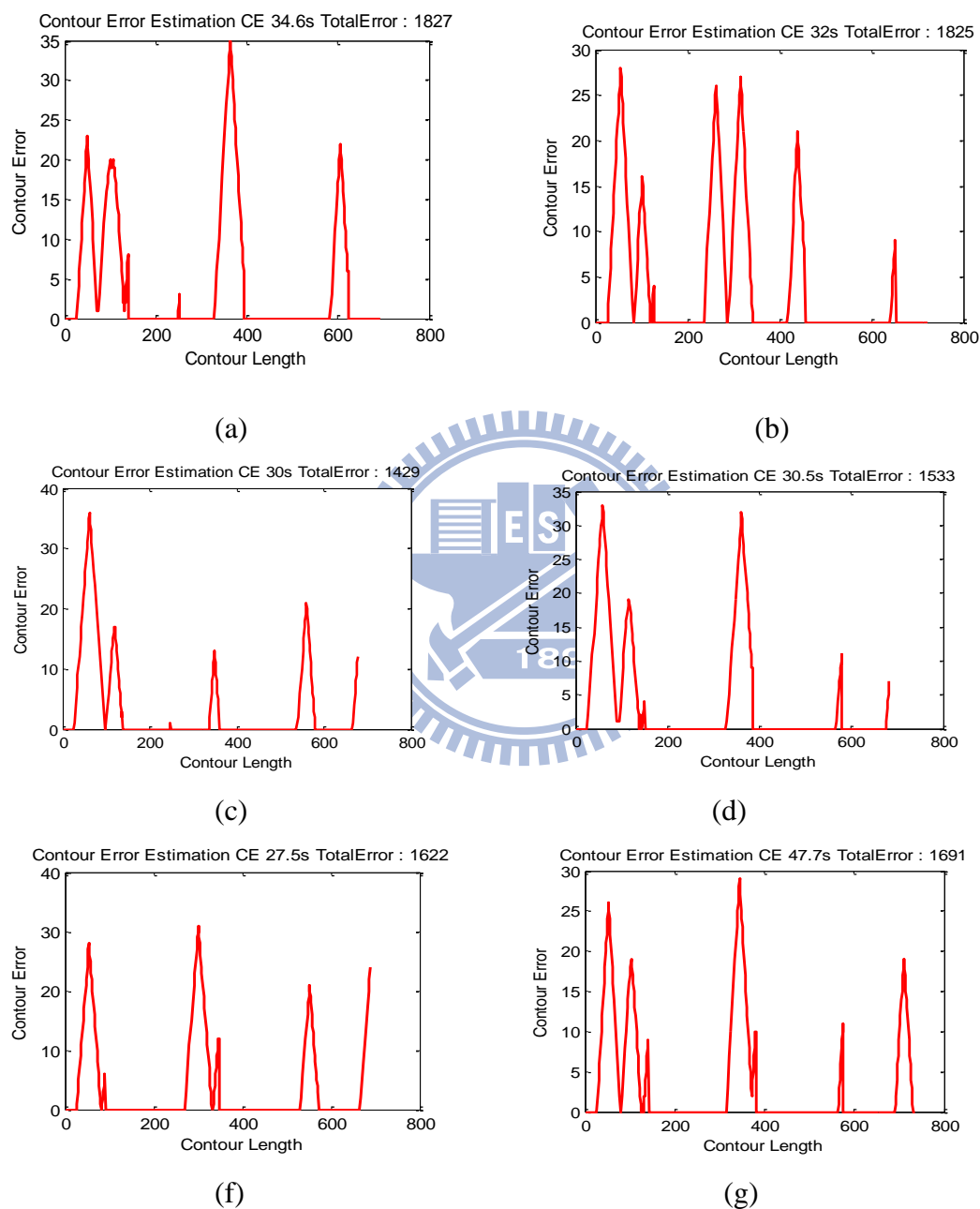
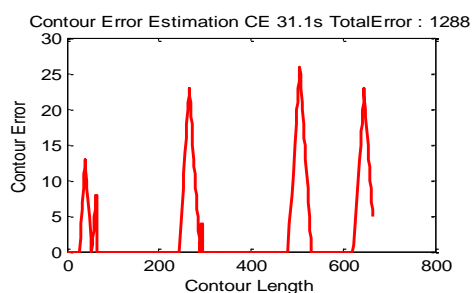


圖 6-17 影像定位誤差時之直線平均循軌誤差

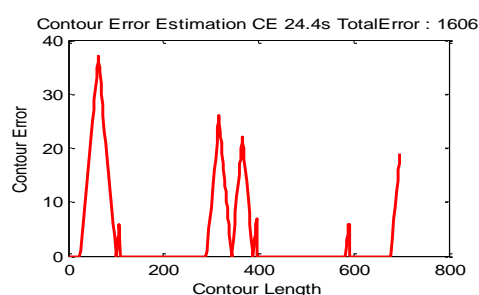
### 6.4.1.2 操作點預測輔助機制

有鑒於此，本論文設計了適用於一般情況且克服影像定位誤差對輔助機制造成不良影響之輔助策略－操作點預測輔助機制，實驗結果如下：





(h)



(i)

圖 6-18 加入操作點預測輔助機制之直線平均循軌誤差

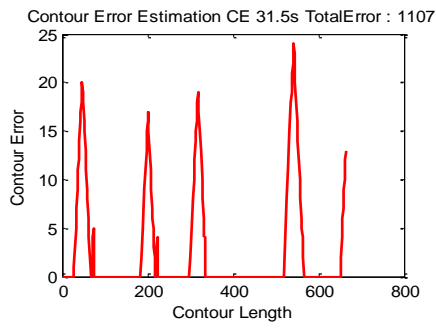
表格 6-3 加入操作點預測機制直線平均循軌誤差數據

	Complete Time	Contour Error
Subject 1 exp1	34.6 sec	1827
Subject 1 exp2	32 sec	1825
Subject 2 exp1	30 sec	1429
Subject 2 exp2	30.5 sec	1533
Subject 3 exp1	27.5 sec	1622
Subject 3 exp2(excluded)	28.5 sec	1192
Subject 4 exp1(excluded)	34.3 sec	1843
Subject 4 exp2	47.7 sec	1691
Subject 5 exp1	31.1 sec	1288
Subject 5 exp2	24.4 sec	1606
Average	32.2 sec	1602.6

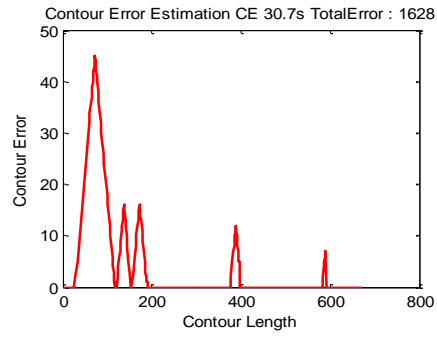
實驗結果顯示，輔助點預測輔助機制依然能有效地降低循軌誤差達 60%，並且提高完成時間 35.7%。

