

以干擾觀測器補償永磁式線性馬達之鈍齒效應

包含DSP即時多工控制系統之移植

Cogging Force Compensation in Linear-Motor-Driven Motion System

by Using DOB Structure and Porting of DSP-Based Control System

研究生：黃大維

Student : Da-Wei Huang

指導教授：李安謙

Advisor : An-Chen Lee

國立交通大學



A Thesis

Submitted to Department of Mechanical Engineering
College of Engineering

National Chiao Tung University
in partial Fulfillment of the Requirements
for the Degree of
Master
in

Mechanical Engineering

October 2007

Hsinchu, Taiwan, Republic of China

中華民國九十六年十月

以干擾觀測器補償永磁式線性馬達之鈍齒效應

包含DSP即時多工控制系統之移植

學生：黃大維

指導教授：李安謙 教授

國立交通大學機械工程學系

碩士論文



在本論文中主要研究運動控制卡即時多工系統的移植，並且補償永磁式線性馬達 (PMLM) 系統中 Cogging force 以及其他干擾所造成的影響。文中將採用德州儀器(TI)開發的 TMS320C6722 浮點運算 DSP 及使用 DSP/BIOS 取代先前的 $\mu C/OS II$ 作為即時核心，並搭配 TI 提供的 DSP 開發軟體 - Code Composer Studio 以及 Microsoft 的 Visual C++ 程式語言來移植即時多工核心程式，以進行即時命令產生、加減速規劃、Blended 路徑規劃及伺服迴路控制。Cogging Force 在馬達驅動控制中是很重要的干擾力，此外，系統還會受到其它干擾(如摩擦力、磁阻力、力矩波漣、輸出干擾等)、雜訊以及系統參數不確定性之影響而使機台無法達到精密定位，因此我們將討論伺服馬達鈍齒效應並進行補償。在本文中將以干擾觀測器(Disturbance Observer, DOB)的架構來補償永磁式線性馬達 (PMLM) 所受到干擾、雜訊以及系統不確定性造成的影響，並且提出實驗室發展的一套 DOB 改良架構 - DCF DOB 架構，來進行伺服迴路控制補償。

Cogging Force Compensation in Linear-Motor-Driven Motion System by Using DOB Structure and Porting of DSP-Based Control System

Student : Da-Wei Huang

Advisor : An-Chen Lee

Department of Mechanical Engineering
National Chiao-Tung University

ABSTRACT

This study focuses on two topics, 1) porting of the real-time multitasking motion control system for a new DSP board, 2) compensation of the cogging for permanent-magnet linear motors(PMLMs). The TMS320 C6722 DSP used in this study is a floating-point DSP which is produced by Texas Instruments(TI). The DSP/BIOS is a real-time kernel which is used to instead of $\mu\text{C}/\text{OS II}$ in the RTOS. There is a DSP development tool which is called Code Composer Studio(CCS), and CCS is provided by TI to help users develop the RTOS. Besides, Microsoft Visual C++ is used in the porting of the real-time multitasking motion control system. There are real-time command generation, acceleration and deceleration planning, and servo loop control implemented in the new RTOS. Cogging force of the permanent-magnet motors is an important disturbance for motion control system. Besides, there are friction force, torque ripple, output disturbance, measurement noise, and system uncertainties that decrease the positioning accuracy. Hence we have to compensate these disturbances with Disturbance Observer(DOB) and Doubly Coprime Factorization DOB(DCF DOB) which was developed by our lab in this study.

致 謝

首先非常感謝指導教授李安謙老師在這兩年的歲月中給予指導與教誨，在這些日子中，學生所學到的不只是書上的知識，還學到了許多做人處事的道理，讓我的碩士生涯不是只有唸書，而是『做人、做事、做學問』，給了我許多人生的啟示，受益良多。

感謝諸位口試委員在論文審閱之後給予的寶貴意見，使得本論文的内容得以更加充實與臻於完整。

感謝洪榮煌學長、潘怡仁學長、黎仁滄學長、黃淵勇學長、吳建峰學長、郭子偉學長以及張時中學長在我研究過程中對我的指導與建議。讓我體會當個研究生要持有一顆研究的心，對於所研究的問題，要能夠不斷的的思考與發掘，並能夠針對問題解決問題。

同時也感謝雷仕全學長、林峰龍學長、吳仲明學長、徐嘉星學長、柯璟銘學長以及張仲豪學長知識的傳承與教導。也感謝昶佑、明宗、忠聖、双偉、文凱同學在這兩年的研究生涯中彼此互相的扶持與鼓勵，讓我留下了一段值得留念的回憶。感謝宣宏、文昱、鼎凱以及正廉學弟在我的研究過程中給予的支持與幫忙，讓我得以順利的完成論文的工作。因為有你們的協助，讓我在研究的過程中成長許多，感謝你們！

最後，感謝爸爸、媽媽、叔叔、妹妹以及女友菁樺，在這段時間不斷的給予鼓勵與關懷。在此將本文獻給你們，也獻上我衷心的感謝，謝謝你們！

目 錄

第一章	緒論	1
1.1	研究目的與動機	1
1.2	文獻回顧	2
1.3	研究方法	4
1.4	內容大綱	5
第二章	硬體設備及架構	6
2.1	運動控制系統架構	6
2.2	實驗平台介紹	7
2.3	DSP 簡介	10
2.3.1	DSP 的特性及應用領域	11
2.3.2	Real time 的操作	11
2.3.3	容易改變參數	11
2.3.4	高穩定性的系統	11
2.3.5	TMS320C6722 數位訊號處理器	12
第三章	運動控制卡 RTOS 移植	14
3.1	軟體架構	14
3.2	RTOS 及 μ C/OS II 簡介	15
3.3	DSP/BIOS	18
3.3.1	硬體中斷(Hardware Interrupt)	19
3.3.2	軟體中斷(Software Interrupt)	19
3.3.3	一般工作(Task, TSK)	20
3.3.4	背景緒程(Background thread, IDL)	21
3.3.5	DSP/BIOS 之 Preemption	21
3.3.6	DSP/BIOS Input/Output	23
3.4	Code Composer Studio (CCS)	26
3.4.1	DSP/BIOS Configuration Tool	28
3.4.2	DSP/BIOS Real-Time Analysis Tool	29
3.5	Task 規劃	34
3.5.1	即時控制命令(Real Time Command)規劃	36
3.5.2	加減速規劃	39
3.5.2.1	梯形加減速規劃(T-Curve)	39
3.5.2.2	正弦加減速規劃(S-Curve)	43
3.5.2.3	圓弧加減速規劃	50

第四章	永磁馬達之鈍齒效應.....	52
4.1	鈍齒效應之特性及現象.....	52
4.2	鈍齒效應的產生.....	54
4.3	鈍齒效應改善方法.....	59
4.3.1	結構上的改良.....	59
4.3.2	控制法則補償.....	60
第五章	系統控制架構.....	61
5.1	DOB(Disturbance Observer)架構.....	61
5.2	Doubly Coprime Factorization DOB 架構.....	66
5.3	伺服迴路控制架構.....	70
5.3.1	Feedforward Control.....	71
5.3.2	Velocity Loop Control.....	72
5.3.3	Position Loop Control.....	74
5.4	系統參數鑑別.....	75
5.5	Output Disturbance.....	82
第六章	實驗結果討論.....	89
6.1	考慮輸入干擾.....	90
6.1.1	速度追蹤實驗.....	90
6.1.2	改變不確定性實驗.....	95
6.2	考慮輸入干擾、輸出干擾.....	101
6.3	實驗問題討論.....	123
6.3.1	$\alpha - \beta$ filter 頻寬問題.....	123
6.3.2	驅動器飽和問題.....	124
6.3.3	控制器實現問題.....	126
第七章	結論與未來展望.....	130
7.1	結論.....	130
7.2	未來展望.....	131

圖 目 錄

圖 2.1 運動控制系統架構圖.....	6
圖 2.2 線性馬達平台以及以色列壓電馬達.....	7
圖 2.3 DSP 架構圖.....	10
圖 2.4 TMS320 C6722 DSP Block Diagram.....	13
圖 3.1 軟體架構圖.....	15
圖 3.2 Task States.....	16
圖 3.3 Multiple Tasks.....	17
圖 3.4 Task Preemption.....	17
圖 3.5 Threads in DSP/BIOS.....	19
圖 3.6 interrupt 執行狀態.....	20
圖 3.7 Task 執行狀態.....	20
圖 3.8 PIP 物件示意圖.....	24
圖 3.9 SIO 物件示意圖.....	25
圖 3.10 DSP/BIOS 與 Code Composer Studio 關係.....	26
圖 3.11 CCS DSP/BIOS 程式發展流程.....	27
圖 3.12 DSP/BIOS Configuration Tool 介面.....	29
圖 3.13 CPU Load Graph.....	30
圖 3.14 Execution Graph.....	30
圖 3.15 Host Channel Contol.....	31
圖 3.16 Message Log.....	31
圖 3.17 Statistics View.....	32
圖 3.18 RTA Control Panel.....	32
圖 3.19 Kernel/Object View.....	33
圖 3.20 Task 規劃圖.....	34

圖 3.21 對稱&非對稱 T-Curve 速度曲線.....	39
圖 3.22 對稱正弦速度曲線.....	43
圖 3.23 圓弧曲線.....	50
圖 4.1 Cogging Torque 示意圖.....	52
圖 4.2 鈍齒效應產生影響示意圖.....	53
圖 4.3 永磁馬達系統示意圖、圖 4.4 The'vinin 等效磁路.....	55
圖 4.5 加入 air gap 之模型及磁路示意圖.....	57
圖 5.1 DOB 方塊圖.....	61
圖 5.2 DOB 等效方塊圖.....	63
圖 5.3 Doubly Coprime Factorization DOB 架構.....	66
圖 5.4 伺服迴路控制架構.....	70
圖 5.5 前饋控制架構.....	71
圖 5.6 速度迴路方塊圖.....	72
圖 5.7 位置迴路方塊圖.....	74
圖 5.8 含 PI 控制器之速度迴路架構圖.....	75
圖 5.9 速度參考命令與輸出響應圖.....	78
圖 5.11 舊系統的摩擦力實驗曲線.....	81
圖 5.12 輸入干擾與輸出干擾.....	82
圖 5.13 以 DOB 架構對抗輸出干擾.....	82
圖 5.14 加入輸出干擾的 DCF DOB 架構.....	84
圖 5.15 設計 $I - QY_1\tilde{N}$ 呈現不同形式的 notch.....	85
圖 5.16 設計為 $Q_1(s)$ 時 $G_{d_oV}(s)$ 頻率響應.....	88
圖 5.17 設計為 $Q_2(s)$ 時 $G_{d_oV}(s)$ 頻率響應.....	88
圖 5.18 設計為 $Q_3(s)$ 時 $G_{d_oV}(s)$ 頻率響應.....	88

圖 5.19 設計為 $Q_4(s)$ 時 $G_{d_v}(s)$ 頻率響應	88
圖 6.1 PI 速度迴授控制	89
圖 6.2 傳統 DOB +PI 速度迴授控制	89
圖 6.3 DCF DOB+PI 速度迴授控制	90
圖 6.4 定電壓開迴路速度響應	90
圖 6.5 等速實驗速度響應	92
圖 6.6 速度響應誤差	92
圖 6.7 正弦速度響應	93
圖 6.8 正弦速度響應誤差	94
圖 6.9 加質量後正弦速度響應	96
圖 6.10 加質量後正弦速度響應誤差	97
圖 6.11 減質量後正弦速度響應	99
圖 6.12 減質量後正弦速度響應誤差	100
圖 6.13 Step DO(1)與正弦速度響應	103
圖 6.14 Step DO(1)之正弦速度響應誤差	103
圖 6.15 Step DO(2)與正弦速度響應	104
圖 6.16 Step DO(2)之正弦速度響應誤差	104
圖 6.17 Ramp DO(1)與正弦速度響應	106
圖 6.18 Ramp DO(1)之正弦速度響應誤差	106
圖 6.19 Step DO(2)與正弦速度響應	107
圖 6.20 Step DO(2)之速度響應誤差	107
圖 6.21 人工 DO 響應	109
圖 6.22 實驗八結果響應	109
圖 6.23 實驗八結果與 DO 之相對關係	110
圖 6.24 實驗八之速度誤差	110

圖 6.25 人工 DO 響應	112
圖 6.26 實驗九結果響應.....	112
圖 6.27 實驗九結果與 DO 之相對關係	113
圖 6.28 實驗九之速度誤差	113
圖 6.29 傳統 DOB $d_o \rightarrow V$ 之頻率響應	114
圖 6.30 Q_1 形式 $d_o \rightarrow V$ 之頻率響應.....	116
圖 6.31 Q_2 形式 $d_o \rightarrow V$ 之頻率響應	116
圖 6.32 Q_3 形式 $d_o \rightarrow V$ 之頻率響應	116
圖 6.33 Q_4 形式 $d_o \rightarrow V$ 之頻率響應	116
圖 6.34 實驗十速度響應誤差.....	116
圖 6.35 notch Q_1 之速度誤差響應	117
圖 6.36 notch Q_2 之速度誤差響應.....	117
圖 6.37 notch Q_3 之速度誤差響應	118
圖 6.38 notch Q_4 之速度誤差響應.....	118
圖 6.41 純 PI 響應之 FFT 分析.....	121
圖 6.42 傳統 DOB+PI 響應之 FFT 分析	121
圖 6.43 Notch1+PI 響應之 FFT 分析	121
圖 6.44 Notch2+PI 響應之 FFT 分析	121
圖 6.45 Notch3+PI 響應之 FFT 分析	121
圖 6.46 Notch4+PI 響應之 FFT 分析	121
圖 6.47 $\alpha - \beta$ filter 為 1 倍系統頻寬.....	123
圖 6.48 $\alpha - \beta$ filter 為 2 倍系統頻寬.....	123
圖 6.49 $\alpha - \beta$ filter 為 7、8、9 倍系統頻寬.....	124
圖 6.50 $\alpha - \beta$ filter 為 3、4、5 倍系統頻寬.....	124
圖 6.51 d_o 對於控制架構的影響	125
圖 6.52 不同控制參數的 D/A 電壓輸出.....	125

圖 6.53 傳統 DOB 以及 DCF DOB 之方塊圖	126
圖 6.54 控制架構實現之方塊圖	127
圖 6.55 理想數位化實現之 pole-zero map	127
圖 6.56 實際數位化實現之 pole-zero map (參數取 9 位數)	128
圖 6.57 實際數位化實現之 pole-zero map (參數取 15 位數)	128



表 目 錄

表 2.1 線性無刷馬達規格表	8
表 2.2 雷射位移計規格表	8
表 2.3 以色列壓電馬達規格表	9
表 2.4 VC33 與 C6722 之比較	12
表 3.1 Thread 間 preemption 的關係	22
表 3.2 pipes(PIP)與 streams (SIO)之比較	23
表 3.3 即時控制命令-基本設定	36
表 3.4 即時控制命令-運動命令	37
表 3.5 即時控制命令-其他命令	38
表 5.1 系統參數表	79
表 6.1 實驗一相關資料	91
表 6.2 實驗二相關資料	91
表 6.3 實驗三相關資料	93
表 6.4 實驗四相關資料	95
表 6.5 實驗五相關資料	98
表 6.6 實驗六相關資料	102
表 6.7 實驗七相關資料	105
表 6.8 實驗八相關資料	108
表 6.9 實驗九相關資料	111
表 6.10 實驗十相關資料	115



第一章 緒論

1.1 研究目的與動機

在工業發達、科技進步、生產自動化要求快速且精密的今天，為提升產品的競爭力，必須更加地提升生產效率及產品品質，因而高精密度的運動控制(Motion Control)成為不可或缺的一環。其所發展出的運動控制器也因此被廣泛的使用於工廠自動化及機電整合系統，如數值控制工具機、機器人、電子零件黏著機、汽車製造廠...等，諸如此類需要精密機構運動的伺服控制大都是藉助於多軸運動控制器。為了改善系統的性能，各種複雜的控制器法則不斷的增加，特別是為了達到多軸同動的要求，伺服系統所需要的高速運算能力也以倍數激增，使得傳統控制系統在執行速度及記憶體容量已不敷使用。

隨著半導體技術的快速發展，SOC(System On Chip)技術日漸的成熟，使得數位訊號處理器(簡稱 DSP)不斷的推陳出新，其性能也愈來愈強大。此類的 DSP 不但具有的高速運算速度、快速的定址能力、大量的記憶體空間、功能強大的指令系統、靈活的介面通訊能力，也讓以往不易實現的控制器如狀態觀測器、Notch 濾波器，都能由數位信號處理器單獨來實現，以 DSP 發展而成的數位系統，不僅改善數位系統的限制，大大改進系統性能，其具有可靠性高、可程式規劃、維護容易、易於模組化設計等優點，也使得在設計上提升許多效率。於此，以 DSP 為核心，發展製作高性能的運動控制器已成為國內外許多機構所努力的目標。

運動控制最主要的基本要求為多軸性及即時性，藉由 DSP 的運算能力，能夠幫助處理多軸運動時所需的大量運算數值，而在即時性方面，隨著嵌入式系統近年來的備受矚目，應用在嵌入式系統上的即時作業系統也相繼推陳出新，如微軟推出的 Win CE、開放原始碼的 Embedded Linux、可靠度高的 $\mu\text{C}/\text{OSII}$...等應用於嵌入式系統產品的核心。而 TI 公司的 DSP/BIOS 發展環境，就提供了一個多緒程(Multi-Thread)作業系統，有效地幫助使用者發展一個嵌入式即時系統，透過相關軟體的輔助，更能使數位訊號處理器的性能最佳化且開發方便。

在機台方面，由於永磁式線性馬達有摩擦力、鈍齒效應、磁阻力、系統參數飄移...