### 6.4.1.3 微調切換機制

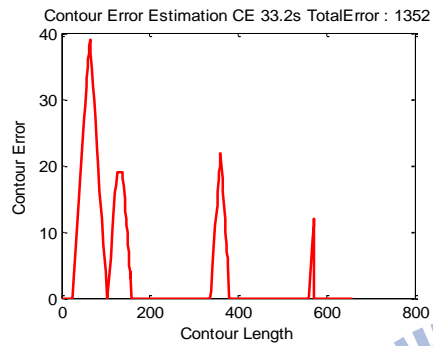
有鑑於在轉折點間微調動作造成循軌誤差，加入微調切換機制：



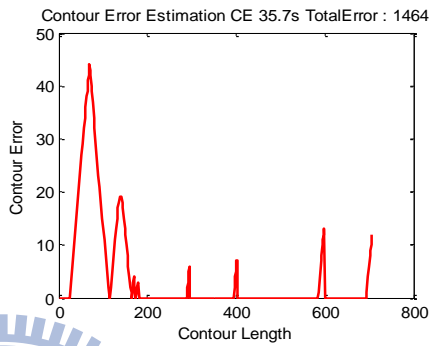
(a)



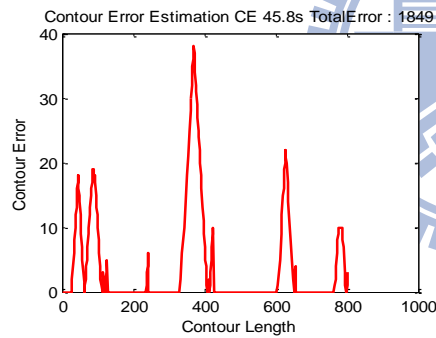
(b)



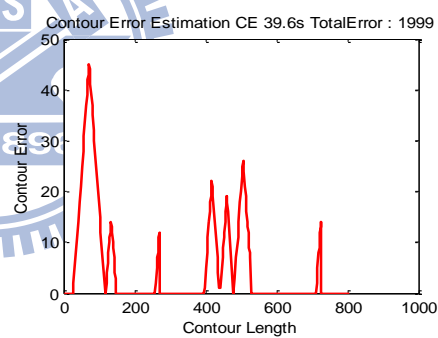
(c)



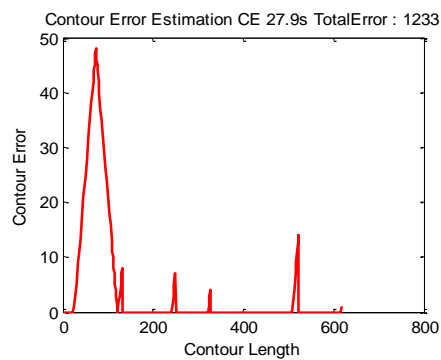
(d)



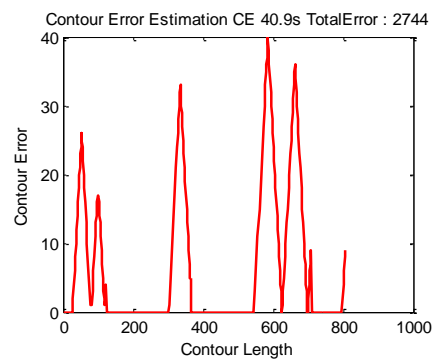
(e)



(f)



(g)



(h)

圖 6-19 加入微調切換機制之直線平均循軌誤差

表格 6-4 加入微調切換機制直線平均循軌誤差數據

	Complete Time	Contour Error
Subject 1 exp1	31.5 sec	1107
<del>Subject 1 exp2(excluded)</del>	<del>38.4 sec</del>	668
Subject 2 exp1	33.2 sec	1352
Subject 2 exp2	35.7 sec	1464
Subject 3 exp1	45.8 sec	1849
Subject 3 exp2	39.6 sec	1999
Subject 4 exp1	27.9 sec	1233
Subject 4 exp2	34.6 sec	737
Subject 5 exp1	30.7 sec	1628
<del>Subject 5 exp2(excluded)</del>	<del>40.9 sec</del>	2744
Average	34.8 sec	1421.1

實驗結果顯示，加入微調切換機制後，可將循軌誤差再降低 11.3%，但是完成時間增加了大約兩秒，原因在於微調切換機制需要操作者手動操作，基於受測者對於切換機制並未能馬上熟練，所以在此即顯示在平均完成時間內。為了凸顯微調切換機制之效能，在此設計了另一個實驗，針對循軌長度較短且轉折點多的情況下進行實驗，如圖 6-20 所示，在實驗過程中，加入微調切換機制時，僅在第一個轉折點進行機制切換，自此後便持續使用 PDC 控制架構，如此可將操作者對於操作切換之熟練度問題排除：

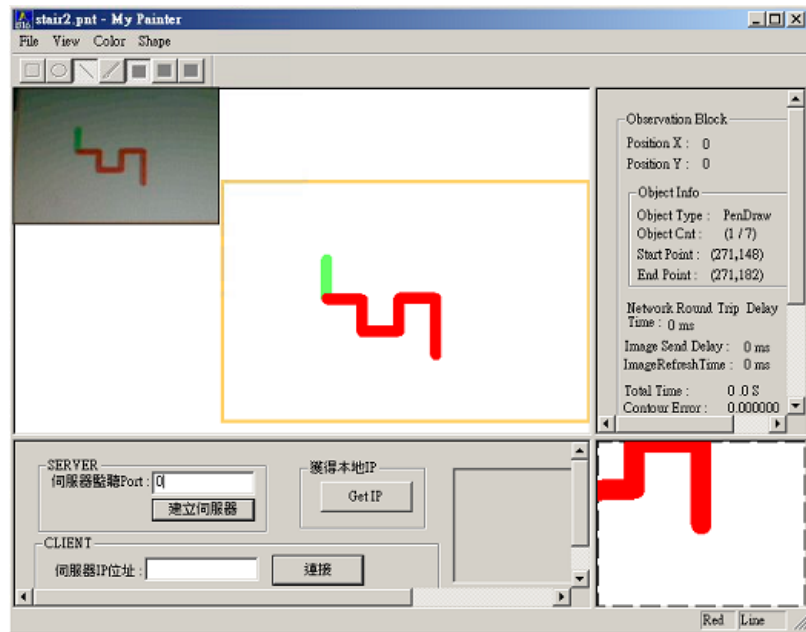
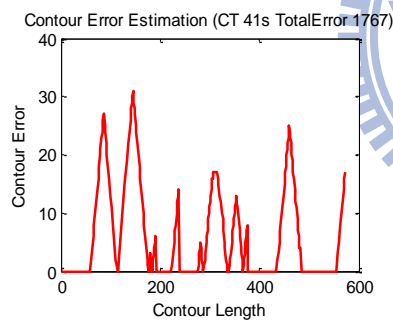


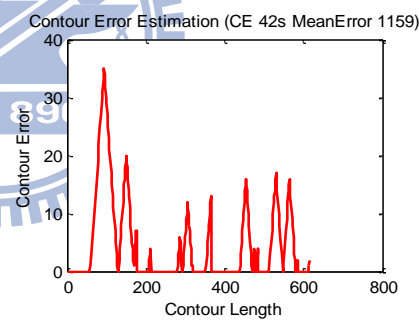
圖 6-20 長度短且轉折點多之循軌實驗

首先考慮未加入微調切換輔助機制，此時僅使用操作點預測輔助機制之實驗

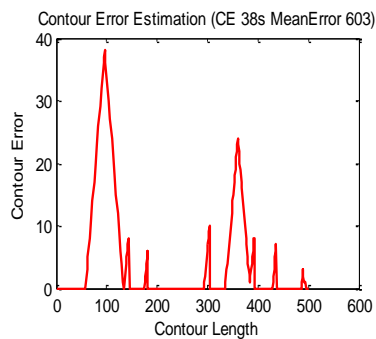
結果：



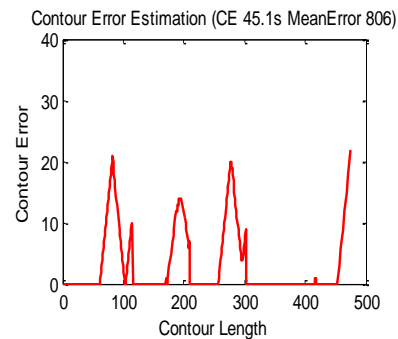
(a)



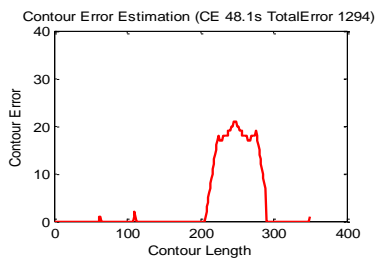
(b)



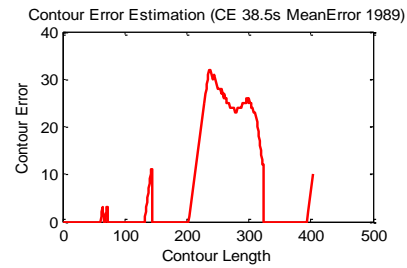
(c)



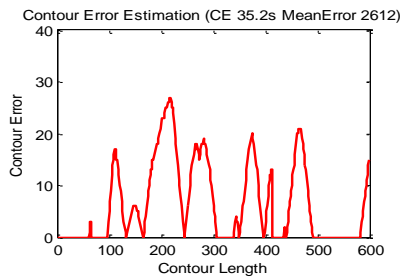
(d)



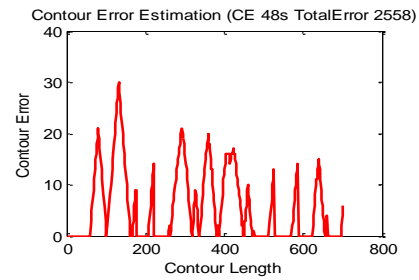
(e)



(f)



(g)



(h)

圖 6-21 僅使用預測點輔助機制之平均循軌誤差

表格 6-5 僅使用預測點輔助機制之平均循軌誤差數據

	Complete Time	Contour Error
Subject 1 exp1	41 sec	1767
Subject 1 exp2	42 sec	1159
Subject 2 exp1	38 sec	603
Subject 2 exp2	45.1 sec	806
Subject 3 exp1	48.1 sec	1294
Subject 3 exp2	38.5 sec	1989
Subject 4 exp1(excluded)	35 sec	83
Subject 4 exp2	48 sec	2558
Subject 5 exp1(excluded)	46.6 sec	4329
Subject 5 exp2	45.1 sec	806
Average	41.9 sec	1598.5