等機台本身的干擾及不確定性，影響了控制的精確性，因此需要設計控制法則將這些外部的干擾降到最低。在本論文研究的目的主要分成兩部分，第一部分：移植高性能的運動控制器至新的 DSP，可以使用在工具機、機器人等自動化機器，期望能達成即時的運動控制；第二部分：為了降低馬達鈍齒效應以及其他各種干擾對於系統影響，選擇適當控制架構抑制其效應，讓其影響降至最低，並將所設計的控制法則移植並實現在 DSP 上，以期能達成精密的運動控制。

1.2 文獻回顧

目前國內外研製的運動控制器可分成三種型式：第一種，運動控制型專用 IC。第二種，模組式的運動控制器。第三種，整合嵌入式系統控制器。其中第三種不但將控制介面電路與微處理器全部設計在同一張電路板上，可同時執行多軸之馬達運動控制，若搭配即時多工核心，不但可獨立運作，又可配合 PC 使用，以發揮最其大功用，此為目前發展之趨勢[1]。

目前工業界之運動控制卡種類繁多，但每一張運動控制卡皆有共同的特色，皆是使用了 DSP 作為其核心處理器，因為 DSP-based 運動控制卡有計算快速之優點，能提供高性能的伺服與運動軌跡控制。硬體設計方面，本實驗室之前曾發展過 DSP 伺服控制板，洪敬堯在[2]中提出主要架構為使用 TMS320VC33 為中央處理器，包含六組 16 bit DA/AD converter，一組 12 bit AD/DA 轉換、數位輸入/數位輸出，以及 4K 的雙埠記憶體(Dual-port ram)提供 PC 及 DSP 之資料讀取。所使用的 DSP 為一浮點運算處理器，採用 32 位元多匯流排(multi-bus)架構，浮點計算採用 40 位元擴展精度，具有以硬體實現浮點乘法器與運算邏輯位元，可大幅縮短浮點運算時間。此外，再搭配 FPGA、CPLD 來做 IC 硬體設計、I/O 界面設計等專司 DSP 與外部周邊的溝通橋樑。

若要發揮運動控制硬體的功能，就必須將運動控制理論與方法整合在一起。目前本實驗室在運動控制器的理論上已有豐富的成果，若能將發展出的理論應用於功能強大的 DSP 上，對於目前應用廣泛的 DSP 運動控制卡能有很大的效能提昇。以往，發展環境上多使用組合語言來設計，然而使用組合語言來開發 DSP 較無法讓設計者隨意的發

揮，且設計者必須事先經過長時間的訓練與學習，才能熟悉組合語言的編撰方式，具備發揮 DSP 強大功能的基礎。所以實驗室採用能以 C 語言設計的 MicroC/OS II 核心來發展 DSP 運動控制卡的應用程式，而本論文將使用 TI 所發展之 DSP/BIOS 作為即時核心來取代 MicroC/OS II，並搭配 TI 推出的整合發展軟體 Code Composer Studio 與以 C 語言來進行設計，並且就 DSP Based 運動控制系統中位置命令的產生、伺服迴路控制的設計及各程式的時程規劃進行移植。在移植中所使用的 DSP 為 TI 公司開發的 TMS320C6722 DSP，與實驗室先前所使用的 TMS320VC33 皆為 TMS320 系列的浮點運算數位訊號處理器，但 C6722 在運算能力、記憶體容量等特性上都較為優秀，這些特色不但方便控制理論在 DSP 上的實現，未來在擴充上也更加容易。

運動控制中有兩個重要因素，一個是路徑規劃，另一個是伺服迴路控制。在路徑規劃方面，位置命令將採用洪敬堯論文[2]提到之 Bicubic 曲線來產生速度及加速度平滑的位置命令。加減速方面，在洪敬堯論文[2]中也提出梯行曲線(Trapzoidal Curve)的速度路徑規劃非常適合應用於工業機器人，於是在本實驗室所發展的運動控制板使用此路徑規劃來給定機台的位置命令；在伺服迴路控制部份，雷仕全[1]採用了位置迴路的迴授控制器和前饋控制器、速度迴路的迴授控制器和前饋控制器、適應性強健控制器以及 Disturbance Observer 的干擾觀測器架構，之後將會依照[1]所設計出的控制器架構進行移植，並且使用傳統 DOB 以及實驗室發展的 DCF DOB 控制架構來抑制永磁式線性馬達之鈍齒效應。

前面提到的鈍齒效應對於馬達驅動控制來說，是影響系統響應非常重要的一個干擾，它是由於馬達永久磁鐵與之鐵心之間的交互作用力所造成，會為馬達的移動產生一干擾的吸引力。為了改善 cogging force 所造成的不良影響，許多人在研究中提出不同的概念及方式來達到改善、補償鈍齒效應的目的。基本上可以分為下面兩種：

1)結構上的改良

2)控制上的補償

例如，在[3]中提出改變馬達齒寬(Tooth width)的匹配，將組成 cogging force 的力量成分錯移一個相位，使其互相抵銷，以此來降低 cogging force；在[4]中則是改變馬達永

久磁鐵的對稱性，藉由避開永久磁鐵與鐵心的 alignment，降低 cogging force 的疊加效應；在[5]中則提出調整馬達之 Slot / Pole 比、使用斜槽取代傳統的直槽、改變 Pole 和 Slot 的外型、使用輔助槽或輔助極來改善馬達之 cogging，以上等等皆屬於結構上的改良。在控制上，我們可以經由建立系統鈍齒效應的 model 並透過迴授和 cogging force model 加以補償，或是透過 Robust control 及 Adaptive control 此類抵抗干擾或不確定性的控制法則補償之[6],[7]，除此之外，我們亦可使用 DOB(Disturbance Observer)來觀測、補償系統之干擾，將系統的干擾及不確定性一併排除，由 I/O 觀點來看，期望內迴路系統趨近於 nominal plant 以利於整個迴路控制的設計[8],[9]。

除了輸入干擾所造成的影響之外，輸出干擾對於系統的影響也是不可忽視的，因此文中我們也將討論輸出干擾對於系統的影響，並且以人工方式另外加入特定之輸出干擾，藉由 DOB 控制架構來抑制鈍齒效應以及各種干擾對系統的影響，另外，在文中將會提出實驗室所發展的 DCF DOB 架構[10]並進行實驗並期待能有良好的控制效果。



1.3 研究方法

本論文主要是以 DOB 的架構來補償永磁式線性馬達的鈍齒效應，並且將以 TI 公司開發的 TMS320C6722 數位訊號處理器加以實現，並對本實驗室先前研製的高性能即時多工運動控制卡進行移植。因此本論文研究內容包括：

<I> 在軟體移植方面，將以 DSP/BIOS 取代先前的 MicroC/OS II 所撰寫的 DSP 之作業系統，並且更改相關的記憶體配置及設定；對 MicroC/OS II 及 DSP/BIOS 語法上的差異進行修改，架構上的差異進行取代；將對及時命令以及路徑加減速規畫方式加以修改並且對 Task 規劃進行修改。以期望能有更順暢的命令產生，更有彈性的加減速規劃。

<II> 在運動控制架構方面，本論文主要的控制架構為內迴路 DOB 搭配外迴路控制器進行運動控制補償。從 DOB 發展至今，不少學者研究 DOB 的特性，並指出

DOB 擁有優越的性能，因此在本論文將採用 DOB 的控制架構，但 DOB 也有其缺點存在，在文中將使用 DCF DOB 架構來改善幾項 DOB 之缺點並進行更彈性的設計，期望能有更好的控制效果。在實驗機台方面使用永磁式同步線性馬達作為測試平台，機台本身擁有的鈍齒效應，在文中將會以控制的方式補償，並以人工方式加入輸出干擾來驗證有無 DOB 架構系統之追蹤效能。

1.4 內容大綱

在本文中分為五章節：

第一章：『緒論』，包括研究的動機與目的、參考的文獻回顧和研究方法。

第二章：『硬體設備及架構』，說明在本文中的硬體架構、實驗機台，並簡單介紹 DSP 之特性、應用、操作等等。

第三章：『運動控制卡 RTOS 移植』，包括軟體架構、即時作業系統的介紹，說明本文所應用的 real time kernel - DSP/BIOS 的特色，以及整合開發軟體 Code Composer Studio，以及系統中 Task 規劃及加減速規畫方式。

第四章：『永磁馬達之鈍齒效應』，將對永磁馬達系統中鈍齒效應的產生原理及概念作簡單的說明、推導，並且介紹目前改善鈍齒效應之方法。

第五章：『系統控制架構』，介紹 DOB 之架構以及傳統 DOB 之優缺點，提出實驗室發展的 DCF DOB 架構，介紹系統控制迴路架構，最後討論輸出干擾對系統之影響。

第六章：『實驗結果討論』，呈現各種實驗之結果，比較有無加入 DOB 架構之差異，並且討論實驗之心得。

第七章：『結論與未來展望』，提出本文之總結，並且提出未來可能繼續發展的方向

第二章 硬體設備及架構

2.1 運動控制系統架構

整個運動系統，包括實驗硬體、軟體、運動控制系統、控制器架構等等，其中運動控制系統通常用於控制物體之位置與移動，如工具機床台與刀具的移動，而控制的方式通常仰賴位置控制迴路及伺服馬達速度控制。目前工業上所使用之伺服系統通常由多個迴路所組成，有內部的電流迴路用來控制馬達力矩，而外部的速度迴路則是用來調整馬達的速度。許多伺服驅動器皆已包含了速度迴路及電流迴路，並無法做設計上的更動。但是也有伺服驅動器僅含電流迴路者，因此運動控制卡上必須可以做位置迴路或是做位置及速度迴路，以便於與不同的驅動器銜接。

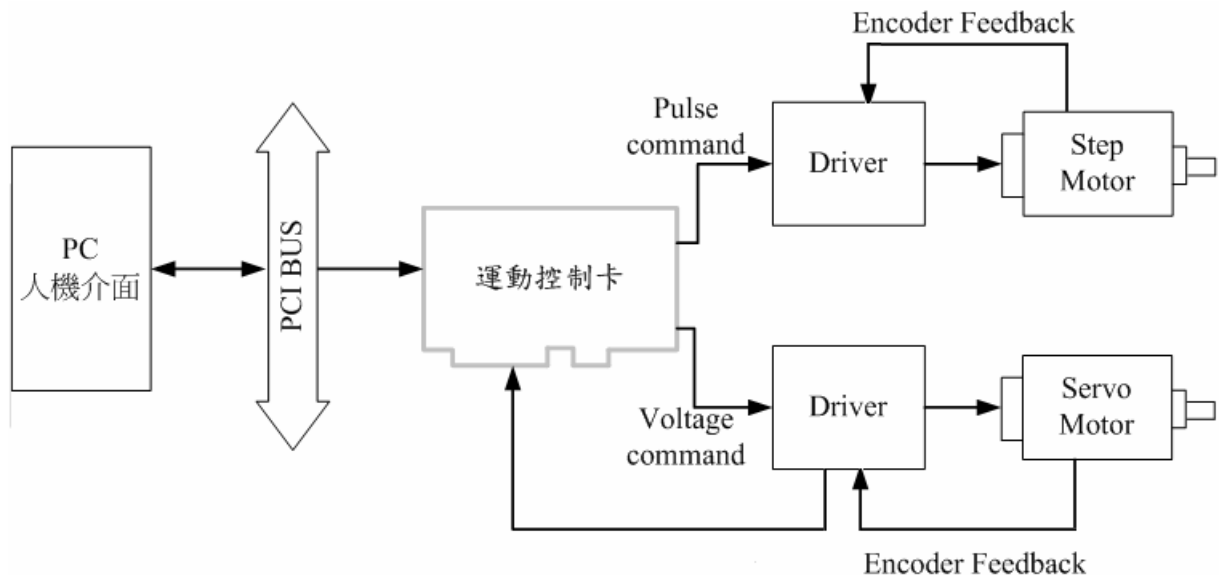


圖 2.1 運動控制系統架構圖

一般運動控制的程序大致如圖 2.1 所示，使用者在 Host PC 端下達路徑命令和需要的參數，因此 Host PC 必須能提供完整的人機介面讓使用者能容易完成輸入的工作。而 Host PC 也需要分擔運動控制卡一些較為繁雜的數學運算，如路徑規畫、粗差點計算等。接著再將粗差點資料送到運動控制卡做細插點的運算與伺服迴路控制，然後再將控制命令送到驅動器驅動馬達作動。而被控的機台也必須將目前的狀態如位置、IO 狀態等，回授給運動控制器和 Host PC 做命令修正和顯示之用。

2.2 實驗平台介紹

本論文中採用的實驗設備為永磁式同步線性馬達，所使用的驅動器是Kollmorgene公司的SERVOSTAR® CD Amplifier，為一數位式驅動器，提供位置、速度、扭力等模式，可根據使用者需求藉由RS232介面更改參數及運作模式。而線性馬達平台亦為Kollmorgen公司所出品的Platinum DDL系列之永磁式無刷線性導軌(IL6-050 A1)如圖2.2，其規格表如表2.1所示。此外，在平台量測方面採用光學尺以及雷射位移計(RLE10)來量測平台位移，其中光學尺精度為0.1 μm ，雷射位移計精度為0.02 μm 和0.079 μm 兩種精度。



圖 2.2 線性馬達平台以及以色列壓電馬達

在圖 2.2 中，我們可以看到另一組小型馬達架構在其中，此馬達為人工輸出干擾的產生因素，其採用的驅動器為 nanomotion AB1A 系列驅動器，其基本規格如表 2.3 所示。

PLATINUM DDL MOTOR (IL6-050 A1)		
Specification	Value	Unit
Force constant	28.5	N/Amp/ms
Back-EMF constant	23.3	V/ m/s
Continuous force	61	N
Peak force	200	N
Continuous current	2.1	Amp/ms
Peak current	7.0	Amp/ms
Electrical resistance	8.6	Ohms
Electrical inductance	3.0	Mh
Electrical time constant	0.3	Ms
Coil Assembly Mass	0.32	Kg
Magnetic way Mass	12.5	Kg

表2.1 線性無刷馬達規格表

Fiber Optic Laser Encoder(RLE10)	
Specification	Value - Unit
Axis travel	0~4 m
Linearity	< ± 10 nm
Accuracy (uncompensated)	50 ppm
Resolution	0.02 μ m (fine) 0.0791 μ m (coarse)
Zero point drift	100 nm/ $^{\circ}$ C

表2.2 雷射位移計規格表

Nanomotion MOTOR (AB1A)	
Specification	Value - Unit
Power Input	+48VDC \pm 5%
Max Motor Output	250-290Vrms
Power Consumption without Load	+48VDC/0.125A
Power Consumption with Max load	+48VDC/6.5Amax
Operating Temperature	0°C to 50°C
Storage Temperature	-40 to 70°C
Operating Humidity	Up to 80%

表2.3 以色列壓電馬達規格表



2.3 DSP 簡介

DSP(Digital Signal Processor, 數位訊號處理器)的基本架構如圖 2.3 所示,其設計在於以數位計算的方式進行訊號處理,因而先天上即具有強大的數值計算能力。目前的 DSP 可在一秒內完成數百萬到數千萬次高速、高精確度的乘法與邏輯計算,並可將程式與資料儲存於記憶體內。典型的 DSP 具有下列特性:程式匯流排與資料匯流排分開,指令 fetch 與執行具有平行處理能力,並具有高度的管線化(pipe-line)。具有硬體乘法器,能夠高速執行 MAC(Multiply And Accumulate)指令。簡單來說,DSP 是為了數位濾波器等經常使用乘加計算的應用,所特殊設計的處理器。

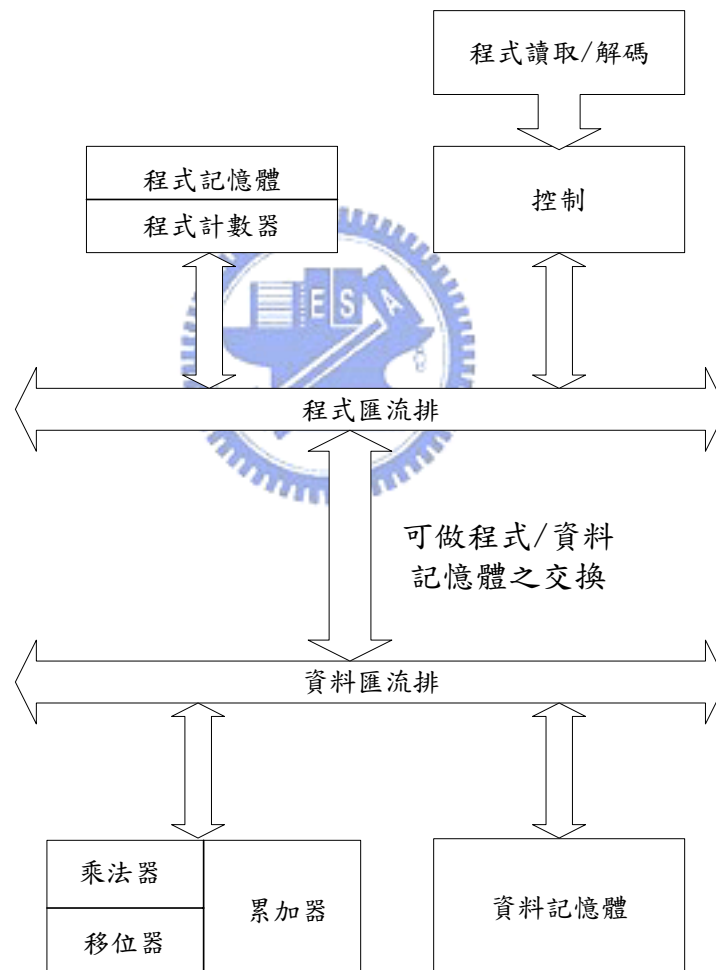


圖 2.3 DSP 架構圖

DSP 可將以往的類比系統以數位系統的方式實現,並利用軟體處理訊號的各種運算,例如:代數運算、數位濾波器、FFT、或是各種控制演算法。

2.3.1 DSP 的特性及應用領域

DSP 的應用通常有幾個共同的特性：

1. 需要大量數學計算的演算法。
2. 需要做即時(Real time)的操作。
3. 提供較佳的穩定系統。
4. 程式可重複設計化。

DSP 已經應用在相當大的領域，例如：數位濾波器、聲音的合成、語音的辨識、影像處理、FFT、伺服控制、調變解調器、影音壓縮解壓縮技術等。

2.3.2 Real time 的操作

DSP 必須有能力在設計的週期內讀取資料，完成計算，再輸出資料，否則 FIR 就無法工作。又以數位控制器為例，DSP 必須在下一個取樣時間(通常為 1 到幾個 ms)前完成所有的計算，否則無法即時輸出控制訊號，結果就是控制器無法工作。再以即時影像處理的應用為例，如果 DSP 無法在下一個更新週期內完成處理並輸出處理後的影像資料，就會產生影像閃爍、跳動或是一些影像不完整的問題。

DSP 的典型應用就是處理資料之數值計算，處理器必須能夠有效率的從外界擷取並處理大量的資料，還要能夠即時完成數學計算。

2.3.3 容易改變參數

類比濾波器是由電子零件所組成，如果想略微改變參數、調整系統特性，就必須調整電阻或電容；如果要做較大的特性改變，就必須重新設計電路。數位濾波器，如果必須改變參數，只需要修改程式，不需要改變電路。

2.3.4 高穩定性的系統

數位濾波器的核心是 DSP，演算法則以數字形態儲存於記憶體之中，不會隨著時間或環境的改變發生特性的變化。整個數位濾波器系統只有少數的類比元件如 A/D 轉換器與 D/A 轉換器可能受溫度改變而發生漂移，因此數位濾波器較類比濾波器有較高的穩定性，對於零件特性之差異也較不敏感。

2.3.5 TMS320C6722 數位訊號處理器

TI 公司所開發的 TMS320 系列之 DSP 種類非常繁多，有做 Digital Media 的 DM 系列、有 High Performance 的 C6x 系列、Power Efficient 的 C5x 系列，以及實驗室之前使用過的 VC33 的 C3x 系列。在 TMS320C6x 系列家族中，最先開發的是 C6201，是一顆整數運算 DSP。第二顆開發出來的是 C6701，它是一顆浮點運算 DSP，之後 TI 陸續開發出來的 C67x 系列浮點運算 DSP 的性能也日益提升，在本文中所使用的 C6722 DSP 與實驗室先前所使用的 VC33 其基本性能及規格上的差異，如表 2.2 所示。

		TMS320 VC33 - 150	TMS320 C6722 - 200
Cycle time		13-ns	5-ns
Frequency		75 MHz	200MHz
MIPS (Million Instruction Per second)		75	1600
MFLPS (Million Float Point operations per second)		150	1200
On-chip Memory		256 B Program Cache 136 KB RAM	32 KB Program Cache 128 KB RAM 384 KB ROM
Voltage	Core (V)	1.8 V	1.2 V
	I/O (V)	3.3 V	3.3 V
Timer		2 (32-bit)	2 (32-bit)
Peripherals	Memory access	DMA	dMAX
	EMIF		1 (16-bit) SDRAM
	Serial ports	1 serial port	2 McA serial ports 2 I2C serial ports 2 SPI ports

表2.4 VC33與C6722之比較

由表 2.4 可見 C6722 整體性能提升了許多，不僅運算速度及運算能力大幅提昇，內部記憶體亦多了 ROM 可供唯讀程式的儲存，擴大了記憶體的容量。在記憶體存取控