接著加入微調切換輔助機制，實驗結果如下：

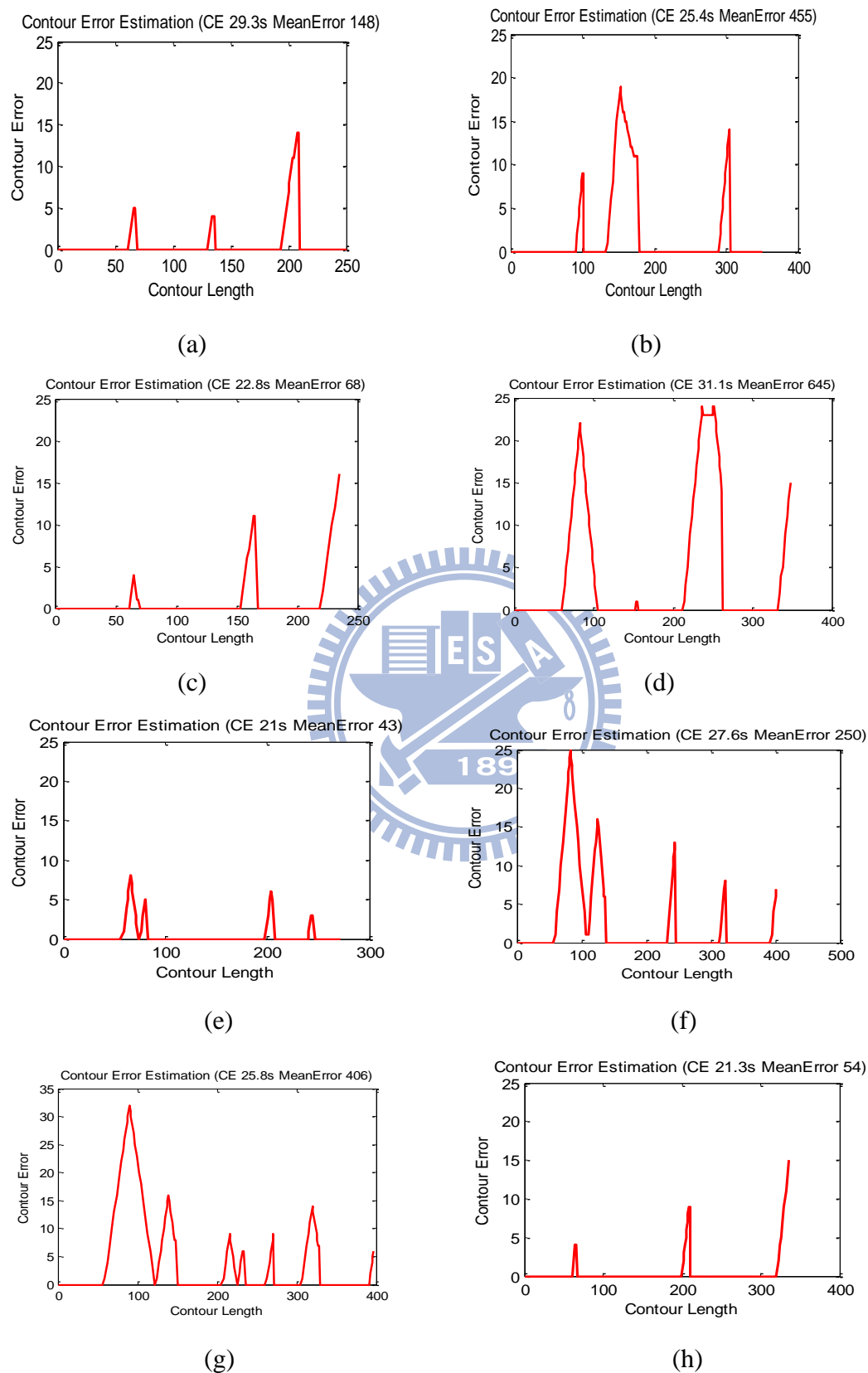


圖 6-22 加入微調切換機制之平均循軌誤差

表格 6-6 加入微調切換機制之平均循軌誤差數據

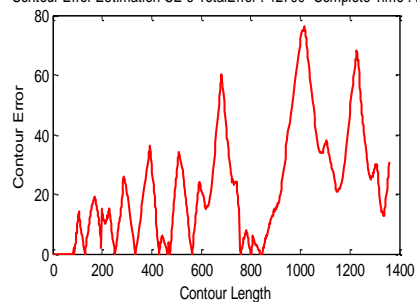
	Complete Time	Contour Error
Subject 1 exp1	25.4 sec	455
Subject 1 exp2	29.3 sec	148
<del>Subject 2 exp1(excluded)</del>	<del>24.2 sec</del>	<del>5</del>
Subject 2 exp2	22.8 sec	68
Subject 3 exp1	31.1 sec	645
Subject 3 exp2	21.3 sec	54
Subject 4 exp1	21sec	43
Subject 4 exp2	27.6 sec	250
Subject 5 exp1	25.8 sec	406
<del>Subject 5 exp2(excluded)</del>	<del>34 sec</del>	<del>1681</del>
Average	25.5sec	258.6

實驗結果證實，在循軌距離短且轉折點較多的情況下，微調切換機制可有效地降低循軌誤差高達 83.8%，並且提升 39.18% 完成時間，證明了微調切換機制在短距離循軌中之適應性。

#### 6.4.2 菱形斜線循軌實驗

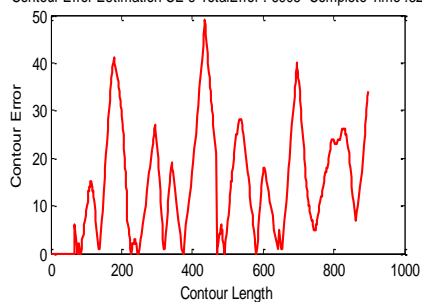
在斜線之循軌實驗中，當操作點落於欲追循線段終點時，系統即會判斷此線段循軌完成，進行下一線段循軌。在搖桿之操作命令不受限制的情況之下，操作點有可能在很短時間之內到達線段終點，但並未依照欲追循之軌跡。如此一來完成時間很短但循軌誤差卻極大，而此時完成時間便未具有循軌結果之指標性。所以在菱形斜線循軌物實驗中，只利用循軌誤差對系統進行評估。

Contour Error Estimation CE s TotalError : 12700 Complete Time : 41.1s



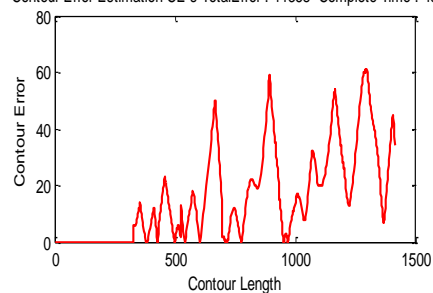
(a)

Contour Error Estimation CE s TotalError : 6905 Complete Time : 32.2s



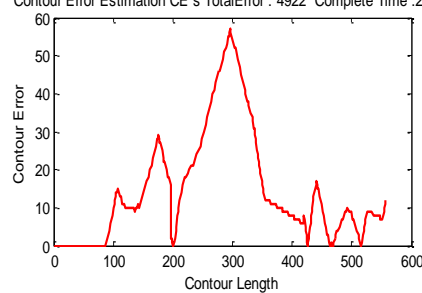
(b)

Contour Error Estimation CE s TotalError : 11358 Complete Time : 43.7s



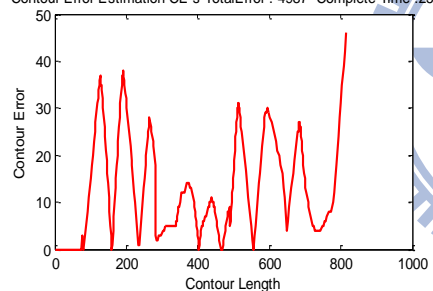
(c)

Contour Error Estimation CE s TotalError : 4922 Complete Time : 26.9s



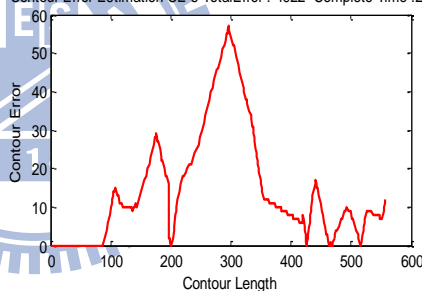
(d)

Contour Error Estimation CE s TotalError : 4987 Complete Time : 25.5s



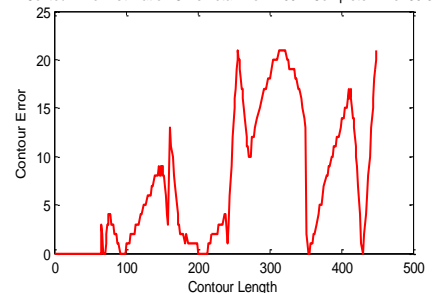
(e)

Contour Error Estimation CE s TotalError : 4922 Complete Time : 26.9s



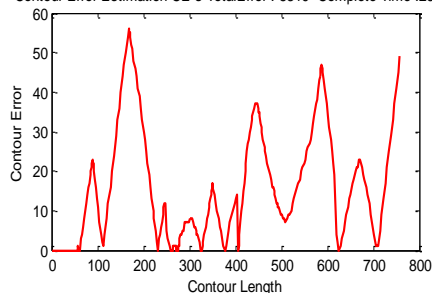
(f)

Contour Error Estimation CE s TotalError : 1891 Complete Time : 30.8s



(g)

Contour Error Estimation CE s TotalError : 5319 Complete Time : 23.5s



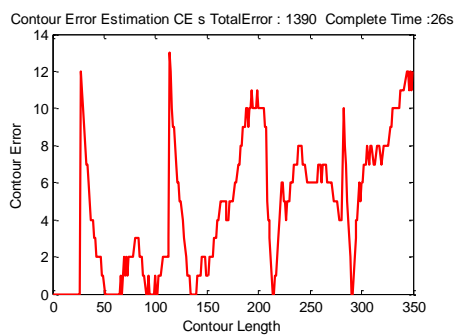
(h)