制採用了 dMAX(Dual Data Movement Acceleration)的方式，dMAX 跟 DMA 一樣都可以直接將資料存取以避免經過 CPU 而影響其工作效率，但 dMAX 有兩組暫存記憶體可以對資料進行三向傳輸，方便資料的分類及整理，此外，dMAX 可以同時接受兩筆資料(從不同的來源、往不同的目的地)的存取要求。對外界的溝通 VC33 只有一個 serial port 可以使用，而 C6722 提供了更多的溝通管道，包括以及 2 個 Multichannel Audio(McA) serial ports 用於 CODECs、DACs、CADs 的溝通、2 個 Inter-Integrated Circuit(I2C) serial ports 使 DSP 可透過 I2C 來控制外部裝置、2 個 Serial Peripheral Interface(SPI) ports 當 DSP 透過 I2C 控制外部設備時，可透過 SPI 對 DSP 下達命令。除此之外，C6722 還多了一組 16 位元的 External Memory Interface(EMIF)最高可支援 128Mb 的 SDRAM。C6722 的方塊圖如圖 2.4 所示。

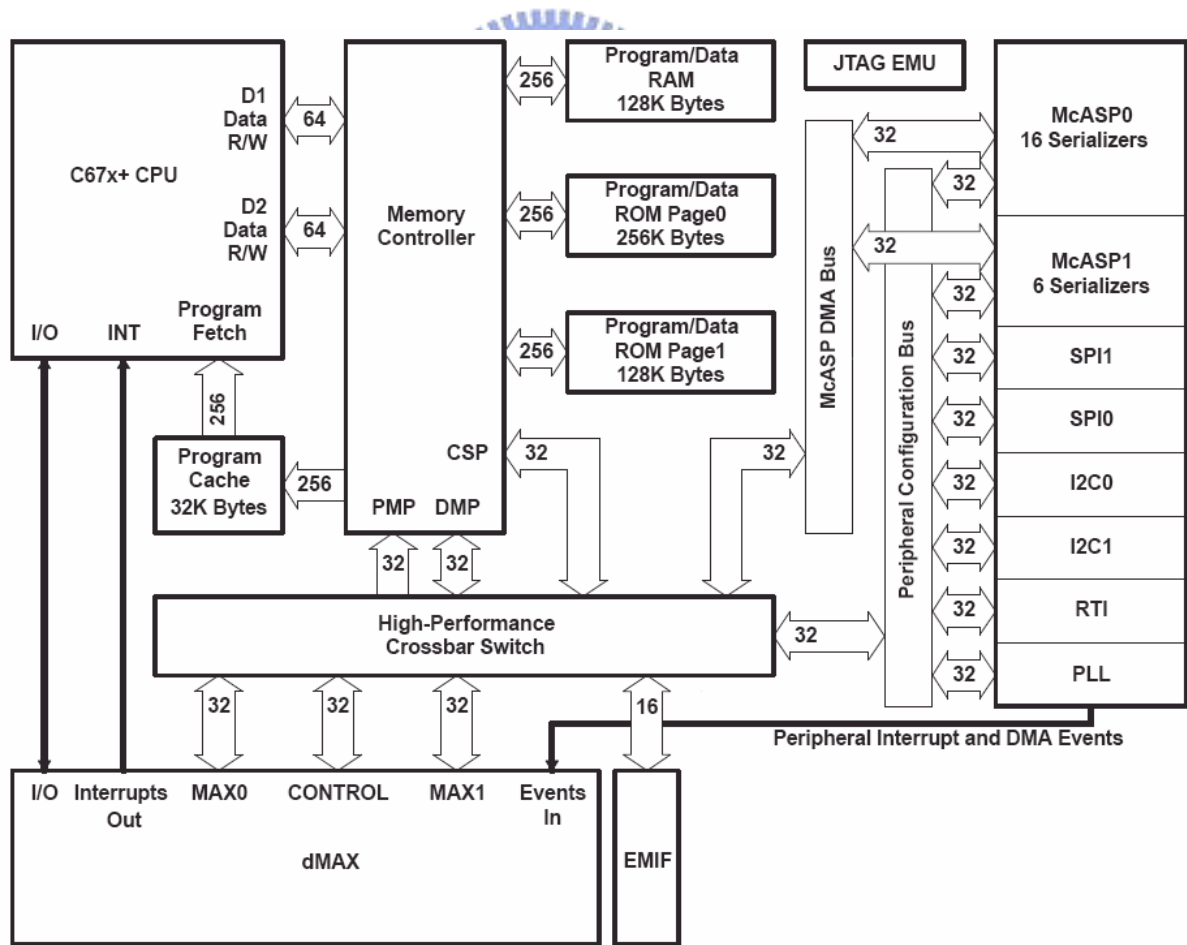


圖 2.4 TMS320 C6722 DSP Block Diagram

第三章 運動控制卡 RTOS 移植

3.1 軟體架構

本論文中之軟體架構沿用先前實驗室發展的 DSP 運動控制卡之軟體架構，著重於 DSP 運動控制卡上應用程式的發展及和 Host PC 之間資訊溝通與命令下達的方法與原則。要達到運動控制系統多軸即時運動的特性，除了硬體上的需求外，另一關鍵則在於系統本身是否具備即時效能與多工執行的能力，而本實驗室在 DSP 運動控制卡的發展上也有卓越的成就，目前已在 TMS320VC33 的 DSP 運動控制卡上建立了一即時多工系統，讓程式的發展能較有系統且配置每個工作的時程使其發揮較高的效能。程式的開發可以用組合語言和 C 語言來共同開發，使用組合語言的好處在於可以充分利用晶片的硬體特色，但開發速度較慢，程式的可讀性差；相對的 C 語言則有編程容易、除錯快速、可讀性好、能大幅縮短開發時間的優勢。

為了不佔用 DSP 太多的記憶體空間，影響其整體執行效率，以及便於日後程式的開發及擴充，實驗室先前採用符合上述要求的 $\mu\text{C}/\text{OS II}$ 來開發 TMS320VC33 運動控制卡，並成功的建立了一即時多工系統。本文中將以 DSP/BIOS 取代 $\mu\text{C}/\text{OS II}$ 作為即時系統的核心，將即時多工系統移植到 TMS320C6722 數位訊號處理器上。為了保留 $\mu\text{C}/\text{OS II}$ 的優點，如核心程式小、支援多平台、能以 C 語言開發，並且增加開發效率以及除錯能力，本文將採用 TI 公司的 DSP/BIOS 為主要開發環境，其特色有：

- ◆ 核心程式小
- ◆ 可搭配 Code Composer Studio 整合開發軟體提升開發效率
- ◆ 提供一套 PC 端即時資料交換的工具
- ◆ 能夠以 C 語言來開發

在 DSP 端主要執行三項工作：(1) 伺服迴路控制(Servo Loop Control)，進行機台定位控制；(2) 即時命令的產生(Generate Real-Time Command)，根據 PC 端的視窗介面命令的下達，分別產生對應的即時命令；(3) 路徑規劃(Trajectory Planning)：規劃梯型與 S 型加減速曲線、圓弧加減速、Blended 路徑。DSP 軟體的架構如圖 3.1 所示，在本文中主要進行 RTOS 以及 Application 的移植。在 RTOS 之下分成 Task 排程、Task 切換以及記憶體管理；在應用程式方面則分成數個 Task 各司其職，包括即時控制命令處理、DSP 狀態、伺服控制迴路、細插值計算……等等。

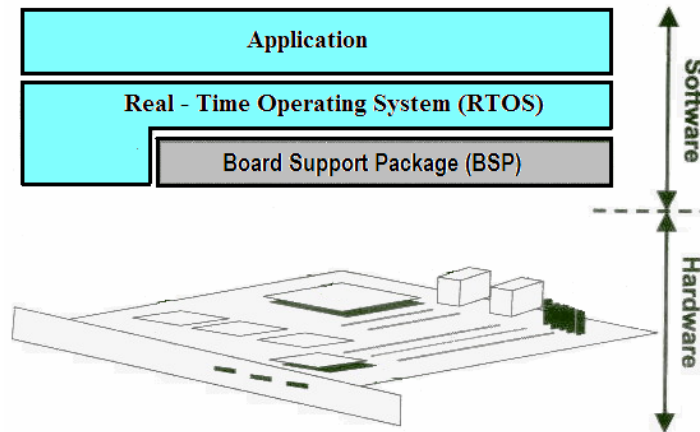


圖 3.1 軟體架構圖

3.2 RTOS 及 $\mu\text{C}/\text{OS II}$ 簡介

即時作業系統(RTOS)是一個程式，它按時序方式排程執行、管理系統資源，並且為開發應用程式碼提供一致的基礎。一個 RTOS 可以是各種模組的組合，包括核心(kernel)、檔案系統、網路協定、堆疊(stack)和應用要求的其他元件。

即時核心(real-time kernel)可以視為一個即時系統的心臟，大部分的即時核心都會包含下面的元件：

- ◆ Scheduler(排程器)：一組演算法，決定何時執行那個 task，常見的有 round-robin 和 preemptive 兩種排程。
- ◆ Object(物件)：特殊的核心構件，幫助建立 RTOS 的應用功能，包括 task、semaphore、queue。
- ◆ Service(服務)：在物件上執行的運算或通用的運算，如計時、中斷、資源管理。

$\mu\text{C}/\text{OS II}$ 為一個 portable、scalable、preemptive、Multi-tasking 的 real-time kernel，在運動控制卡的發展上，我們特別需要即時多工的性能，而 $\mu\text{C}/\text{OS II}$ 在即時多工的方面表現了相當優異的性能。

Task(工作元)也稱作 thread(執行緒)，是一個程式，該程式可以看成當它在執行時

CPU 完全只屬於自己。每個 Task 都是一個無窮的迴圈，並且都處在五種狀態之一的狀態下，如圖 3.2 所示，分別是休眠(DORMANT)、就緒(READY)、執行(RUNNING)、等待(WAITING)和中斷服務常式(ISR)這五種狀態。DORMANT STATE 相當於該 Task 仍存在於記憶體中，但並不被多工核心(kernel)所調度。READY STATE 意味著該 Task 已經準備好可以執行了，但由於該 Task 的 priority 比目前正在執行的 Task 低，所以在 Ready List 中就緒。RUNNING STATE 是指該 Task 得到了 CPU 的控制權，正在執行中。WAITING STATE 指該 Task 在等待某一事件發生，如外部的 I/O 訊號、共用資源或延遲時間的結束。最後，ISR STATE 是指中斷發生時，CPU 會提供相對應的中斷服務，而原來正在執行的 Task 將被停止進入了被中斷狀態。圖 3.2 表示 OS 所提供的函數服務，這些函數能使 Task 從某一狀態變到另一狀態。

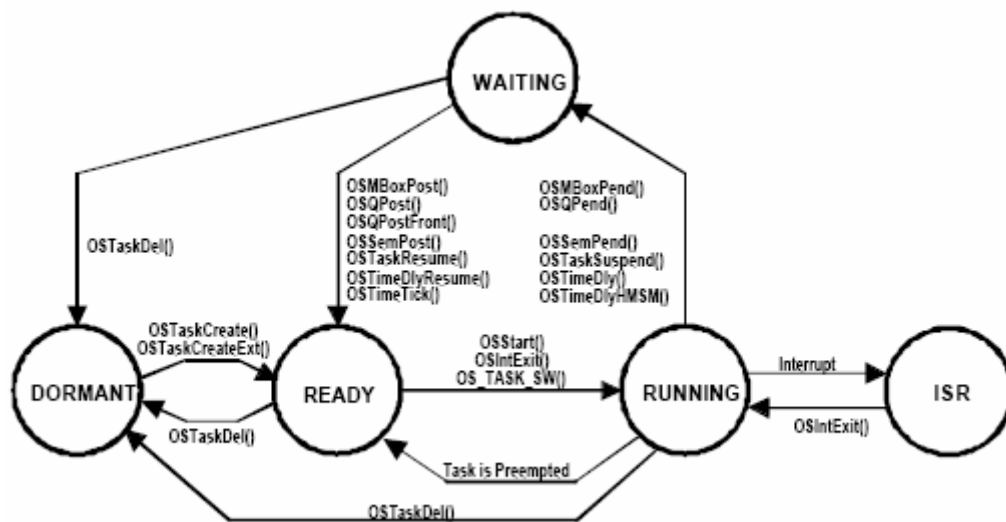


圖 3.2 Task States

多工(Multi-tasking)也就是即時應用程式在設計的過程中，會分成多個 Task，每個 Task 分別負責某個功能，全部 Task 的功能整合起來才是整個應用程式。當每個 Task 被創立時，Task 同時會被賦予屬於自己的 priority(優先權)、TCB(Task Control Block)及 stack(堆疊)空間，如圖 3.3。當 Task 被排程器依據 priority 排定執行後，CPU 會執行該 Task 所設計的功能，若此時被 priority 較高的 Task 打斷其進行，則會執行 Task 的 preemption，如圖 3.4，之後繼續執行原本的 Task 直到下一次的 preemption 或程式結束。

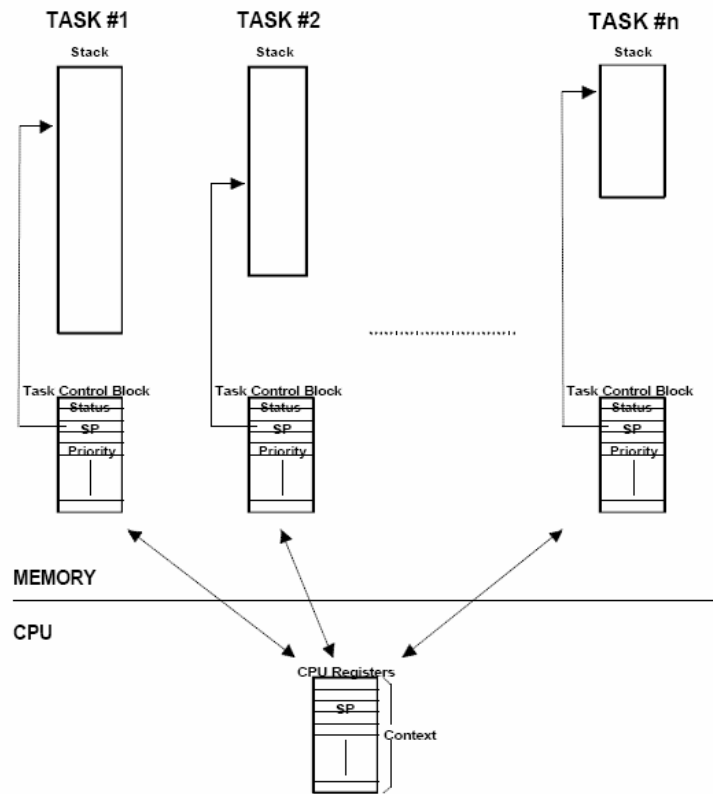


圖 3.3 Multiple Tasks

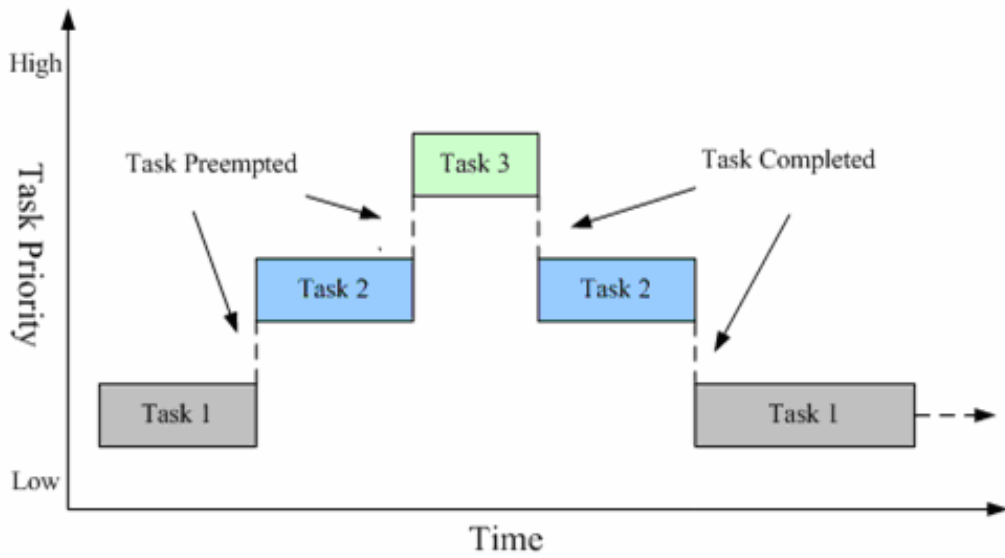


圖 3.4 Task Preemption

3.3 DSP/BIOS

DSP/BIOS 是一個 scalable、preemptive、multi-thread 的 real-time kernel，DSP/BIOS 可以用來設計需要 real-time scheduling、real-time synchronize、host-to-target communication 功能的應用程式，為 TI 公司 TMS320 系列的 DSP 提供了一個強而有力的發展環境。此外，DSP/BIOS 提供了下面幾種工具讓使用者方便設計即時作業系統[11]。

- ◆ DSP/BIOS API：

提供了一套應用函式庫，可以呼叫超過 150 個 DSP/BIOS functions。

- ◆ DSP/BIOS Configuration Tool：

此工具允許使用者靜態的建立並組態設定程式要使用的 DSP/BIOS 物件。如記憶體、執行緒優先權及中斷控制 (interrupt handlers) 等。

- ◆ DSP/BIOS Analysis Tool(Real-Time Analysis)：

允許使用者即時觀察系統的狀況以及程式的動作，如 CPU 使用率 (CPU load)、執行時序圖(execution graph)、執行記錄(logs)等等。

- ◆ RTDX(Real-Time Data eXchange)：

讓 DSP 端在應用程式執行時，可以與 Host PC 互相傳遞資料。

在 DSP/BIOS 中所提供的多緒程(Multi-Thread)系統，如圖 3.5，將緒程(Threads)依 priority 分為以下幾類：

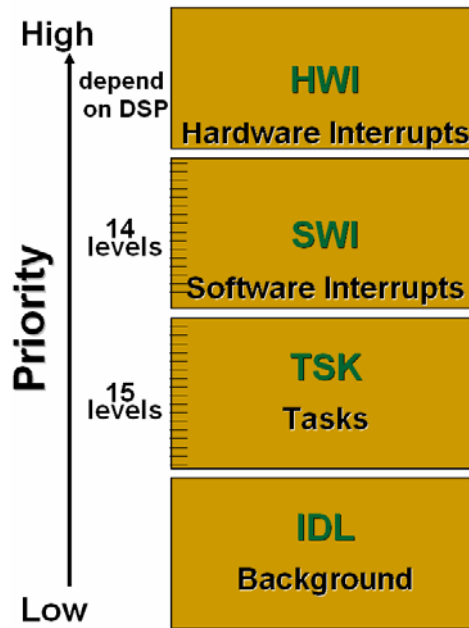


圖 3.5 Threads in DSP/BIOS

3.3.1 硬體中斷(Hardware Interrupt)

也就是硬體中斷服務函式(ISR)，是由 DSP 外部的硬體機構觸發的中斷，且 HWI 執行時不會釋出執行權(run to completion)，除非被更高優先權的 HWI 搶佔執行權，在 HWI 中的優先權定義取決於硬體的設定。此外 HWI 的 deadline 為 2~100 μ s，表示 HWI 的執行必須在 2~100 μ s 之間完成，否則可能會有硬體損害的危險。

3.3.2 軟體中斷(Software Interrupt)