圖 6-23 未加入操作輔助機制之斜線平均循軌誤差

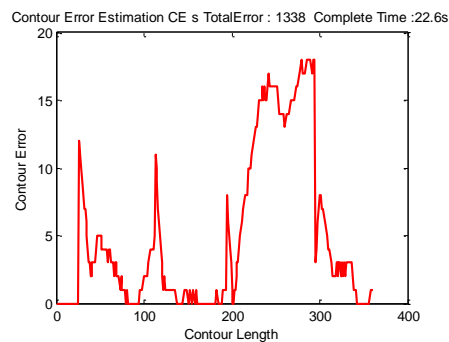
表格 6-7 未加入操作輔助機制斜線平均循軌誤差數據

	Contour Error
Subject 1 exp1	12700
Subject 1 exp2	6905
Subject 2 exp1	11358
Subject 2 exp2	4922
Subject 3 exp1(excluded)	13431
Subject 3 exp2	9833
Subject 4 exp1(excluded)	4442
Subject 4 exp2	4987
Subject 5 exp1	5525
Subject 5 exp2	5319
Average	7693.6

#### 6.4.2.1 虛擬操作點影像輔助機制

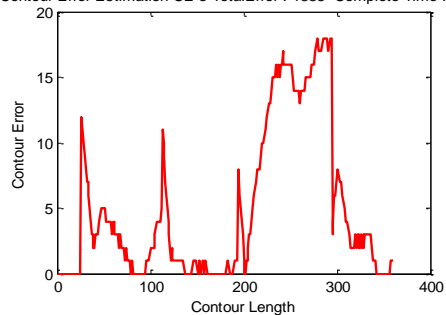


(a)



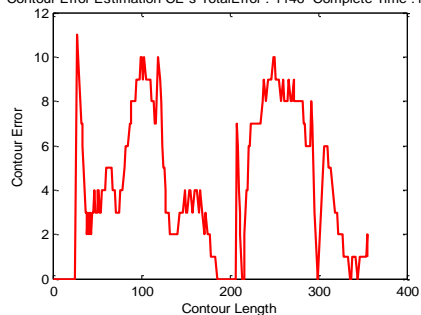
(b)

Contour Error Estimation CE s TotalError : 1338 Complete Time :23.4



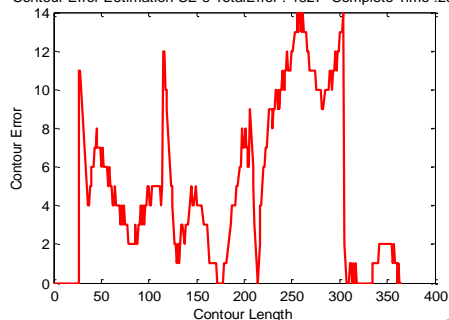
(c)

Contour Error Estimation CE s TotalError : 1146 Complete Time :19.9s



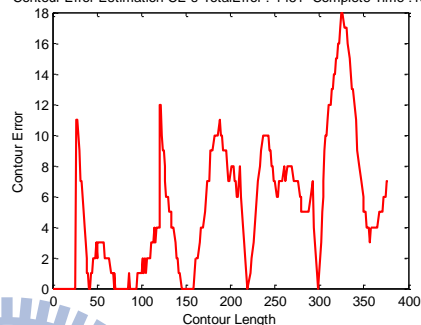
(d)

Contour Error Estimation CE s TotalError : 1627 Complete Time :28.2s



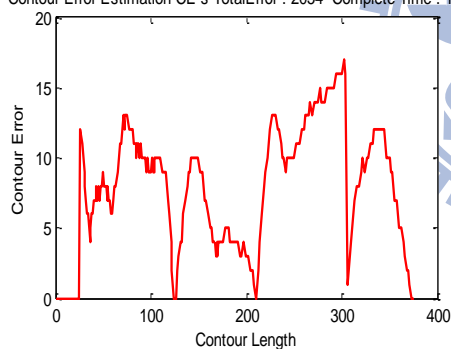
(e)

Contour Error Estimation CE s TotalError : 1451 Complete Time :19.2s



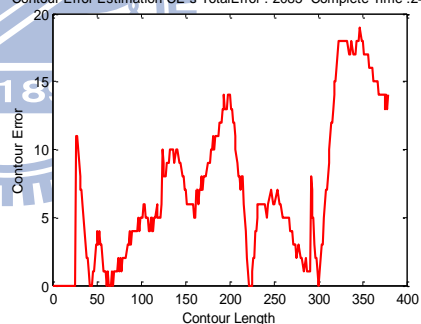
(f)

Contour Error Estimation CE s TotalError : 2054 Complete Time : 17.9s



(g)

Contour Error Estimation CE s TotalError : 2085 Complete Time :24.6s



(h)

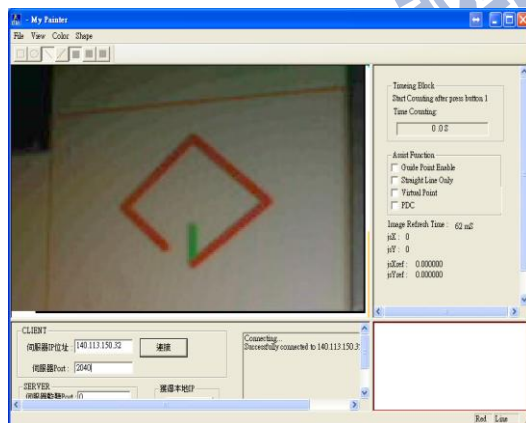
圖 6-24 加入操作點影像輔助機制之斜線平均循軌誤差

表格 6-8 加入操作點影像輔助機制於斜線平均循軌誤差數據

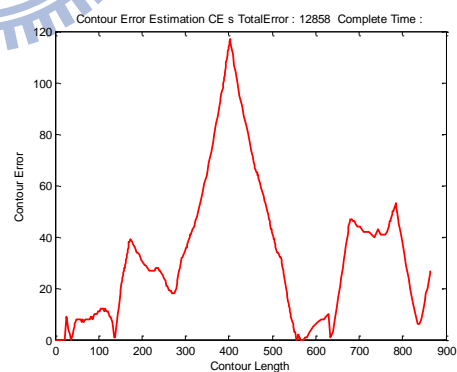
	Contour Error
Subject 1 exp1	1390
Subject 1 exp2	1338
Subject 2 exp1	1338

Subject 2 exp2(excluded)	3148
Subject 3 exp1(excluded)	996
Subject 3 exp2	1146
Subject 4 exp1	1627
Subject 4 exp2	1451
Subject 5 exp1	2085
Subject 5 exp2	2054
Average	1386.3

由實驗結果得知，使用虛擬操作點影像輔助機制後可將平均循軌從 7693.6 降低至 1386.3，可大幅降低循軌誤差，所以由實驗結果證明，此輔助機制於菱形斜線循軌中亦能有良好性能。但同樣的，在影像定位誤差的情況下，可以預期的是造成循軌誤差增加之情況亦會發生在菱形斜線循軌當中，由實驗結果亦顯示如此(圖 6-25)，所以接著使用操作點預測輔助機制，以改善影像定位所造成的誤差。