在 DSP/BIOS 中 SWI 可以由靜態的方式(DSP/BIOS configuration)或動態的方式(SWI_create(attrs))來建立，屬於中斷的一種，是由軟體所觸發的中斷指令，其 Thread 的 priority 介於 TSK 與 HWI 之間，而 SWI 之中又分成 14 個 priority 可以區分 SWI 的優先順序，SWI 與 HWI 一樣，在執行時都是執行到緒程結束，除非被更高優先權的 SWI 或 HWI 搶佔其執行權。此外，SWI 的 deadline 大於 100 μ s，因此，對於執行時間要求較寬的中斷指令，就可以撰寫在 SWI 中。圖 3.6 表示 SWI 與 HWI 的執行狀態。

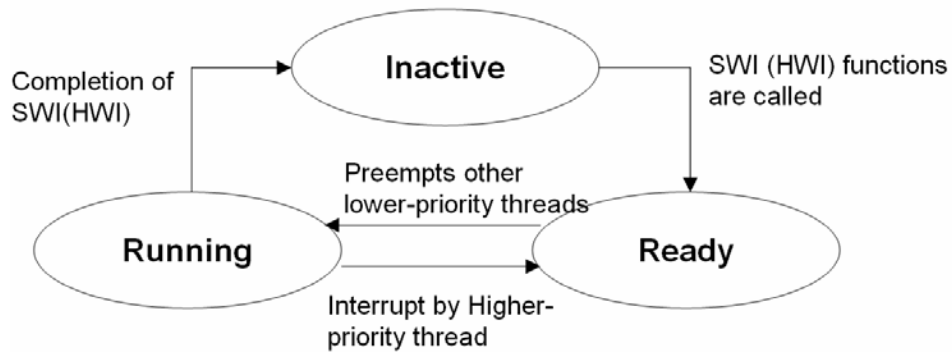


圖 3.6 interrupt 執行狀態

3.3.3 一般工作(Task, TSK)

在 DSP/BIOS 之中，TSK 可由靜態或動態的方式建立，我們可在 TSK 中撰寫程式的工作內容。在 TSK 之中分為 15 個 priority 等級，TSK 在執行時會被更高優先權的 TSK 或是被 SWI、HWI 搶佔其執行權，除此之外，TSK 在執行時也可以自行讓出執行權 (TSK_yield) 或是被暫停來等待 (wait) 其執行所需的一些必要資源，因此，DSP/BIOS 也提供一些 TSK 間的通訊及同步功能的模組，如 semaphore、queue、mailbox 等等。在 DSP/BIOS 定義之下，task 與 interrupt 已經分成不同的 thread，因此不需要用 task 來撰寫中斷，所以 Task 的執行狀態如圖 3.7 所示。

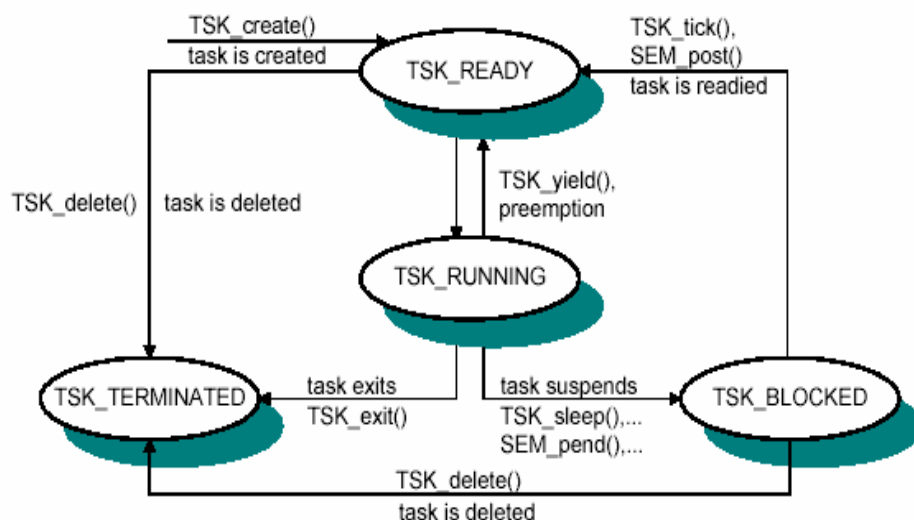


圖 3.7 Task 執行狀態

3.3.4 背景緒程(Background thread, IDL)

IDL 就是一個 idle loop，當沒有任何 HWI、SWI、TSK 執行的時候就會進入 IDL 之中，IDL 是一個無窮回圈，不斷的重複執行 IDL function，且一個 IDL function 必須執行到結束才可執行下一個 IDL function，當最後一個 IDL function 執行結束後將會重新執行第一個 IDL function，在 DSP/BIOS 定義的 thread 中，IDL 的 priority 最低，可以在任何時間點被任何的 thread (HWI、SWI、TSK) 搶佔其執行權，此外，DSP/BIOS 提供的 RTA(Real-Time Analysis)及 RTDX(Real-Time Data eXchange)都是在 IDL 之下執行的。

3.3.5 DSP/BIOS 之 Preemption

DSP/BIOS 核心是一個事件驅動(event-driven)的架構，DSP/BIOS Configuration Tool 就是用來幫助使用者在撰寫事件處理函式時，設定事件(在 Configuration Tool 中稱為物件，如 SWI、TSK、SEM、QUE 等等)的特性與參數。當一 Thread 在處理某件工作時，因為別的事件驅動了另一 Thread，此時 DSP/BIOS 提供的四個 Thread 的互相搶佔關係如表 3.1 所表示。

Thread post	Thread Running			
	HWI	SWI	TSK	IDL
Enabled,higher-priority HWI	Preempts	Preempts	Preempts	Preempts
Disabled HWI	Waits until reenabled	Waits until reenabled	Waits until reenabled	Waits until reenabled
Lower-priority HWI	Waits			
Enabled,higher-priority SWI		Preempts	Preempts	Preempts
Disabled SWI	Waits	Waits until reenabled	Waits until reenabled	Waits until reenabled
Lower-priority SWI	Waits	Waits		
Enabled,higher-priority TSK			Preempts	Preempts
Disabled TSK	Waits	Waits	Waits until reenabled	Waits until reenabled
Lower-priority TSK	Waits	Waits	Waits	
IDL	Waits	Waits	Waits	Waits

表3.1 Thread間preemption的關係

當 HWI 正在執行時，除了優先權較高的 HWI 可以搶佔其執行權之外，其他 thread 均要等其執行完才能執行，值得一提的是，如果遇上未啟動(Disabled)的 HWI 則必須等待其啟動後再依據優先權來決定是否搶佔目前的 HWI。

當 SWI 正在執行時，HWI 以及高優先權的 SWI 可以搶佔其執行權，低優先的 SWI 以及 TSK 都必須等待其執行結束才能執行，遇上未啟動的 SWI 也是待其啟動後在依據優先權決定是否搶佔目前的 SWI。

TSK 在執行時可以被任何的 HWI 或 SWI 或高優先權的 TSK 搶佔執行權，低優先權的 TSK 必須等待。

IDL 是沒有任何 Thread 執行時才會執行，它會被任何 Thread 在任何時間搶佔。

3.3.6 DSP/BIOS Input/Output

DSP 與輸入/出元件間一般都會加上一個資料緩衝器(buffer)，以增加系統效能。在 DSP/BIOS 中提供了兩種不同的輸入/出資料緩衝物件：pipes (資料導管，PIP Object) 以及 streams (資料流，SIO Object)。因為有資料緩衝器存在，資料的輸入/出物件都是非同步(asynchronous)的。PIP 物件之輸出資料會送到 HST 物件或從 HST 物件取得 (HST 物件則是 C6x 系列 HPI 輸入埠的驅動程式)。而 SIO 物件之輸出資料會送到 DEV 物件或從 DEV 物件取得(DEV 物件可以是一般的輸入元件驅動程式)。兩種資料傳送方式的比較如表 3.2 所示：

Pips (PIP and HST)	Streams (SIO and DEV)
Programmer must create own driver structure.	Provides a more structured approach to Device-driver creation.
Reader and writer may be any thread type or host PC.	One end must be handled by a task(TSK) using SIO calls. The other end must be handled by an HWI using Dxx calls.
PIP function are non-blocking. Program must check to make sure a buffer is available before reading from or writing to the pipe.	SIO_put, SIO_get, and SIO_reclaim are blocking functions and will cause task to wait until a buffer is available.(SIO_issue is non-blocking.)
Uses less memory and is generally faster.	More flexible; generally simpler to use.
Each pipes owns its own buffers.	Buffers can be transferred from one stream to another without copying.(In practice, copying is usually necessary anyway because the data is processed.)
Pipes must be created statically with the Configuration Tool.	Streams may be created either at run-time or statically with the Configuration Tool. Streams may be opened by name.
No built-in support for stacking devices.	Support is provided for stacking devices.
Using the HST module with pipes is an easy way to handle data transfer between the host and target.	A number of device drivers are provided with DSP/BIOS.

表 3.2 pipes(PIP)與streams (SIO)之比較

一般來說，PIP 物件是一個比較低階的輸入/出物件，它的資料傳輸是以資料框 (frame) 為單位，使用者必須呼叫多個處理函式一步步處理資料及 PIP 物件中的緩衝器才能完成資料輸入/出的動作，此即為上表所說的 non-blocking，雖然傳輸速度較快，但資料量小。而 SIO 物件為較高階、與元件無關(device-independent)的輸入/出系統，傳輸資料量大，使用上更有彈性，但傳輸速度較慢是其缺點。下面將對兩個 I/O 方式做進一步的介紹：

〈1〉 Pipe Object (PIP) :

pipe 物件是用來管理區塊資料輸入/出用的。在每一個 pipe 物件中含有一塊資料緩衝記憶體(buffer)，此 buffer 被分為固定個數的資料框(frame)，而每一個 frame 中含有相同的 byte 數。在 pipe 中每次輸入的資料都是以 frame 為單位。圖 3.8 為 PIP 物件的示意圖。

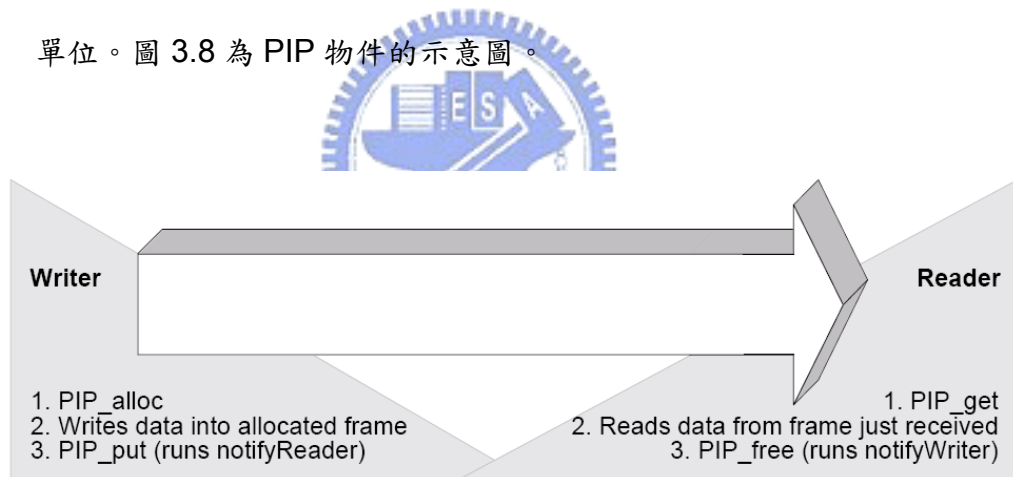


圖 3.8 PIP 物件示意圖

如上圖(3.8)所示，一個 pipe 的兩端分別為 Writer 端以及 Reader 端，Writer 端負責將資料寫入 frame 中，Reader 端則負責從 frame 中將資料讀取出來。

〈2〉 Stream Object (SIO) :

SIO 物件可將程式的輸入/出經過資料緩衝器連接至一個 DEV 物件，DEV 物件即為輸入/出元件的驅動程式(device driver)，如圖 3.9。因此，在 DSP/BIOS 下制定不同的 DEV 物件，我們的 SIO 便可以和不同的元件溝通，使得 SIO 成為一個與元件無關的輸入/出系統。

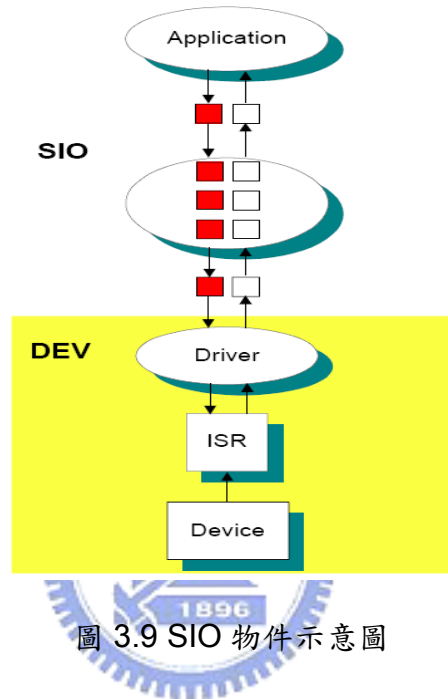


圖 3.9 SIO 物件示意圖

3.4 Code Composer Studio (CCS)

在 DSP/BIOS 的開發過程中可以搭配使用 Code Composer Studio 來提升開發效率，Code Composer 原先是一套由 Go-DSP 公司所發展的 DSP 整合式發展環境，曾經專門為 TI 公司 C6x 系列的 DSP 推出了 Code Composer emulator 4.0 版。後來 Go-DSP 在被 TI 公司併購之後陸續新版的 Code Composer 並且新增了 simulator 的功能，讓使用者可以純粹模擬程式的功能，隨著 Code Composer 版本的更新，也加入了 DSP/BIOS 即時作業核心的發展環境，因此，在 DSP/BIOS 的發展上，CCS 扮演著墊腳石的重要角色。

DSP/BIOS 與 CCS 的關係密切且相輔相成，DSP/BIOS 與 Code Composer Studio 的架構如圖 3.10 所示。

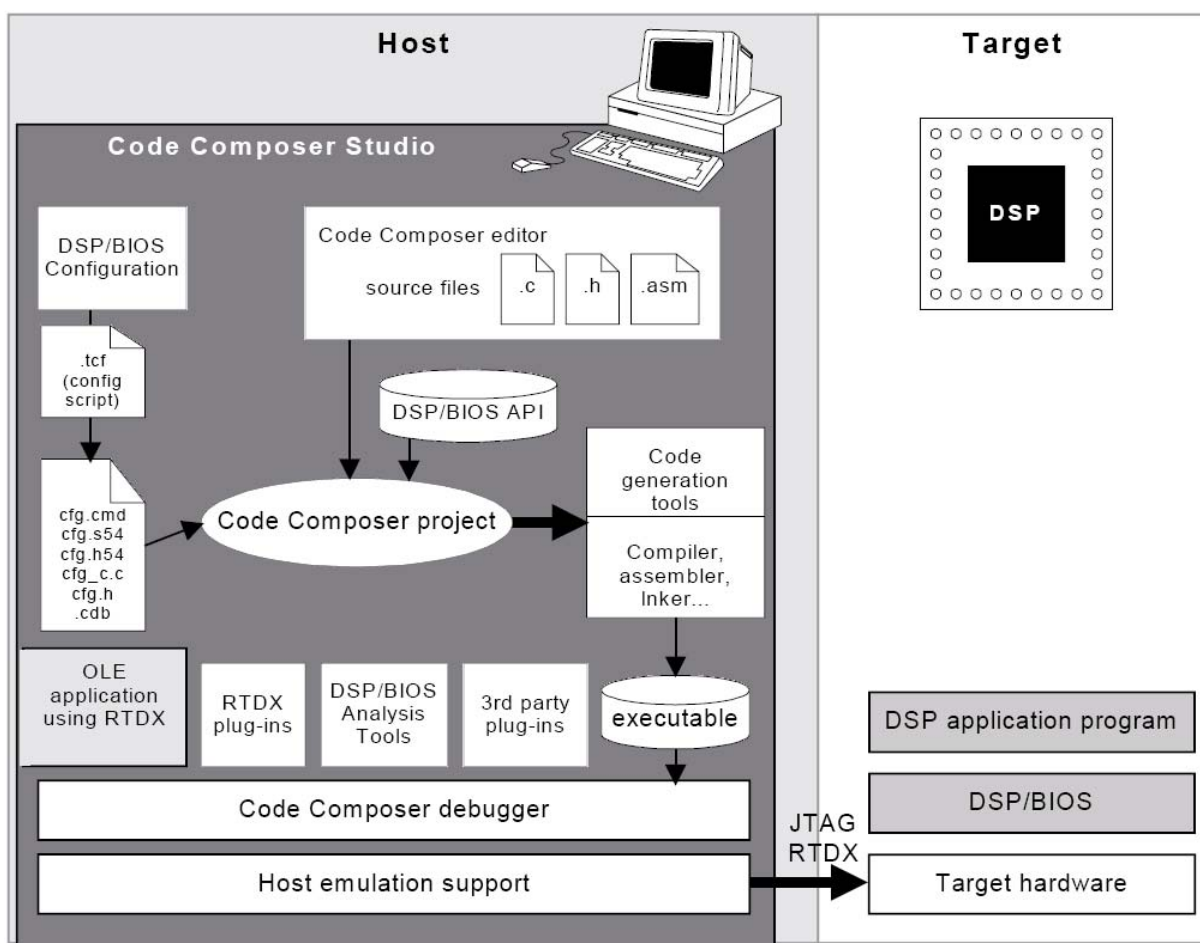


圖 3.10 DSP/BIOS 與 Code Composer Studio 關係

在 Host PC 端我們以 Code Composer Studio 靜態的來規劃即時系統的內容，以及設計 DSP/BIOS 應用程式。透過 DSP/BIOS Configuration Tool 規劃出來的物件會被 CCS 儲存成一組態設定檔，和使用 C 語言撰寫出來之函式及工作內容的 source code，以及在整個程式中會被使用的 DSP/BIOS API functions 一起組成 Code Composer project，再經由 CCS compiler、assembler 進行編譯、組譯後成為一個可執行檔(.out 檔)，並且透過 JTAG 傳輸方式將程式 Load 到目標的 DSP 上，建立起 DSP/BIOS 即時核心並執行應用程式。在程式執行時可透過 JTAG 來進行 RTDX(即時資料交換)或是 RTA(即時分析)，應用程式的發展流程如圖 3.11。