(a) 影像定位誤差

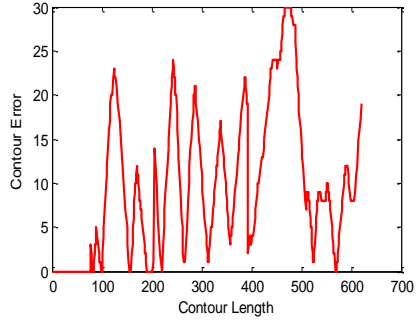


(b) 循軌誤差: 12858.0

圖 6-25 影像定位誤差下之循軌誤差

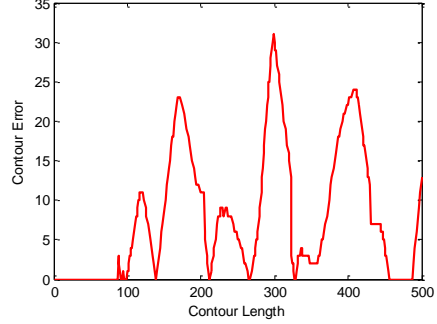
### 6.4.2.2 操作點預測輔助機制

Contour Error Estimation CE s TotalError : 4706 Complete Time :36.9s



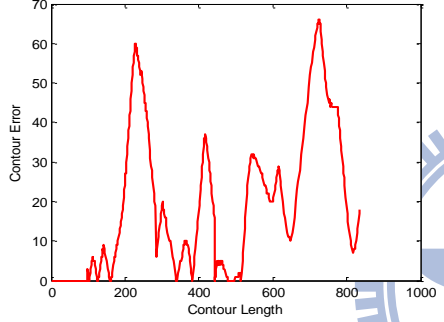
(a)

Contour Error Estimation CE s TotalError : 2803 Complete Time :27.4s



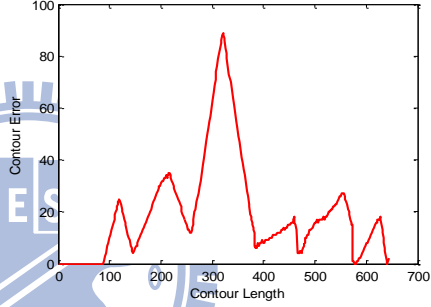
(b)

Contour Error Estimation CE s TotalError : 10729 Complete Time :38.2s



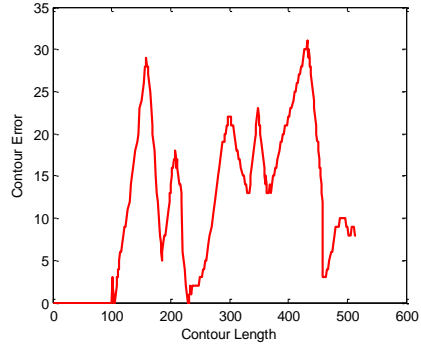
(c)

Contour Error Estimation CE s TotalError : 7447 Complete Time :41.1s



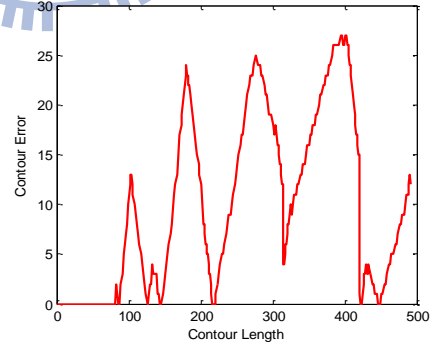
(d)

Contour Error Estimation CE s TotalError : 3979 Complete Time :34.2s

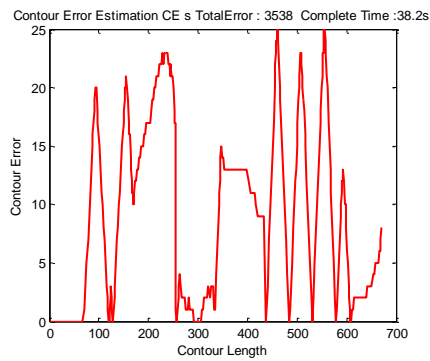


(e)

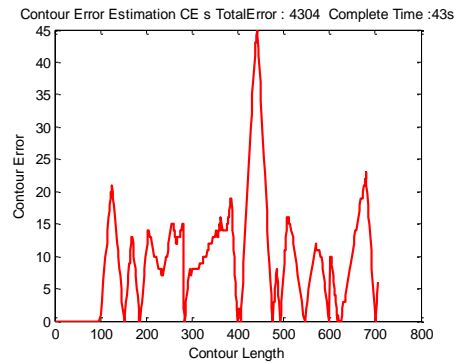
Contour Error Estimation CE s TotalError : 3098 Complete Time :32.1s



(f)



(g)



(h)

圖 6-26 加入操作點預測輔助機制之斜線平均循軌誤差

表格 6-9 加入操作點預測機制斜線平均循軌誤差數據

	Contour Error
Subject 1 exp1(excluded)	41367
Subject 1 exp2	4706
Subject 2 exp1	2704
Subject 2 exp2	10729
Subject 3 exp1	7447
Subject 3 exp2	3979
Subject 4 exp1	3098
Subject 4 exp2	3538
Subject 5 exp1	4304
Subject 5 exp2(excluded)	2561
Average	5063.1