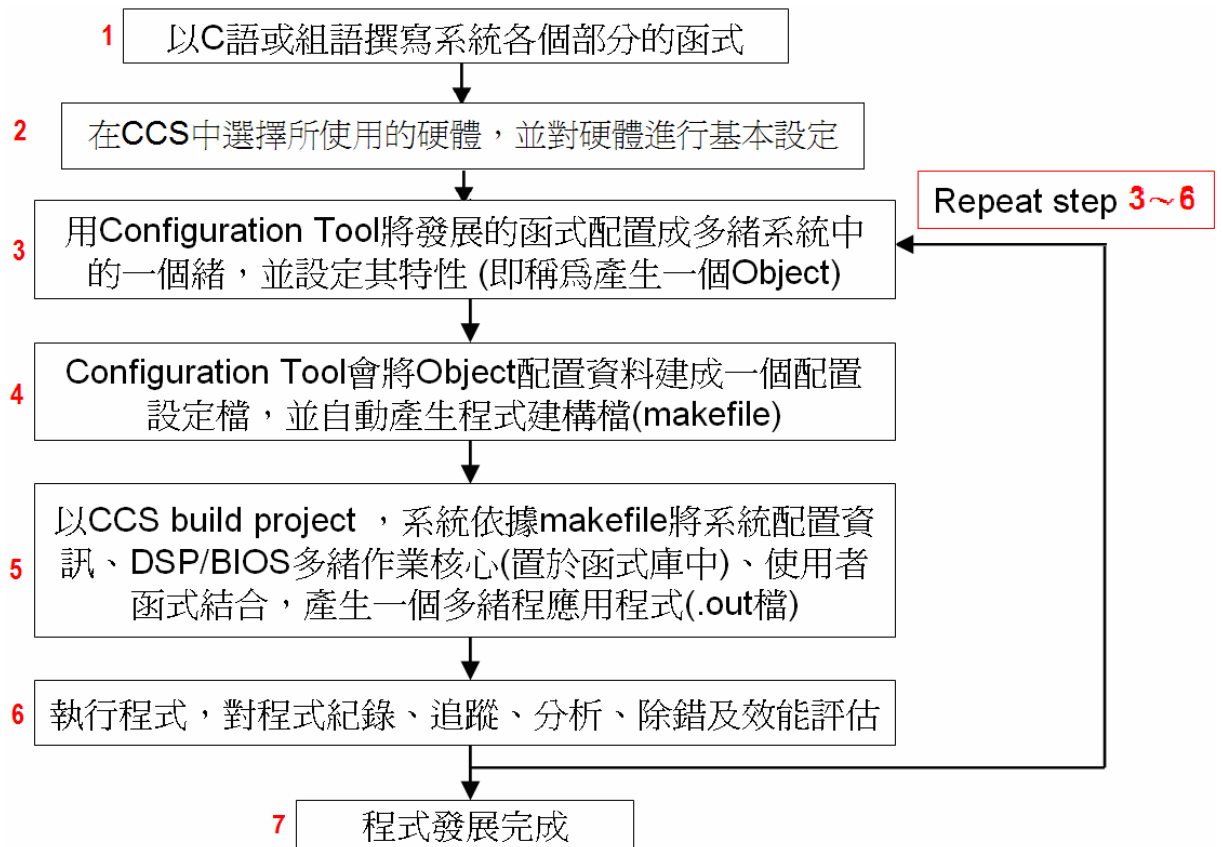


圖 3.11 CCS DSP/BIOS 程式發展流程

應用程式的發展，需先撰寫程式中各部分的功能函式，接著使用 Configuration Tool 配置即時系統中的執行緒(Thread)，如軟、硬體中斷、基本工作元、同步機制、資料傳輸佇列等等。並針對不同的元件分別設定其特性，如：優先權、函式呼叫、信號機和資

料佇列的名稱以及數量等等。設定完成後，Configuration Tool 會將此設定儲存並產生程式建構檔(makefile)。當 CCS 進行 building 時，會從 makefile 中取得系統設置資訊，並且從函式庫中取出所需要的 DSP/BIOS 作業核心，將前二者與使用者撰寫的 C 語言原始檔結合、建構成一個應用程式(.out 檔)。最後使用 CCS 的輔助工具進行測試，從中取得程式記錄判斷是否有誤，並對程式進行追蹤及分析，除錯完成後即可評估此應用程式的效能，進而完成程式的發展。

接下來將會針對幫助發展 DSP/BIOS 即時系統的 Code Composer Studio，對其操作介面以及其強大的輔助功能進行簡單的介紹。

3.4.1 DSP/BIOS Configuration Tool

在本文中所使用的 Code Composer Studio 版本為 3.1 版，其圖形化 DSP/BIOS 靜態設置介面如圖 3.12，我們可以直接在此介面靜態的設定系統中 DSP/BIOS 元件，將之分類如下：

- ◆ Hardware interrupts (HWI)
- ◆ Software interrupts (SWI)
- ◆ Tasks (TSK、IDL)
- ◆ Timing (CLK、PRD)
- ◆ Synchronization and Communication (SEM、MBX、QUE、LCK)
- ◆ Data and I/O Streams (RTDX、SIO、PIP、HST)
- ◆ Logging、Statistics、Tracing (LOG、STS、TRC)

在 CCS 的 DSP/BIOS Configuration Tool 介面之下可以直接對各個元件進行新增、移除、修改等動作，也可藉由此介面設定所使用的記憶體區間，還可以使用 LOG、STS 等元件來觀看程式執行時的事件記錄、或是程式執行的相關統計數據，如運算了幾次、運算時間多久、閒置時間多久等等。

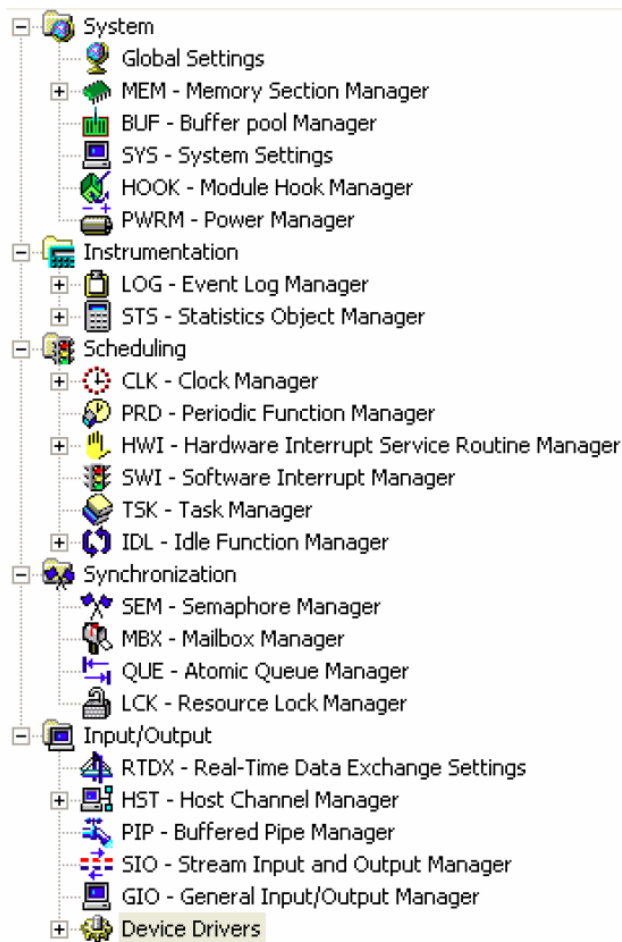


圖 3.12 DSP/BIOS Configuration Tool 介面

3.4.2 DSP/BIOS Real-Time Analysis Tool

DSP/BIOS 的特色之一，就是能在程式執行時對系統進行即時的觀察、分析，像先前使用的 $\mu\text{C}/\text{OS II}$ 就不支援此功能。在 DSP/BIOS 中，Real-Time Analysis (RTA) 提供系統相關資訊，因此必須進行計算、統計、記錄、傳輸等工作，DSP/BIOS 為了避免進行 RTA 而影響 CPU 工作效率，因此 RTA 是規劃在 IDL 之下執行，其功能包括了：

◆ CPU Load Graph :

RTA 會計算 CPU 的使用率，CPU 使用率為 $\text{CPU 工作時間} / (\text{CPU 工作時間} + \text{CPU 閒置時間})$ ，並繪出 CPU 使用率圖，如圖 3.13。

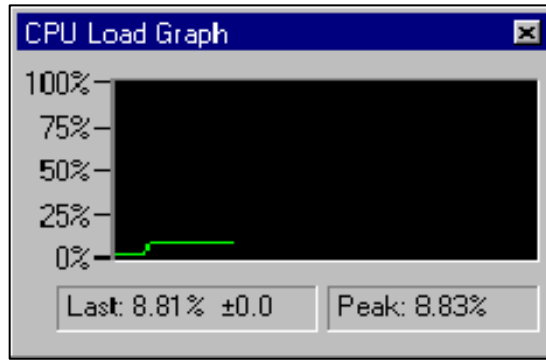


圖 3.13 CPU Load Graph

◆ Execution Graph :

執行時序圖為程式執行時，將每一個所要執行的工作(如 PRD、SWI、TSK、SEM、CLK 等)的執行過程圖形表現出來，我們可以清楚的由圖形知道工作的等待、搶佔、執行、結束等過程。且 execution graph 的時間軸是分別記錄 CLK 的 tick 以及系統事件觸發，也就是說時間軸並不是線性的，事件觸發越頻繁，在圖上一個 tick 的就會拉的越長，但一個 tick 的時間並不會改變，反之，如果沒什麼事件觸發，則在圖上一個 tick 將會縮短，如圖 3.14。

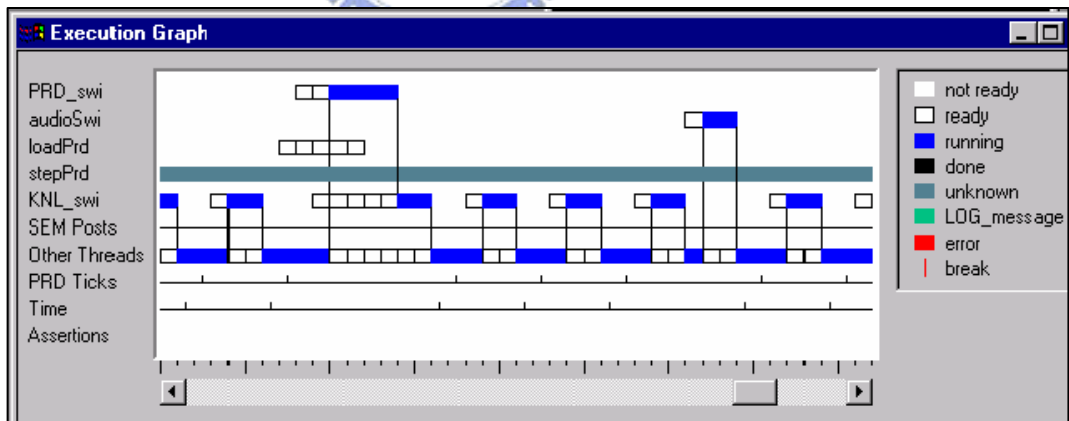


圖 3.14 Execution Graph

◆ Host Channel Control :

Host Channel Control 是用來管理 Host Channel Object 的工具，負責 Host 端和 Target 端資料的流動，分成 input channel 和 output channel，而 input 是從 Host 端讀取資料到 Target 端，output 是從 Target 端送資料到 Host 端。如圖 3.15 所示。

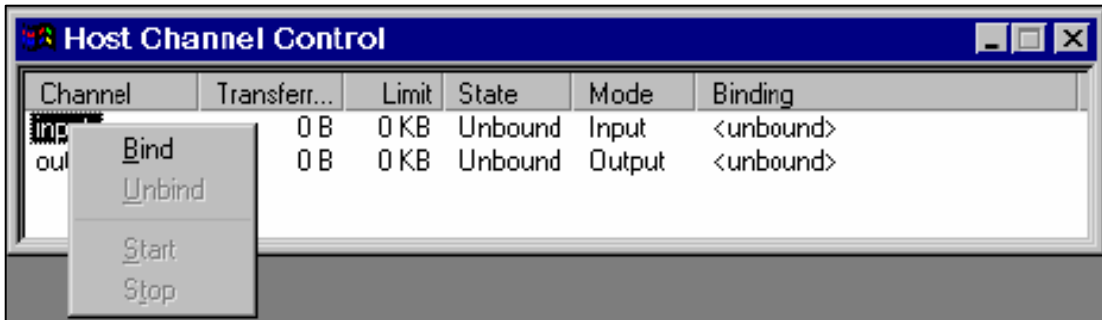


圖 3.15 Host Channel Control

◆ Message Log :

如圖 3.16 所示，透過此視窗可以獲得程式執行的相關記錄，我們可以由此來判斷程式是否正常的執行、結果是否正確，與 Execution Graph 的功能相同。

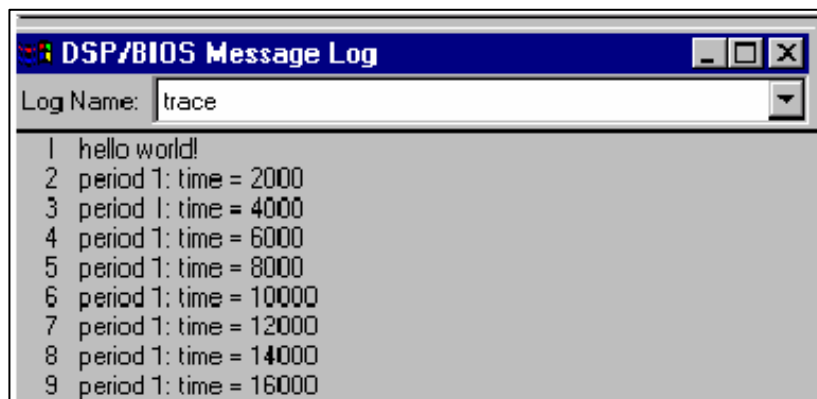
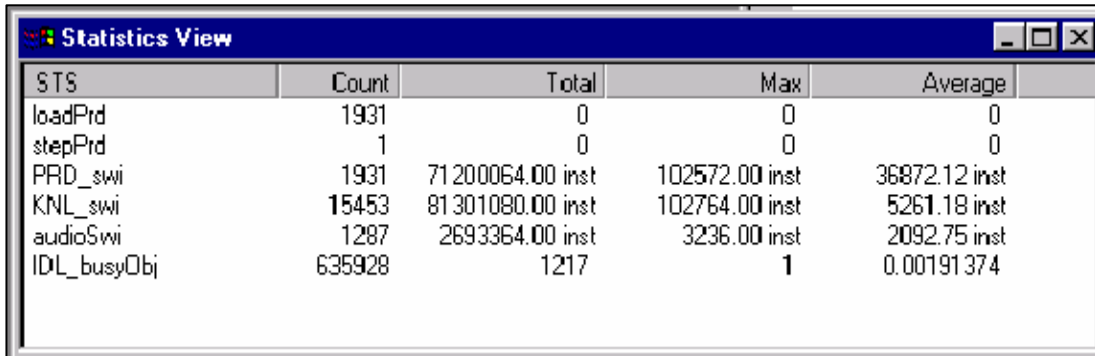


圖 3.16 Message Log

◆ Statistics View :

在 Statistics View 中，圖 3.17，主要有四項數據，Count、Total、Max、Average，其中 Count 為執行的次數、Total 為總運算次數、Max 為最大的運次數、Average 為 Total / Count 是平均運算次數。



STS	Count	Total	Max	Average
loadPrd	1931	0	0	0
stepPrd	1	0	0	0
PRD_swil	1931	71200064.00 inst	102572.00 inst	36872.12 inst
KNL_swil	15453	81301080.00 inst	102764.00 inst	5261.18 inst
audioSwil	1287	2693364.00 inst	3236.00 inst	2092.75 inst
IDL_busyObj	635928	1217	1	0.00191374

圖 3.17 Statistics View

◆ RTA Control Panel :

在此視窗中，如圖 3.18，我們可以依據所需要觀察的資訊，藉由 enable 以及 disable 這些選項來開啟或關閉對這些項目的追蹤，我們必須在『獲取資訊』以及『佔用的時間』中取捨，例如取消了幾個項目會讓我們減少許多資訊，但卻可以降低 RTA 佔用的時間。另外，我們也可以從中修改 RTA 顯示資料的更新速率，設定為 0 代表不會 polling 來更新畫面直到手動更新。

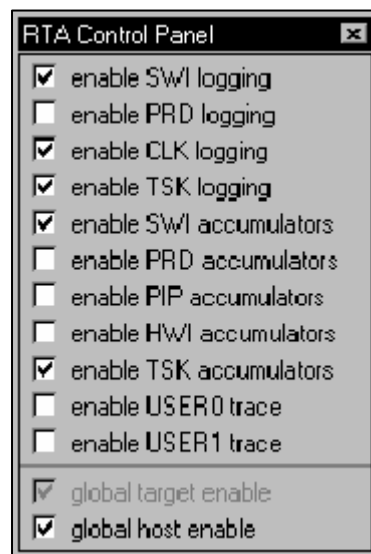


圖 3.18 RTA Control Panel

◆ Kernel/Object View :

Kernel/Object View 可以將現在正在使用的 DSP/BIOS 物件之設定、狀態由此視窗表現出來，畫面中可分成左方的 Tree View Area 表示所使用的物件、右方的 Property View Area 表示其參數和特性、以及下方的 Status Bar 表示執行的 thread 及狀態，如圖 3.19 所示。

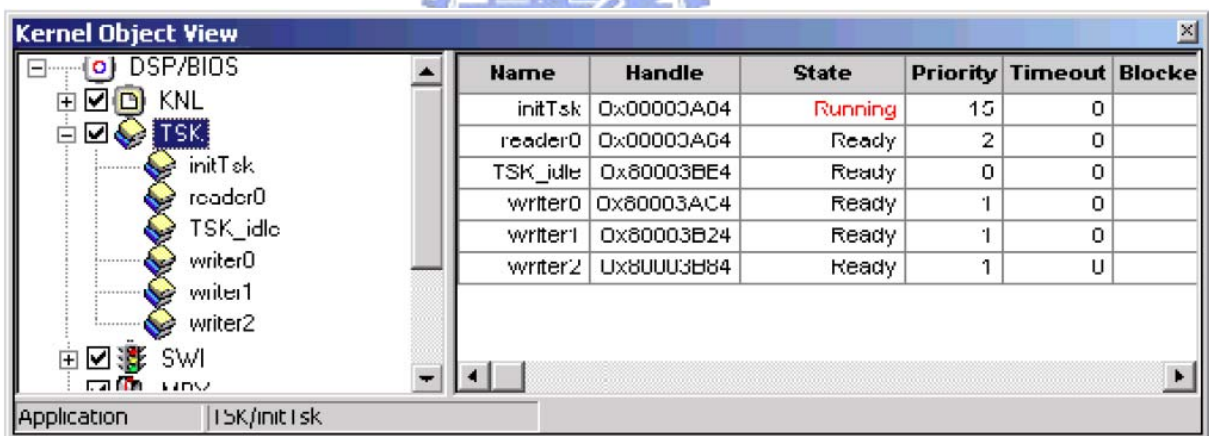
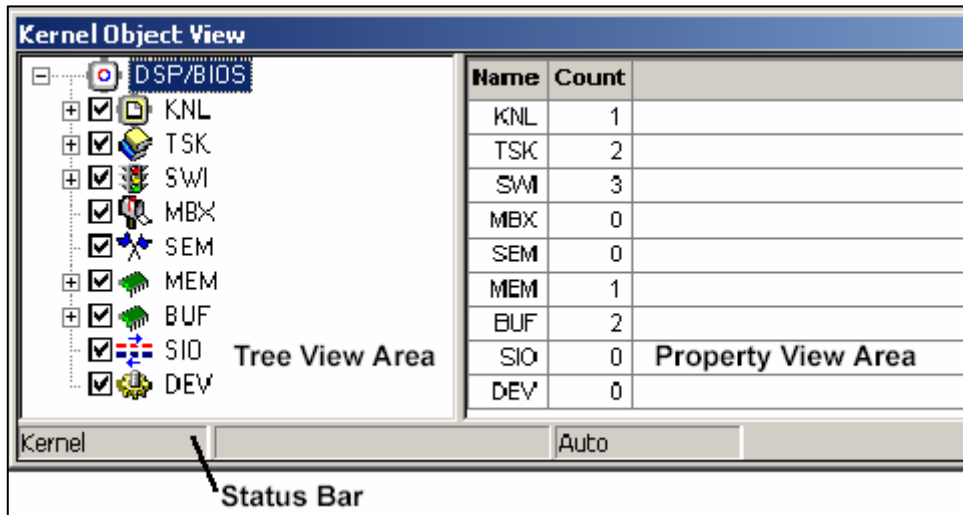