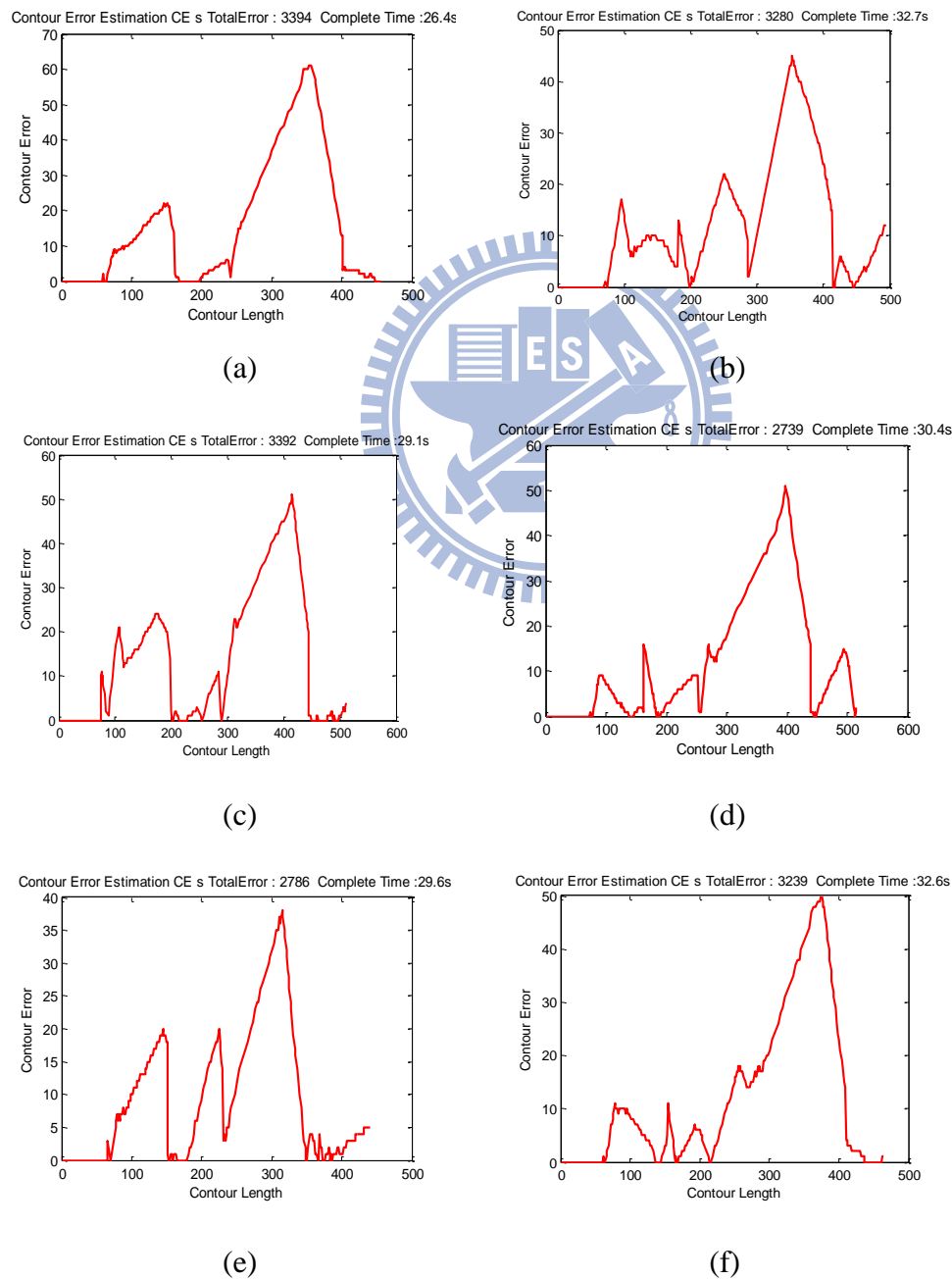
實驗結果顯示，操作點輔助機制在菱形斜線循軌中亦可改善平均循軌誤差，但改善幅度並不如方形直線循軌中來的顯著。其原因在於，菱形斜線循軌實驗中，搖桿之操作命令並未僅限於 X 方向或 Y 方向，如此一來操作點維持某個方向等速移動的機率會比方形直線循軌中來的小，一旦操作點維持相同方向等速移動之時間未超過網路往返延遲時間時，操作者所看到的預測點出現的方向即會因為網

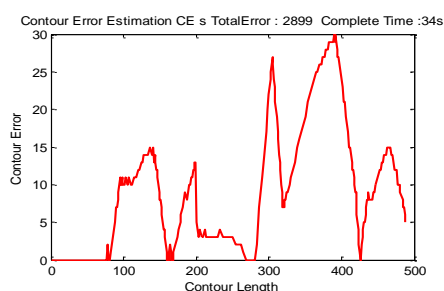


路影像延遲效應未如預期且即時性不足，進而無法有效降低透明度。所以在菱形斜線循軌中預測點輔助機制之效能比於方形直線循軌中低，而接下來加入之微調切換機制之重要性便相對提升。

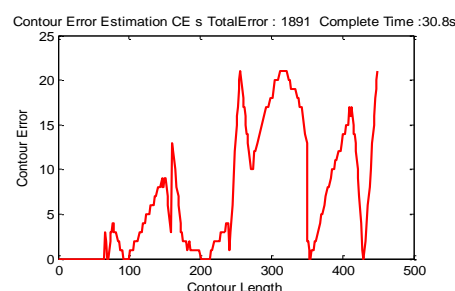
### 6.4.2.3 微調切換機制

在菱形斜線循軌實驗中，操作點預測機制之效能因為搖桿之自由度增加而減少，所以在此微調機制之重要性增加，也更能發揮其功效。





(g)



(h)

圖 6-27 加入微調切換機制之斜線平均循軌誤差

表格 6-10 加入微調切換機制斜線平均循軌誤差數據

	Contour Error
Subject 1 exp1(excluded)	5440
Subject 1 exp2(excluded)	1562
Subject 2 exp1	3394
Subject 2 exp2	3280
Subject 3 exp1	3392
Subject 3 exp2	2739
Subject 4 exp1	2786
Subject 4 exp2	3239
Subject 5 exp1	2899
Subject 5 exp2	1891
Average	2952.5

加入微調切換機制後，可將循軌誤差進一步降低至 2925.5，由此實驗結果發現微調切換機制在菱形斜線循軌中，對循軌誤差的改善幅度較方形直線循軌中明顯，原因在於方形直線循軌中，操作點預測輔助機制可有效地改善影像延遲造成的透明度不足，但在菱形斜線循軌中由於搖桿的自由度增加而造成操作點往往未維持特定方向之等速操作，加上影像延遲效應，進而造成預測點輔助機制之效能受限，所以在此微調切換機制之重要性與改善效能便提升。

## 6.5 小結

針對上小節之實驗結果，在此進行整理比較。

1. 操作點預測機制於方形直線循軌中較菱形斜線循軌中具有較佳改善效果，原因為在菱形斜線循軌中，搖桿之自由度增加造成此預測機制必須維持同方向等速度之前提不易滿足，進而限制此輔助機制之效能。

2. 微調切換機制於菱形斜線循軌中較方形直線循軌中具有較顯著改善效果，因此微調切換機制於搖桿自由度較大之情況下，具有較良好改善效果。

3. 在短距離且多轉折點循軌中，加入微調切換機制對循軌誤差改善之效能更加明顯。

表格 6-11 操作點預測與微調切換機制於不同軌跡之效能比較

	方形直線循軌	菱形斜線循軌	短距離且多轉折點循軌
未加入輔助機制	4028.7	7693.6	
操作點預測 (影像定位誤差)	1602.6	5063.1	1598.5
操作點預測 + 微調切換機制 (影像定位誤差)	1421.2	2952.5	258.6

## 第七章 結論與未來發展

在遠端影像操作系統之底層控制上，本論文提出了整合 MPC 與 CDOB 及 PDC 切換之網路控制架構，在平均延遲時間 450 ms 的情形下有效地穩定系統，並在上層控制操作上，設計三項輔助策略克服網路延遲對系統透明度影響，大幅降低平均循軌誤差且縮短完成時間。