圖 3.19 Kernel/Object View

使用 Code Composer Studio 提供的這些即時分析工具，我們能夠快速的測試、評估程式的效果以及觀察 CPU 的使用量，若程式出現問題需要進行除錯時，透過這些工具，讓我們能夠快速的抓取程式各部分的資料進行比對，並且能夠即時觀察各個 Task 切換狀況、RTOS 中元件的使用狀況等。CCS 強大的功能讓使用者能夠以更短的時間設計、檢查、測試、評估其應用程式，並且成功的建構出一個完整的即時作業系統。

3.5 Task 規劃

在即時作業系統(RTOS)中，OS 負責工作的分配、切換以及記憶體的管理，而 Task 為即時作業系統中所有執行的工作，因此 Task 的規劃是很重要的一環，Task 規劃的完整與否將直接影響整個系統的性能。在[1]中提到實驗室先前所發展的運動控制卡規劃了八個 Task，每個 Task 各司其職、相輔相成，為了配合需求，在移植的過程中將之規劃成六個 Task，如圖 3.20 所示：

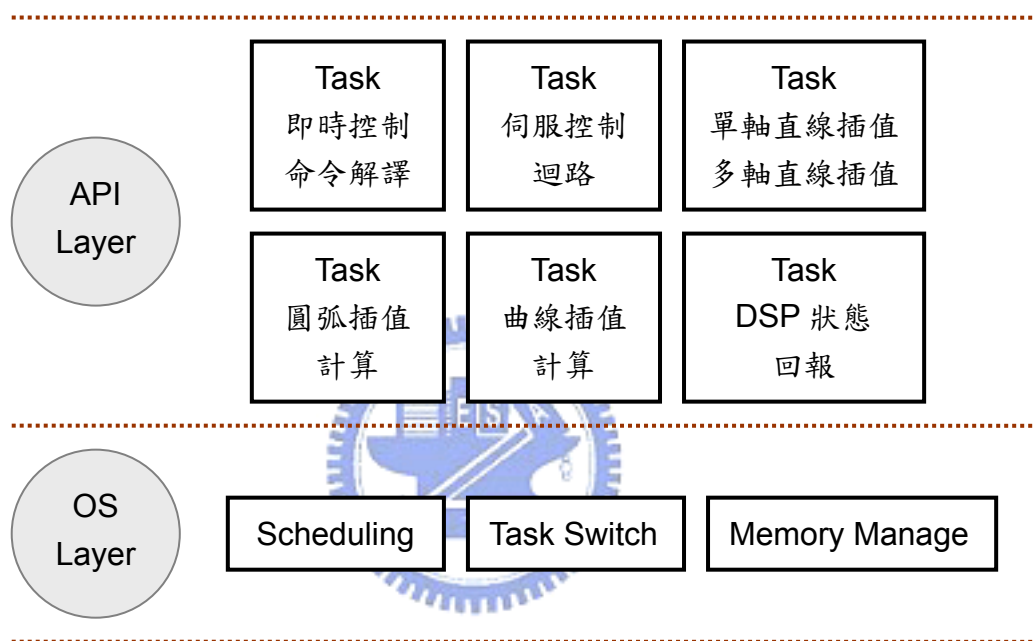


圖 3.20 Task 規劃圖

以下將對圖 3.20 中的六個 Task 做簡單的介紹：

(1) 即時控制命令解譯：

將 PC 端下達的即時命令(Real Time Command)進行解譯，並且擷取出命令的種類、條件參數、儲存命令資訊至記憶體中，最後呼叫對應的 Task 執行工作。

(2) 伺服控制迴路：

負責將插值 Task 中計算出來之運動命令(位置、速度命令)送入伺服控制迴路中，經過控制器補償之後，透過 D/A 轉換成對應的電壓輸出。

(3) 單軸、多軸直線插值：

接收使用者下達的單軸、多軸直線運動命令，視需求選擇 T-Curve 或 S-Curve

之加減速規劃，直接計算出位置、速度命令之細插值，並存入佇列(Queue)中供伺服迴路使用。

(4) 圓弧插值：

產生雙軸圓弧順時針與逆時針曲線的命令細插值，並儲存於佇列中以供伺服控制迴路使用。

(5) 曲線插值運算：

主要將粗插值經過三次曲線擬合(Cubic Spline Interpolation) 疊代出細插值並儲存於佇列(Queue)中以供伺服控制迴路存取

(6) DSP 狀態回報：

回傳給 Host 端 DSP 的執行狀態以及系統資訊如位置、速度.....等。

考慮 DSP/BIOS 的基本特性與系統效能上的平衡，在本文中將幾個必要的 Task (如即時命令解譯、伺服控制迴路)以靜態的方式產生，而其它計算插值命令的 Task 則由動態方式產生，節省記憶體的使用量，另外，根據 DSP/BIOS 的特色，我們可以將 DSP 狀態回傳的 Task 配置在 IDL Thread 中(如圖 3.5)以避免狀態回傳影響了 CPU 運算的效能。

接下來將會對即時控制命令(Real Time Command)所規劃的內容以及分類做說明介紹，還有將會對路徑規劃中的加減速規劃做進一步的說明。

3.5.1 即時控制命令(Real Time Command)規劃

在本中所規劃的即時控制命令可以分為幾種不同類型的命令群組，第一大類型是基本設定函式，下表將舉出比較重要的即時命令來說明：

類型	命令名稱	說明
Card Basic Setting Function	Reset_FPGA	對 DSP 下達重置指令
	Clear_Card_Buffer	清除 DPRam 中的指令
Axis Basic Setting Function	Set_DDA_Pluse_Mode	設定 DDA 輸出模式
	Set_MaxSpeed	設定軸的最大速度(電壓限制)
Axis Controller Setting Function	Set_Control	設定控制器的模式
	Set_P_PD	設定位置迴路 PD 控制器參數
	Set_V_PID	設定速度迴路 PID 控制器參數
Group Basic Setting Function	Set_Group	設定多軸群組
Auto Homing	Set_HomeSpeed	設定 Homing 速度
	Set_HomeMode	設定 Homing 方式及參數
	Home_Start	設定軸開始執行

表3.3 即時控制命令-基本設定

在表 3.3 中列出較重要的即時命令包括卡端基本設定函式、各軸基本設定、各軸控制器參數設定、設定多軸群組、以及 Homming 運動設定等等。透過 Reset_FPGA 和 Clear_Card_Buffer 我們可以重置 DSP 的狀態及資訊以及清除 DPRam 中所儲存的命令；而 Set_MaxSpeed 可設定機台最大速度來保護機台；Set_Control 等函式可提供使用者自行設定位置、速度迴路的控制器參數；Set_Group 可將不同的軸設定成同一群組，若有多軸需要進行相同的運動，則只需要一個群組指令即可；在 Homming 部分則可以由使用者選擇 Homming 的速度以及方式，在此 Homming 的方式是採用[2]中所提出來幾種方式進行 Homming。

即時控制命令的另一類的群組則是運動命令群組，如下表所示：

類型	命令名稱	說明
Axis Move Groups	Fix_Move	各軸固定位置對稱加減速運動
	Fix_Move_Para	固定位置對稱加減速運動參數
	As_Fix_Move	固定位置非對稱加減速運動
	As_Fix_Move_Para	固定位置非對稱加減速參數
Interpolation Groups	Line2_Move	兩軸直線對稱加減速運動
	Line3_Move	三軸直線對稱加減速運動
	Line_Move_Para	直線對稱加減速運動參數
	Line2_As_Move	兩軸直線非對稱加減速運動
	Line3_As_Move	三軸直線非對稱加減速運動
	Line_As_Move_Para	直線非對稱加減速運動參數
	Arc2_Move	兩軸圓弧加減速運動
	Arc2_Move_Para	兩軸圓弧加減速運動參數
	Motion_Start	補間運動開始
	Motion_End	補間運動停止

表3.4 即時控制命令-運動命令

在表 3.4 中可以知道在即時運動命令中主要分兩類，分別是 Axis Move Groups 以及 Interpolation Groups 兩類，在 Axis Move Groups 中所規劃命令都是各軸獨立的命令，並不會考慮軸與軸間相互的關係，而在 Interpolation Groups 中所規劃的命令則是多軸補間的運動命令。其中，Fix_Move 以及 Fix_Move_Para 是各軸獨立運動，給定終點位置、初始速度、最大速度、加減速度、Offset 速度後，會依據參數中所設定的加減速方式進行對稱的加減速規劃(等速、T-Curve 或 S-Curve)；另一方面，As_Fix_Move

以及 `As_Fix_Move_Para` 則是在給定相關參數後進行非對稱的加減速規劃；在 `Line2_Move`、`Lin3_Move`、`Line_Move_Para` 中規劃著兩軸或三軸直線補間運動，使用者給定終點位置、初始速度、最大速度、加減速度、`Offset` 速度參數後，進行投影拆解成 X、Y 軸後進行對稱的加減速規劃；同理 `Line2_As_Move`、`Line3_As_Move`、`Line_As_Move_Para` 則是進行非對稱的加減速規劃；對於 `Arc2_Move` 以及接下來的 `Arc2_Move_Para` 來說，則是進行圓弧補間的速度規劃，其內容及演算過程皆在[1]、[2]中有詳細的說明。

類型	命令名稱	說明
Other Function	<code>Drv_Hold</code>	暫停目前運動
	<code>Drv_Start</code>	開始目前運動
	<code>Set_SdStop</code>	設定系統減速停止
	<code>Set_EmgStop</code>	設定系統緊急停止

表3.5 即時控制命令-其他命令

在表 3.5 中列出幾個其他的功能函式，包括 `Drv_Hold` 的暫停以及 `Drv_Start` 的繼續開始，以及設定系統減速停止的 `Set_SdStop` 以及設定系統緊急停止的 `Set_EmgStop` 都是運動過程中可能需要用到的重要指令。

知道即時命令規劃的類型之後，接下來將會針對上面曾經提到的加減速規劃做簡單的說明，並且提出本文對加減速規劃進行的修改，期望產生更準確的運動命令以及更彈性的命令下達。

3.5.2 加減速規劃

在運動控制中，速度的規劃是很重要的，因為步階式的速度命令需要極大的扭矩，如此會造成馬達過大的力矩輸出，使得機台產生振動減少機台壽命。藉由加減速的規劃，可以有效的使馬達的輸出力矩能更加平滑、有效率，避免伺服系統產生上述的缺點。目前已有不同的加減速曲線規劃被廣泛的運用在 CNC 加工機的速度控制上，如直線梯形加減速規劃、正弦加減速規劃等，在實驗室所發展的即時運動控制系統中已經成功規劃出上述的加減速規劃方式[1]，在此小節中將會參考[1]與[12]中的加減速規劃方式配合需求做彈性化的修改。

3.5.2.1 梯形加減速規劃(T-Curve)

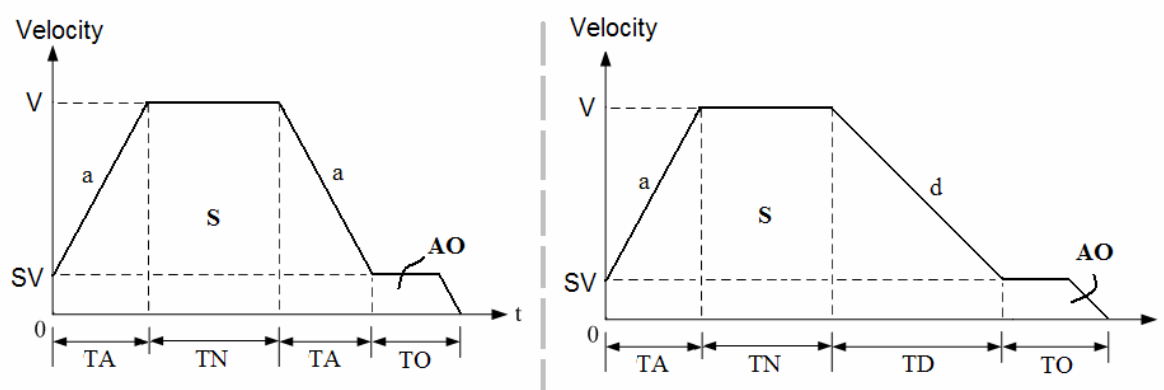


圖 3.21 對稱&非對稱 T-Curve 速度曲線

對稱與非對稱梯形速度曲線之差別在於加減速是否相同，若加減速相同則稱為對稱，反之則稱為非對稱加減速曲線。如圖 3.21 中表示， SV 為初始速度、 V 為最大速度、 a 為加速度、 d 為減速度、 TA 為加速時間、 TN 為等速段時間、 TD 為減速時間、 S 為命令位移(為到達命令位置所需的位移)、而 AO 表示停止前所預留的一段低速移動距離。在梯形加減速曲線的規劃中，加速、減速部分皆是線性加減速，在本文中將一般的 T-Curve 速度曲線做了改變，希望能有更好的效果以及更大的彈性。

考慮摩擦力的影響，因此提供使用者給定初始速度衝過靜摩擦部分，並且考慮停止

位置的準確度，會先預留一段距離以初始速度做低速移動至目標位置，在此規劃之下，使用者需要輸入終點位置、加速度、減速度、初始速度、最大速度、AO 位移，就可以計算出每一瞬間的位置速度命令，我們將以下面的步驟說明本文中 T-Curve 如何產生。

- 1.) 定義整段加減速為兩階段，第一階段為執行 AO 之前，第二階段為執行 AO 之後。
- 2.) 在第一階段中，加減速的時間可由最大速度 V 與加速度 a 、減速度 d 求得。

$$\begin{cases} TA = N_a T_s = \frac{V - SV}{a} \\ TD = N_d T_s = \frac{V - SV}{d} \\ TN = N T_s \end{cases} \quad (3.1)$$

其中 T_s 為即時系統之 sampling time(本文中為 0.0005 sec)， N_a 為加速所需要的取樣時間， N 為等速所需要取樣時間， N_d 為減速所需要的取樣時間。其中 N 的計算可由計算第一階段的移動距離以及(3.1)式得到：

$$\begin{aligned} S - AO &= SV(TA + TN + TD) + \frac{(V - SV)(2TN + TA + TD)}{2} \\ S - AO &= SV(N_a + N + N_d)T_s + \frac{(V - SV)(2N + N_a + N_d)T_s}{2} \\ N &= \frac{S - AO}{VT_s} - \left(\frac{SV + V}{V}\right)\left(\frac{N_a}{2} + \frac{N_d}{2}\right) \end{aligned} \quad (3.2)$$

(3.2)式中 V 是最大速度，但考慮到命令距離太短時無法加速到最大速度(無等速段)，此時必須對最大速度做修改，以維持加減速時間之正確，修改後的實際最大速度 V_m 如下：

$$V_m = \begin{cases} \frac{(S - AO) - SV\left(\frac{N_a}{2} + \frac{N_d}{2}\right)T_s}{\left(\frac{N_a}{2} + \frac{N_d}{2}\right)T_s} & N \leq 0 \\ \frac{(S - AO) - SV\left(\frac{N_a}{2} + \frac{N_d}{2}\right)T_s}{\left(N + \frac{N_a}{2} + \frac{N_d}{2}\right)T_s} = V & N > 0 \end{cases} \quad (3.3)$$

若 $N > 0$ 時，表示移動距離夠長，足夠加速到最大速度 V ，並維持在最大速度移動 NT_s 的時間。若 $N \leq 0$ 表示沒有等速段的速度規劃，所以代表無法加速到最大速度或者是一加速到最大速度就開始減速，因此參考[12]來調整實際最大速度 V_m 來規劃出無等速段的速度曲線。

- 3.) 在第二階段的 AO 中，我們根據使用者提出的 AO 距離、減速度 d 來計算減速停止的時間以及等速行進的時間，如下所示：

$$\begin{aligned}
 TO &= TO_N + TO_d \\
 TO_d &= \frac{SV}{d} = N_{od}T_s \\
 TO_N &= \frac{AO - SV\left(\frac{N_{od}}{2}\right)T_s}{SV(T_s)} = N_{ON}T_s
 \end{aligned} \tag{3.4}$$

其中 TO 為整個 AO 段的時間、 TO_N 為維持等速的時間、 TO_d 為減速到停止的時間、 N_{ON} 為等速段的取樣時間、 N_{od} 為減速段的取樣時間。

在此同樣考慮 AO 距離太短或是減速度 d 太小時，導致 AO 距離不夠減速，因此必須對此狀況作修正，與前面不同，在此將會以修正減速度 d 來達成目標，

$$d_m = \begin{cases} \frac{SV^2}{2AO} & N_{ON} \leq 0 \\ \frac{SV^2}{2(AO - N_{ON}T_sSV)} = d & N_{ON} > 0 \end{cases} \tag{3.5}$$

- 4.) 計算每個取樣時間下位置的增量 δP 為，分別考慮 N 以及 N_{ON} 等速段是否存在：

若 $N > 0$

$$\delta P(kT_s) = \begin{cases} \left(SV + \frac{(2k-1)(V_m - SV)}{2N_a} \right) T_s, & 1 \leq k \leq N_a \\ V_m T_s, & (N_a + 1) \leq k \leq (N_a + N) \\ \left(SV + \frac{(2(N_a + N + N_d - k) + 1)(V_m - SV)}{2N_d} \right) T_s, & (N_a + N + 1) \leq k \leq (N_a + N + N_d) \end{cases} \tag{3.6}$$

若 $N \leq 0$

$$\delta P(kT_s) = \begin{cases} \left(SV + \frac{(2k-1)(V_m - SV)}{2N_a} \right) T_s, & 1 \leq k \leq N_a \\ \left(SV + \frac{(2(N_a + N + N_d - k) + 1)(V_m - SV)}{2N_d} \right) T_s, & (N_a + 1) \leq k \leq (N_a + N_d) \end{cases} \quad (3.7)$$

若 $N_{ON} > 0$

$$\delta P(kT_s) = \begin{cases} (SV)T_s & N_{1st} \leq k \leq N_{1st} + N_{ON} \\ \frac{SV(N_{1st} + N_{2nd} - k + 0.5)T_s}{N_{Od}} & N_{1st} + N_{ON} + 1 \leq k \leq N_{1st} + N_{2nd} \end{cases} \quad (3.8)$$

若 $N_{ON} \leq 0$

$$\delta P(kT_s) = \frac{SV(N_{1st} + N_{2nd} - k + 0.5)T_s}{N_{Od}} \quad N_{1st} + 1 \leq k \leq N_{1st} + N_{2nd} \quad (3.9)$$

在(3.8)、(3.9)式中的 N_{1st} 表示第一階段的總時間，必須考慮 $N > 0$ 、 $N \leq 0$ 兩種狀況會有不同的 N_{1st} ，若 $N > 0$ 則 $N_{1st} = N_a + N + N_d$ ，若 $N \leq 0$ 則 $N_{1st} = N_a + N_d$ 。同理 N_{2nd} 則為第二階段所需要的總時間，若 $N_{ON} > 0$ 則 $N_{2nd} = N_{ON} + N_{Od}$ ，若 $N_{ON} \leq 0$ 則 $N_{2nd} = N_{Od}$ 。

根據上面方程式的加減速段的規劃，以及本文所進行的調整，我們可以在某一時間點之前就可以預先計算出該點瞬間的速度及位置，此速度曲線可有效的計算並準確的到達期望的位移量 S ，並且提供了更多的設定參數供使用者制訂出更彈性化的命令。

3.5.2.2 正弦加減速規劃(S-Curve)

在[12]中推導出任意加減速曲線，同上面的 T-Curve，在本文中將會修改推導出來的 S-Curve 加減速曲線，使其具有更優良的效果，如下圖所表示即為本文中所使用的 S-Curve 加減速曲線。

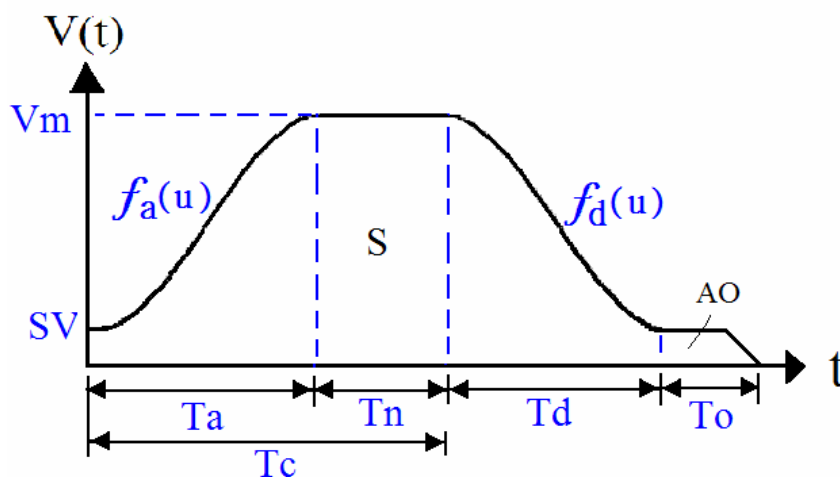


圖 3.22 對稱正弦速度曲線

圖 3.22 中， T_a 為加速段時間、 T_n 為等速段時間、 T_c 為開始減速的時間、 T_d 為減速段時間、 T_o 為停止前的低速位移時間、 SV 為初始速度、 V_m 為最大速度、 S 為命令位移、 AO 為停止前所預留的低速移動距離、 $f_a(u)$ 與 $f_d(u)$ 為修改後之[12]中所提出的加減速函式。

假設任意的加減速段為 $f_a(u)$ 與 $f_d(u)$ ，其在 $0 < u < 1$ 可微分，在 $0 \leq u \leq 1$ 為連續函數，速度 $V(t)$ 如下：

$$V(t) = \begin{cases} f_a\left(\frac{t}{T_a}\right), & 0 \leq t \leq T_a \\ V_m, & T_a \leq t \leq T_c \\ f_d\left(\frac{(t-T_c)}{T_d}\right), & T_c \leq t \leq T_c + T_d \end{cases} \quad (3.10)$$

其中修改後之速度曲線 $f_a(u)$ 與 $f_d(u)$ 為：

$$f_a(u) = SV + \frac{V_m - SV}{2} \left[\sin\left(\left(u - \frac{1}{2}\right)\pi\right) + 1 \right], \quad \text{其中 } u = \frac{t}{T_a} \quad (3.11)$$

$$f_d(u) = SV + \frac{V_m - SV}{2} \left[\sin\left(\left(u - \frac{3}{2}\right)\pi\right) + 1 \right], \quad \text{其中 } u = \frac{t - T_c}{T_d} \quad (3.12)$$

因此，在每個取樣時間下加速段位置增加量 δP_a 為：

$$\begin{aligned} \delta P_a(kT_s) &= \int_{(k-1)T_s}^{kT_s} f_a\left(\frac{t}{T_a}\right) dt \\ &= T_a \int_{(k-1)\frac{N_a}{N_s}}^{k\frac{N_a}{N_s}} f_a(u) du \\ &\equiv {}^a\gamma_k V_m T_s, \quad 1 \leq k \leq N_a \end{aligned} \quad (3.13)$$

減速段位置增加量 δP_d 為：

$$\begin{aligned} \delta P_d(kT_s) &= \int_{T_c + (k-1)T_s}^{T_c + kT_s} f_d\left(\frac{(t - T_c)}{T_d}\right) dt \\ &= T_d \int_{(k-1)\frac{N_d}{N_s}}^{k\frac{N_d}{N_s}} f_d(u) du \\ &\equiv {}^d\gamma_k V_m T_s, \quad 1 \leq k \leq N_d \end{aligned} \quad (3.14)$$

其中 ${}^a\gamma_k$ 、 ${}^d\gamma_k$ 定義為加減速段中每個取樣時間點所對應的位置增量參數。

定義整段加速段 $f_a(u)$ 的移動距離(面積) S_a 為：

$$\begin{aligned} S_a &= \int_0^{T_a} f_a\left(\frac{t}{T_a}\right) dt \\ &= T_a \int_0^1 f_a(u) du \\ &\equiv \alpha_a V_m T_a = \alpha_a V_m N_a T_s \end{aligned} \quad (3.15)$$

定義整段減速段 $f_d(u)$ 的移動距離 S_d 為：

$$\begin{aligned}
 S_d &= \int_{T_c}^{T_c+T_d} f_d \left(\frac{(t-T_c)}{T_d} \right) dt \\
 &= T_d \int_0^1 f_d(u) du \\
 &\equiv \alpha_d V_m T_d = \alpha_d V_m N_d T_s
 \end{aligned} \tag{3.16}$$

其中 N_a 、 N_d 為加速段及減速段的取樣時間數。

經由上面(3.10)~(3.16)式的定義，我們可以開始計算、產生 S-Curve 加減速曲線：

- 1.) 定義整段加減速為兩階段，第一階段為執行 AO 之前，第二階段為執行 AO 之後。
- 2.) 第一階段的加減速經由(3.11)~(3.14)計算每一取樣頻率所對應的係數 ${}^a\gamma_k$ 及 ${}^d\gamma_k$ ，積分如下所示：

$$\begin{aligned}
 &\left\{ \begin{aligned}
 {}^a\gamma_k &= \frac{T_a \int_{\frac{(k-1)T_s}{N_a}}^{\frac{kT_s}{N_a}} f_a(u) du}{V_m T_s} = \frac{T_a \int_{\frac{(k-1)T_s}{N_a}}^{\frac{kT_s}{N_a}} \left[SV + \frac{V_m - SV}{2} \left[\sin\left(\left(u - \frac{1}{2}\right)\pi\right) + 1 \right] \right] du}{V_m T_s} \\
 {}^d\gamma_k &= \frac{T_d \int_{\frac{(k-1)T_s}{N_d}}^{\frac{kT_s}{N_d}} f_d(u) du}{V_m T_s} = \frac{T_d \int_{\frac{(k-1)T_s}{N_d}}^{\frac{kT_s}{N_d}} \left[SV + \frac{V_m - SV}{2} \left[\sin\left(\left(u - \frac{3}{2}\right)\pi\right) + 1 \right] \right] du}{V_m T_s}
 \end{aligned} \right. \\
 &\Rightarrow \left\{ \begin{aligned}
 {}^a\gamma_k &= \frac{V_m + SV}{2V_m} + \frac{V_m - SV}{2\pi V_m} N_a \left[\cos\left(\frac{k-1}{N_a} - \frac{1}{2}\right)\pi - \cos\left(\frac{k}{N_a} - \frac{1}{2}\right)\pi \right] \\
 {}^d\gamma_k &= \frac{V_m + SV}{2V_m} + \frac{V_m - SV}{2\pi V_m} N_d \left[\cos\left(\frac{k-1}{N_d} - \frac{3}{2}\right)\pi - \cos\left(\frac{k}{N_d} - \frac{3}{2}\right)\pi \right]
 \end{aligned} \right. \tag{3.17}
 \end{aligned}$$

其中 $k \geq 1$

3.) 由(3.15)、(3.16)式計算整段加速段與減速段的參數 α_a 、 α_d ：

$$\begin{cases} \alpha_a N_a V_m T_s = T_a \int_0^1 f_a(u) du \\ \alpha_d N_d V_m T_s = T_d \int_0^1 f_d(u) du \end{cases}$$

$$\Rightarrow \begin{cases} \alpha_a = \frac{T_a \int_0^1 SV + \frac{V_m - SV}{2} \left[\sin\left(\left(u - \frac{1}{2}\right)\pi\right) + 1 \right] du}{N_a V_m T} \\ \alpha_d = \frac{T_d \int_0^1 SV + \frac{V_m - SV}{2} \left[\sin\left(\left(u - \frac{3}{2}\right)\pi\right) + 1 \right] du}{N_d V_m T} \end{cases}$$

$$\Rightarrow \begin{cases} \alpha_a = \frac{V_m + SV}{2V_m} \\ \alpha_d = \frac{V_m + SV}{2V_m} \end{cases} \quad (3.18)$$

4.) 考慮是否遇到需要修正最大速度 V_m 的情況，因此需要先找出是否有等速段，使用(3.15)、(3.16)式與命令位移 S 推導如下：

$$\begin{cases} S_a = \alpha_a V_m T_a = \alpha_a V_m N_a T_s \\ S_d = \alpha_d V_m T_d = \alpha_d V_m N_d T_s \\ S_N = V_m T_N = V_m N T_s \end{cases} \quad (3.19)$$

因此第一階段的移動距離為：

$$\begin{aligned} S - AO &= S_a + S_N + S_d \\ \Rightarrow S - AO &= S_a + S_N + S_d \\ \Rightarrow S_N &= S - S_a - S_d - AO \\ \Rightarrow N V_m T_s &= S - S_a - S_d - AO \\ \Rightarrow N &= \frac{S - S_a - S_d - AO}{V_m T} \end{aligned} \quad (3.20)$$

由(3.20)中得到等速段的取樣時間 N ，討論有無等速段的情況，並且適當的修正最大速度 V_m ：

$$V_m = \begin{cases} \frac{S - AO}{T_s(\alpha_a N_a + \alpha_d N_d)}, & N \leq 0 \\ \frac{S - AO}{T_s(N + \alpha_a N_a + \alpha_d N_d)}, & N > 0 \end{cases} \quad (3.21)$$

其中 α_a 、 α_d 為加速減速段整段的位移參數、 N_a 、 N_d 為加速段及減速段的取樣時間數、 AO 為預留位移量、 S 為命令位移量。

由(3.21) 式我們可以知道命令下達之後是否能夠加速到最大速度，如果不能夠加速到最大速度時，透過(3.21)式可以對最大速度進行修正，以確保加速時間、減速時間、移動距離的正確。



- 5.) 開始執行 AO 即進入第二階段的加減速規劃，參考梯形加減速規劃中的第二階段內容，根據使用者提出的 AO 距離、減速度 d 來計算減速停止的時間以及等速行進的時間，如下頁所示：

$$\begin{aligned} TO &= TO_N + TO_d \\ TO_d &= \frac{SV}{d} = N_{Od} T_s \\ TO_N &= \frac{AO - SV(\frac{N_{Od}}{2}) T_s}{SV(T_s)} = N_{ON} T_s \end{aligned} \quad (3.22)$$

其中 TO 為整個 AO 段的時間、 TO_N 為維持等速的時間、 TO_d 為減速到停止的時間、 N_{ON} 為等速段的取樣時間、 N_{Od} 為減速段的取樣時間。

在此同樣考慮 AO 距離太短或是減速度 d 太小時，導致 AO 距離不夠減速，因此必須對此狀況作修正，與前面不同，在此將會以修正減速度 d 來達成目標，

$$d_m = \begin{cases} \frac{SV^2}{2AO} & N_{ON} \leq 0 \\ \frac{SV^2}{2(AO - N_{ON}T_sSV)} = d & N_{ON} > 0 \end{cases} \quad (3.23)$$

6.) 計算每個取樣時間下位置的增量 δP 為，分別考慮 N 以及 N_{ON} 等速段是否存在：

若 $N > 0$

$$\delta P(kT_s) = \begin{cases} {}^a\gamma_k V_m T_s, & 1 \leq k \leq N_a \\ V_m T_s, & (N_a + 1) \leq k \leq (N_a + N) \\ {}^d\gamma_{(k-N_a-N)} V_m T_s, & (N_a + N + 1) \leq k \leq (N_a + N + N_d) \end{cases} \quad (3.24)$$

若 $N \leq 0$

$$\delta P(kT_s) = \begin{cases} {}^a\gamma_k V_m T_s, & 1 \leq k \leq N_a \\ {}^d\gamma_{(k-N_a)} V_m T_s, & (N_a + 1) \leq k \leq (N_a + N_d) \end{cases} \quad (3.25)$$

若 $N_{ON} > 0$

$$\delta P(kT_s) = \begin{cases} (SV)T_s & N_{1st} \leq k \leq N_{1st} + N_{ON} \\ \frac{SV(N_{1st} + N_{2nd} - k + 0.5)T_s}{N_{Od}} & N_{1st} + N_{ON} + 1 \leq k \leq N_{1st} + N_{2nd} \end{cases} \quad (3.26)$$

若 $N_{ON} \leq 0$

$$\delta P(kT_s) = \frac{SV(N_{1st} + N_{2nd} - k + 0.5)T_s}{N_{Od}} \quad N_{1st} + 1 \leq k \leq N_{1st} + N_{2nd} \quad (3.27)$$

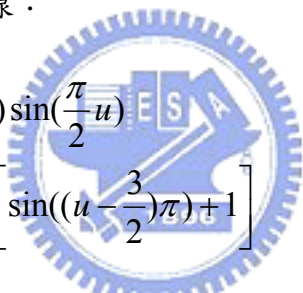
在(3.26)、(3.27)式中的 N_{1st} 表示第一階段的總時間，必須考慮 $N > 0$ 、 $N \leq 0$ 兩種狀況會有不同的 N_{1st} ，若 $N > 0$ 則 $N_{1st} = N_a + N + N_d$ ，若 $N \leq 0$ 則 $N_{1st} = N_a + N_d$ 。同理 N_{2nd} 則為第二階段所需要的總時間，若 $N_{ON} > 0$ 則 $N_{2nd} = N_{ON} + N_{Od}$ ，若 $N_{ON} \leq 0$ 則 $N_{2nd} = N_{Od}$ 。

在[12]中提到的非對稱 S-Curve 速度曲線規劃方式，與上一節所提到的非對稱速度曲線略有差別，在上一節的 T-Curve 速度曲線規劃中非對稱的定義為加速度、減速度不相同，但在[12]中提到的非對稱則是速度曲線定義不同，如下面所表示。

對稱 S-Curve 之加減速曲線：

$$\begin{cases} f_a(u) = SV + \frac{V_m - SV}{2} \left[\sin\left(\left(u - \frac{1}{2}\right)\pi\right) + 1 \right] \\ f_d(u) = SV + \frac{V_m - SV}{2} \left[\sin\left(\left(u - \frac{3}{2}\right)\pi\right) + 1 \right] \end{cases}$$

非對稱 S-Curve 之加減速曲線：

$$\begin{cases} f_a(u) = SV + (V_m - SV) \sin\left(\frac{\pi}{2}u\right) \\ f_d(u) = SV + \frac{V_m - SV}{2} \left[\sin\left(\left(u - \frac{3}{2}\right)\pi\right) + 1 \right] \end{cases} \quad (3.28)$$


綜合上述，若選擇非對稱的 S-Curve 加減速曲線，則以(3.28)式中的 $f_a(u)$ 與 $f_d(u)$ 重複進行(3.10)~(3.16)式的定義及運算，再繼續步驟 1.) 到步驟 6.) 的計算來產生 S-Curve 加減速命令。比較 S-Curve 與 T-Curve 加減速規劃可以知道，S-Curve 曲線比 T-Curve 更為連續，並不會有速度轉折點錯過取樣時間點的問題存在，啟動及停止也較 T-Curve 更順暢，但缺點為每次取樣時間的計算量較大，因此，選用 T-C 或 S-C 之間的取捨端看使用上的需求決定。

3.5.2.3 圓弧加減速規劃

在[2]中提到之圓弧插值器，如圖 3.23，假設半徑為 R 的圓弧，其切線速度為 V ，則 x 、 y 軸的速度須滿足下列方程式

$$\begin{cases} V_x = V \sin \theta \\ V_y = V \cos \theta \end{cases} \quad (3.29)$$

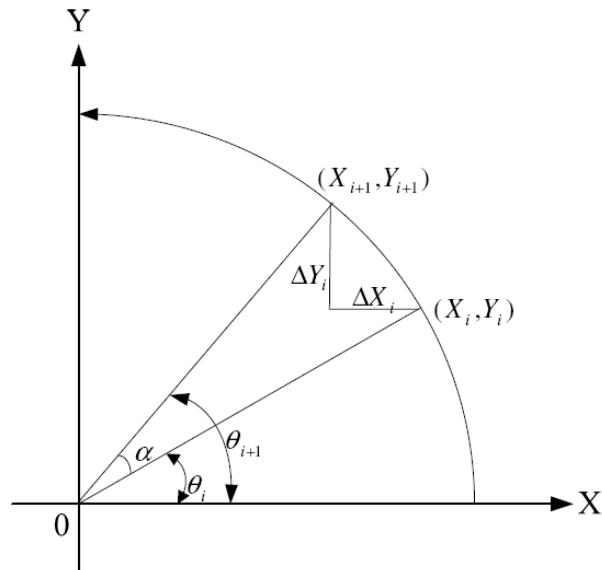


圖 3.23 圓弧曲線

其中 $\theta = \frac{V}{R}$ 。插值器在每一取樣時間內，會計算出命令，而這命令在取樣時間內是固定的，所以實際上圓弧運動是由片段直線所組成的。因此線段密度越高，所插值出的圓弧其精度也越高。如下圖所示，假設圓弧插值器演算每疊代一次， θ 就增加 α 角，求得差分方程式

$$\begin{aligned} \cos(\theta(i+1)) &= A \cos(\theta(i)) - B \sin(\theta(i)) \\ \sin(\theta(i+1)) &= A \sin(\theta(i)) + B \cos(\theta(i)) \end{aligned} \quad (3.30)$$

其中 $A = \cos \alpha$ ， $B = \sin \alpha$ ， $\theta(i+1) = \theta(i) + \alpha$ 。而線段對應的終點 $X(i+1)$ 、 $Y(i+1)$ 為

$$\begin{aligned} X(i+1) &= R \cos(\theta(i+1)) \\ Y(i+1) &= R \sin(\theta(i+1)) \end{aligned} \quad (3.31)$$

將式差分方程式代入可得

$$\begin{aligned} X(i+1) &= AX(i) - BY(i) \\ Y(i+1) &= AY(i) + BX(i) \end{aligned} \quad (3.32)$$

疊代出對應的插值點。

每一種不同的圓弧插值法，採用不同大小的 α 值即不同的 A 及 B ，一但此三個數值決定，則插值器取樣區間的線段長度方程式為

$$\begin{aligned} \Delta X(i) &= X(i+1) - X(i) = (A-1)X(i) - BY(i) \\ \Delta Y(i) &= Y(i+1) - Y(i) = (A-1)Y(i) + BX(i) \end{aligned} \quad (3.33)$$

其對應的速度命令為

$$\begin{aligned} V_x(i) &= V \frac{\Delta X(i)}{\Delta S(i)} \\ V_y(i) &= V \frac{\Delta Y(i)}{\Delta S(i)} \end{aligned} \quad (3.34)$$

其中 $\Delta S(i) = \sqrt{\Delta X^2(i) + \Delta Y^2(i)}$ 。由於 α 值很小，故 $\Delta S \cong R\alpha$ ，即線段長可用圓弧線段長代替，因此上式可簡化為

$$\begin{aligned} V_x(i) &= K\Delta X(i) \\ V_y(i) &= K\Delta Y(i) \end{aligned} \quad (3.35)$$