所達成之研究成果整理如下：

### 1. 解決網路延遲對底層控制系統響應造成之影響

#### (a) 通訊擾動估測器(CDOB)

通訊擾動估測器不需要網路延遲時間模型，透過將網路延遲效應視為系統擾動量估測後進行補償，可將系統響應振幅降低 42.4%，但由於實驗平台之取樣頻率不足造成估測值穩態誤差，進而將其架構簡化改良消除此穩態誤差，進一步提升穩定度。而系統加入改善的 CDOB 後，響應之振幅可降低 50.4%、收斂時間提升超過 68%。實驗證明 CDOB 在網路延遲時間劇烈震盪時之性能比 adaptive Smith predictor 較佳。

#### (b) 模型預測控制器(MPC)

模型預測控制器運用在網路控制系統中，結合 CDOB 估測之輸出迴授，可有效克服網路延遲對系統響應造成之影響，將收斂時間再提升 28.88%、振幅降低 25.6% 且 rise time 亦在合理範圍內，並且可根據系統模型對迴授進行估測。實驗亦證明 MPC 於網路控制系統中比 PI 控制器具有更佳之性能。

#### (c) 估測迴授訊號之權重分配機制

利用 MPC 與 CDOB 估測之迴授響應，選擇迴授訊號的最佳權重分配，並在迴授資料遺失時切換至只使用 MPC 估測之迴授，進一步將響

應振幅再降低 8.5%、收斂時間再縮短 66.5%且 rise time 依然在容忍範圍內。由實驗結果證明本論文提出之整合 MPC 與 CDOB 之權重架構，比 PI + adaptive Smith predictor、PI + CDOB 與 MPC+ CDOB 具有更佳響應。

## 2. 解決網路延遲對上層系統所造成之操作困難

### (a) 長距離移動操作

1) 底層採用 MPC+ CDOB 與估測迴授之權重分配網路控制架構，達到快速且穩定之系統響應

2) 並以兩個影像輔助機制，克服影像延遲造成之操作誤差

#### i. 虛擬操作點輔助機制：

將 3D 降為 2D 之虛擬操作點機制，於方形直線循軌中可大幅降低循軌誤差 98.9%與完成時間 55%，於菱形斜線循軌中亦可降低循軌誤差 81.9%，但在 Webcam 影像定位誤差時，須另有校正之機制。

#### ii. 操作點預測輔助機制：

整體考慮網路延遲與人因特性，改善 Webcam 影像定位誤差，建立其運動模式，於方形直線循軌中進一步降低循軌誤差 60%與完成時間 35.7%，於菱形斜線循軌中降低循軌誤差 34.2%。

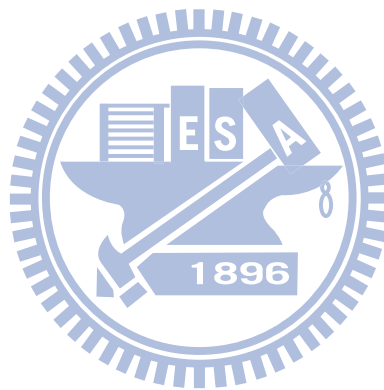
### (b) 短距離移動操作

操作點預測 + 微調切換機制：

1) 由於 PDC 之更快速響應特性，因此底層採用此網路控制架構，進行微調。

2) 系統影像操作於需要微調並可容忍震盪時，切換至 PDC 控制架構，可進一步於方形直線循軌中降低循軌誤差 83.8%、縮短完成時間 39.5%，於菱形斜線循軌中降低循軌誤差 61.6%。

本論文提出在取樣頻率不足時，CDOB 估測值穩態誤差之改善架構，接著利用 MPC 之移動時域控制，將兩者之估測迴授進行權重分配，並於資料遺失時切換權重分配，將系統響應調整至於長距離循軌中最適合操作之表現。最後利用操作輔助機制針對不同軌跡特性、操作者動態特性與延遲時間，改善循軌誤差並提升任務完成效率。



## 參考文獻

- [1] Goertz, R. C. “Electronically controlled manipulator.” *Nucleonics*, 1954.
- [2] 12(11), 46–47. Ferrell, W. R., & Sheridan, T. B. “Supervisory control of remote manipulation.” *IEEE Spectrum*, p.81–88, 1967.
- [3] Anderson, R. J., & Spong, M. W. “Bilateral control of teleoperators with time delay.” *IEEE Transactions on Automatic Control*, 34(5), p. 494–501, 1989.
- [4] Lawrence, D. A. “Stability and transparency in bilateral teleoperation.” *IEEE Transactions on Robotics and Automation*, 9(5), p.625–637, 1992.
- [5] Yokokohji, Y., & Yoshikawa, T. “Bilateral control of master–slave manipulators for ideal kinesthetic coupling—formulation and experiment.” *IEEE Transactions on Robotics and Automation*, 10(5), p. 605–620, 1994
- [6] Da Vinci Surgery, The da Vinci® Surgical System,  
<http://www.davincisurgery.com/>
- [7] Imaida, T., Yokokohji, Y., Doi, T., Oda, M., & Yoshikawa, T. “Ground-space bilateral teleoperation of ETS-VII robot arm by direct bilateral coupling under 7-s time delay condition.” *IEEE Transactions on Robotics and Automation*, 20(3), p. 499–511, 2004
- [8] Wang, W., & Yuan, K. Teleoperated manipulator for leak detection of sealed radioactive sources. *In Proceedings of the IEEE international conference on robotics and automation*, vol. 2, p. 1682–1687, New Orleans, LA, April 2004
- [9] Natori, K. & Ohnishi, K. “A Design Method of Communication Disturbance Observer for Time-Delay Compensation, Taking the Dynamic Property of Network Disturbance Into Account.” *Industrial Electronics, IEEE Transactions on*, vol. 55, p. 2152 – 2168, 2008.
- [10] 溫卓謀, “反應時間的概念及其在認知歷程研究上的應用 “.  
<http://www1.nttu.edu.tw/hepe/sfm/learning/Lpaper/lp02.htm>

- [11] 鄭景文, “動態網路控制系統之時間延遲分析,” 交通大學, 碩士論文, 中華民國 95 年十月
- [12] <http://www.rfc-editor.org/>
- [13] 孫海民, “精通 Windows Sockets 網路程式設計 使用 Visual C++”, 碁峰資訊股份有限公司, 2009.
- [14] Chien-Liang Lai & Pau-Lo Hsu “Design the Remote Control System With the Time-Delay Estimator and the Adaptive Smith Predictor, ” *Industrial Electronics, IEEE Transactions on*, vol. 6, p. 73 - 80, 2010.
- [15] Ramani Duraiswami, ”Camera Calibration”  
<http://www.umiacs.umd.edu/~ramani/cmsc828d/lecture9.pdf>
- [16] Liuping Wang , “Model predictive control system design and implementation using MATLAB,” London : Springer London, 2009.
- [17] 位元文化, “精通 MFC 視窗程式設計 Visual Studio 2008 版”, 文魁資訊股份有限公司, 2008.
- [18] P. F. Hokayem and M. W. Spong, “Bilateral teleoperation: An historical survey,” *Automatica*, vol. 42, no. 12, p. 2035–2057, 2006