其中 $K = \frac{V}{R\alpha}$ ，如插值器取樣時間為 T ，則 $\alpha = \frac{V}{R}T$ ，由於 α 和 K 均為常數，因此在插值中只要計算一次即可。之後即可依照(4-18)與(4-20)公式疊代出圓弧的位置增量及速度。

上述所插值出的圓弧曲線為等速運動，在本文中將結合 4.3.1.1 節所規劃的梯型加減速曲線。給定起點終點的位置座標、中心點座標，可求得起點至終點弧線長度 R_a ，並可決定圓弧切線速度 V_{\max} 與加速度 A_{\max} 利用 4.3.1.1 節的方程式求得在每個取樣時間下的切線速度，予以求得圓弧的旋轉角度增量 $\alpha = \frac{V}{R}T$ 及 $K = \frac{V}{R\alpha}$ ，並分別應用(4-18) 與 (4-20)的方程式即可求得投影於兩軸的相對位置與速度曲線。

第四章 永磁馬達之鈍齒效應

在[3]~[6]中提到，永磁式馬達會受到鈍齒效應(Cogging)的干擾而降低響應精度，甚至有可能會產生噪音及震動。因此在馬達運動控制中，補償鈍齒力(Cogging Force)的影響是很重要的一環。在本章節中，我們將會探討鈍齒效應(Cogging)及鈍齒力(Cogging Force)的特性以及產生。

4.1 鈍齒效應之特性及現象

鈍齒效應所產生的現象，就是馬達在運作中(轉動或移動)，動子會受到一個週期性的干擾力，且此干擾力具有以下幾種特性：

- (1) 力的週期與大小只與動子轉動的角度或移動的距離有關。
- (2) 力的產生與電流無關，此力一直存在於具有鐵心的永磁系統中。

以永磁式旋轉馬達為例，即便在馬達斷電之下，伸手旋轉馬達的軸就能感覺到鈍齒力的存在，[13]、[14]中以下面簡單的示意圖來說明鈍齒效應的概念及特性。

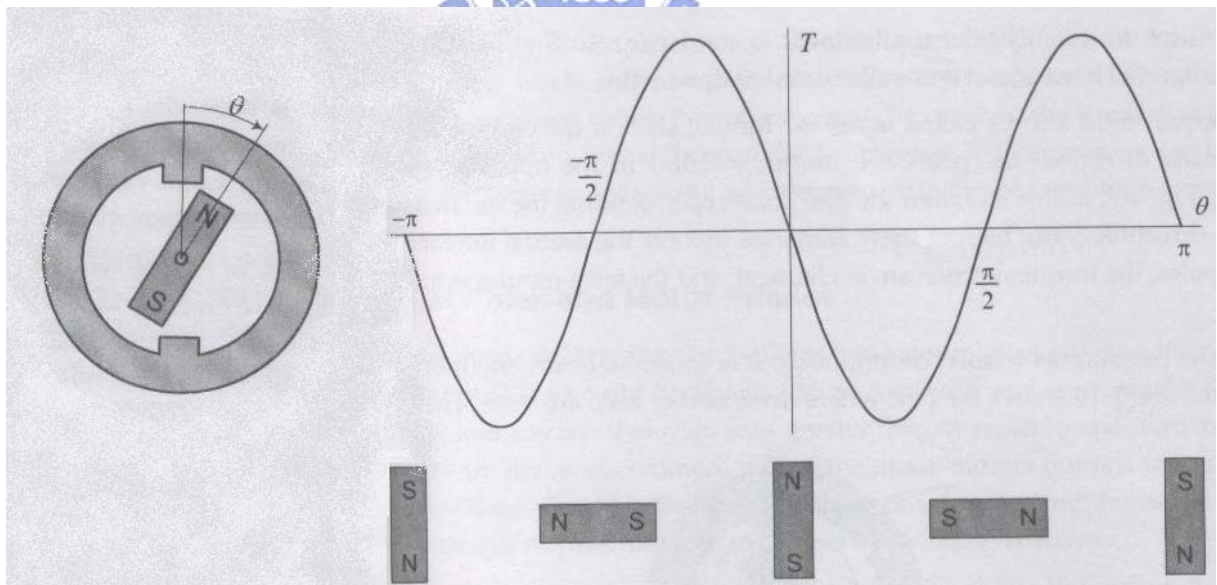


圖 4.1 Cogging Torque 示意圖

圖 4.1 中，一永久磁鐵中心固定，在兩極的鐵環內自由轉動，其轉動角度為 θ 順時

針為正，則永久磁鐵所受到的 Torque 與轉動角 θ 的關係如 4.1 圖右側所示，當轉子順時鐘轉動時，將會受到一週期性的作用力。在此定義 Torque 為 0 時永久磁鐵的位置稱為“detent position”，圖中所提到的 Torque 通稱“reluctance torque”，或稱為“cogging torque”。

鈍齒效應對於馬達控制系統產生的影響，透過圖 4.2 來作簡單的說明。圖 4.2 左圖中，經過繞線以及通電流，使鐵齒成為電磁鐵，對永久磁鐵產生吸力或斥力來驅動馬達，稱為“mutual torque”或“alignment torque”，若 T_m 驅使轉子順時鐘旋轉時，則轉子會另外受到週期性的鈍齒力影響，影響馬達控制系統的效能。鈍齒效應及鈍齒力對馬達系統產生的不良影響包括：

- (1) 使馬達的轉動(移動)不平滑
- (2) 產生震動或噪音
- (3) 降低馬達響應的精確度

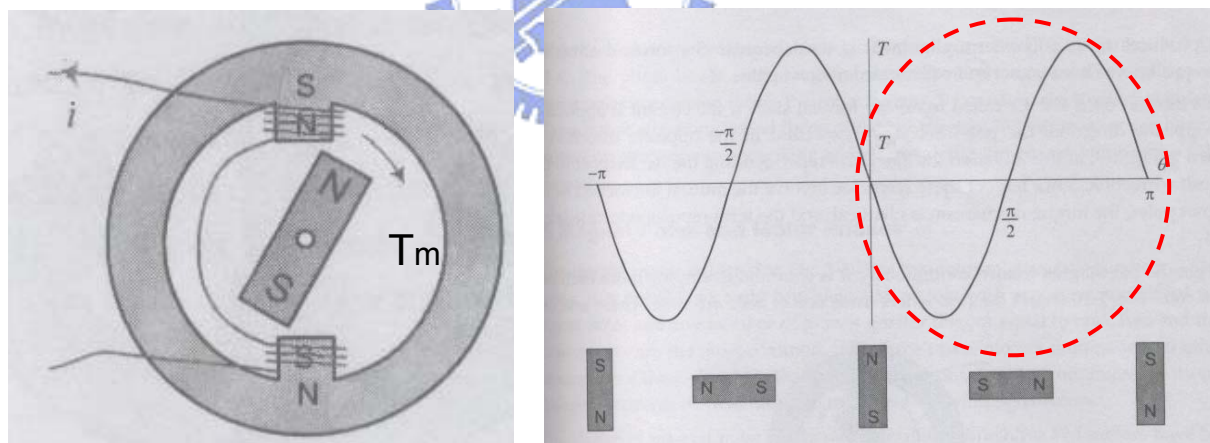


圖 4.2 鈍齒效應產生影響示意圖

知道鈍齒效應的特性及對系統造成影響，為了迴避鈍齒效應的影響，我們必須深一層瞭解鈍齒效應，因此，接下來我們將會討論鈍齒效應產生的原因。

4.2 鈍齒效應的產生

一般的永磁馬達系統中由電能產生磁能，再由磁能轉為機械能(感應力矩)進而驅動馬達轉子，但另一方面，永久磁鐵的磁能轉換而成的機械力即成為干擾馬達的鈍齒力。

參考[14]中對符號定義：

N : 線圈匝數	ϕ : 總磁通量(total flux)
λ : 磁通鏈(flux linkage)	e : 感應電壓(induced voltage)
L : 磁感(inductance)	p : 功率(power)
W : 能量 (energy)	W_c : 輔能(co-energy)
T : 感應力矩(torque)	

在[14]中提到，電流經過線圈以及永久磁鐵一起產生了總磁通(total flux)與磁通鏈(flux linkage)，如(4.1)所示：

$$\lambda = N\phi \quad (4.1)$$

經由 Faraday's law，我們可以由磁通鏈對時間微分計算出感應電壓(induced voltage)，如(4.2)所示：

$$e = \frac{d\lambda}{dt} \quad (4.2)$$

由感應電壓與電流可以計算出功率(power)，進而推導出磁能(energy)，並且將磁能對轉動角度偏微分可以得到感應力矩，如(4.3)、(4.4)、(4.5)所示：

$$p = ei \quad (4.3)$$

$$\begin{cases} W = \int_0^t p dt = \int_0^t i \frac{d\lambda}{dt} dt = \int_{\lambda(0)}^{\lambda(t)} i d\lambda \\ W_c = \int_{i(0)}^{i(t)} \lambda di \end{cases} \quad (4.4)$$

$$T = - \left. \frac{\partial W}{\partial \theta} \right|_{i=const} = \left. \frac{\partial W_c}{\partial \theta} \right|_{\lambda=const} \quad (4.5)$$

其中，(4.5)式的感應力矩 T 包括了馬達驅動力矩(mutual torque)與影響馬達響應的磁阻力(reluctance torque)，有關磁阻力的說明在[14]中有詳細的說明。

為了進一步討論影響馬達響應的磁阻力，並且找尋鈍齒力的發源，我們以一簡單的示意圖代表永磁馬達系統，進行磁路分析與感應力矩的概念推導。如圖 4.3 所示，圖左半邊以一纏繞線圈的 C 型鐵和一永久磁鐵簡單表示永磁馬達系統，而整個系統的磁路則如圖 4.3 右半所示。

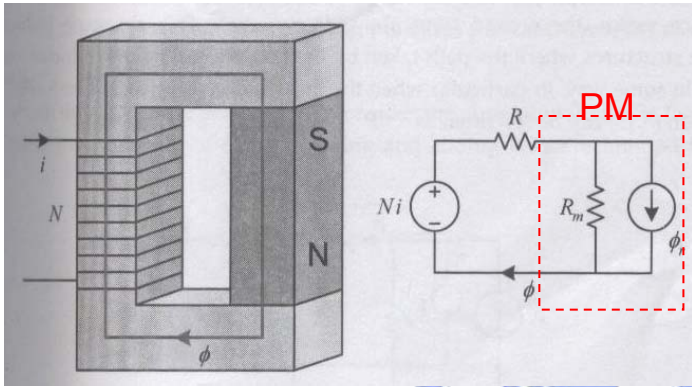


圖 4.3 永磁馬達系統示意圖

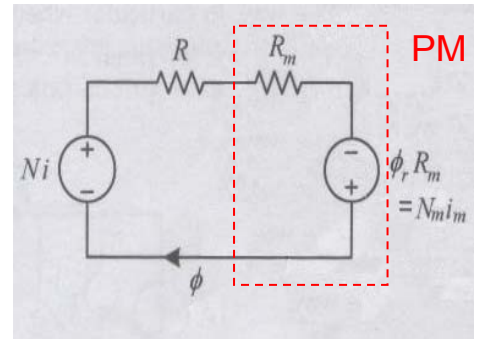


圖 4.4 The'vinin 等效磁路

在圖 4.3 中， N 為線圈匝數， i 為輸入電流， R 和 R_m 分別為 C 型鐵和永久磁鐵之磁阻， ϕ 及 ϕ_r 為磁路的總磁通與永久磁鐵的磁通，經由 The'vinin equivalent 可將圖 4.3 中的磁路等效成圖 4.4 中所示，我們計算 Co-Energy 來表示圖 4.4 中磁路的磁能：

$$\begin{aligned}
 W_c &= \frac{1}{2} L_1 i^2 + \frac{1}{2} L_m i_m^2 + L_{1m} i i_m \\
 &= \frac{1}{2} L_1 i^2 + \frac{1}{2} \left(\frac{N_m^2}{R + R_m} \right) \left(\frac{\phi_r^2 R_m^2}{N_m^2} \right) + \left(\frac{N N_m}{R + R_m} \right) i \left(\frac{R_m}{N_m} \phi_r \right) \\
 &= \frac{1}{2} L_1 i^2 + \frac{1}{2} \left(\frac{R_m^2}{R + R_m} \right) \phi_r^2 + Ni \left(\frac{R_m}{R + R_m} \right) \phi_r \\
 &= \frac{1}{2} L_1 i^2 + \frac{1}{2} (R + R_m) \phi_m^2 + Ni \phi_m
 \end{aligned} \tag{4.6}$$

其中， L_1 、 L_m 、 L_{1m} 分別表示鐵心的自感、永久磁鐵的自感以及鐵心和永久磁鐵之間的互感。另外， ϕ_m 表示由互感產生的磁通量，稱為 mutual flux，定義如下：

$$\phi_m = \frac{R_m}{R + R_m} \phi_r \quad (4.7)$$

因此，由(4.5)、(4.6)可以計算出系統的感應力矩如下：

$$\begin{aligned} T &= \frac{\partial W_c}{\partial \theta} \\ &= \frac{1}{2} i^2 \frac{dL_1}{d\theta} - \frac{1}{2} \phi_m^2 \frac{d(R + R_m)}{d\theta} + Ni \frac{d\phi_m}{d\theta} \end{aligned} \quad (4.8)$$

分析此力矩的成分，我們可以知道：

- 方程式(4.8)中第一項 $\frac{1}{2} i^2 \frac{dL_1}{d\theta}$ 為線圈的自感因素所產生的磁阻力矩(reluctance torque)。此力是由系統結構導致線圈自感隨著旋轉角度變化而改變所產生。
- 方程式(4.8)中第二項 $\frac{1}{2} \phi_m^2 \frac{d(R + R_m)}{d\theta}$ 為永久磁鐵自感因素所產生的磁阻力矩。基本上與第一項磁阻力相似，但此處由永及磁鐵磁通經過的磁阻改變而產生。
- 而(4.8)中的第三項 $Ni \frac{d\phi_m}{d\theta}$ 為線圈與永久磁鐵之間互感產生的相互力矩(mutual torque)，mutual torque 又稱為 alignment torque，是馬達系統中主要的驅動力矩。

承上所說，相互力矩(mutual torque)是馬達系統中的驅動力矩，提供力矩驅使馬達轉子轉動。另一方面，磁阻力則是隨著自感以及磁阻的改變而產生，會影響到馬達的響應，因此對馬達系統而言是不希望發生的非預期力(undesired torque)。

當系統在無電流通過($i=0$)的情況下，由線圈自感所引發的磁阻力便不會產生，如(4.8)中第一項，但是，方程式(4.8)中將會剩下第二項的磁阻力矩，此磁阻力由永久磁鐵產生的磁通經過會改變的磁阻產生，因而與電流無關，此一力矩為磁阻力矩中的特例，所以

我們又稱此種與電流無關的力矩為鈍齒力矩(cogging torque)，如(4.9)所表示，而此種產生鈍齒力的效應為鈍齒效應。

$$T_{cog} = -\frac{1}{2}\phi_m^2 \frac{d(R+R_m)}{d\theta} \quad (4.9)$$

在前面我們以簡單模型討論鈍齒力的產生，並且從概念上推導鈍齒力的方程式，若進一步考慮馬達系統中存在的空氣隙(air gap)對鈍齒力推導的影響，我們將空氣隙加入圖 4.3 的模型之中，如下圖 4.5 所示：

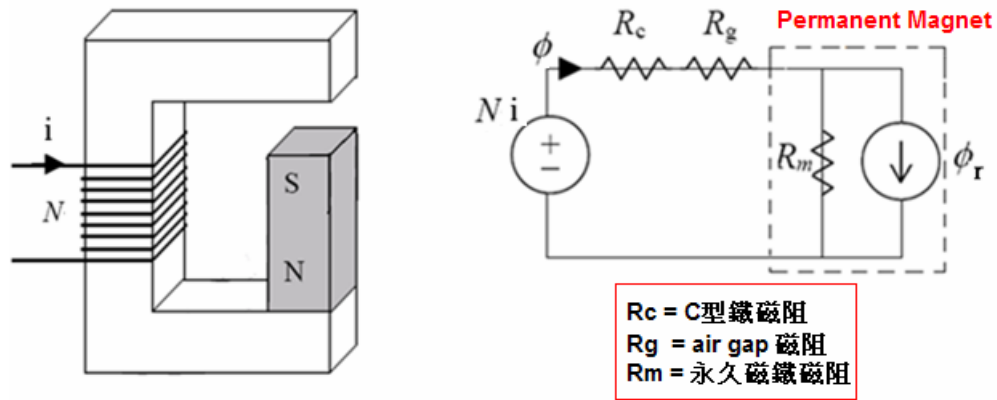


圖 4.5 加入 air gap 之模型及磁路示意圖

圖 3.5 左半為加入空氣隙後的模型，以此簡單表示永磁馬達系統，而磁路如圖右半邊所表示，其中 N 為線圈匝數、 i 為輸入電流、 ϕ 是總磁通、 R_c 為 C 型鐵磁阻、 R_g 為 air gap 的磁阻、 R_m 為永久磁鐵的磁阻、 ϕ_r 為永久磁鐵產生的磁通。經過推導，圖 4.5 中加入空氣隙後系統的 co-energy 如下：

$$W_c = \frac{1}{2}L_1 i^2 + \frac{1}{2}(R_c + R_g + R_m)\phi_m^2 + Ni\phi_m \quad (4.10)$$

其中 L_1 為線圈的自感， ϕ_m 為 mutual flux，表示如下：

$$\phi_m = \frac{R_m}{R_c + R_g + R_m} \phi_r \quad (4.11)$$

因此，考慮空氣隙之後，系統所產生的感應力矩 T 可由(4.10)式 co-energy 對轉動角偏微分求得，如(4.12)所示，並且由前面的討論可知，此時系統的鈍齒力 T_{cog} 如(4.13)所示：

$$\begin{aligned} T &= \frac{\partial W_c}{\partial \theta} \\ &= \frac{1}{2} i^2 \frac{dL_1}{d\theta} - \frac{1}{2} \phi_m^2 \frac{d(R_c + R_g + R_m)}{d\theta} + Ni \frac{d\phi_m}{d\theta} \end{aligned} \quad (4.12)$$

$$T_{cog} = -\frac{1}{2} \phi_m^2 \frac{d(R_c + R_g + R_m)}{d\theta} \quad (4.13)$$

並且由前面的討論可知，(4.13)式即是考慮了空氣隙之後系統所產生的鈍齒力矩，其中空氣隙之磁阻改變將會直接影響到鈍齒力矩之產生。回顧圖 4.2，從中我們可以知道隨著轉子的轉動，轉子與鐵心之間空氣隙的大小隨著轉子轉動而不同，所以空氣隙之磁阻也會隨著轉動角度而改變，因此，鈍齒力將會隨著空氣隙磁阻的改變而產生。

因此，在[14]中提到幾種空氣隙模型(air gap model)用以計算空氣隙之磁阻的概念，在此不加以說明空氣隙模型以及空氣磁阻的計算方式。我們從[14]中可歸納出幾個結論：

1. 在馬達控制中，所有的 Reluctance Torque 皆為 undesirable。
2. Cogging Torque 為 Reluctance Torque 的一種特例。
3. Cogging Torque 可經由 FEM 方式計算，也可由磁能微分計算。
4. 磁能的計算方式：

a. 計算磁通鏈，再推導磁能

$$W_c = \frac{1}{2} Li^2$$

b. 單位體積磁能的積分

$$W_c = \int_V \frac{\mu H^2}{2} dV = \int_V \frac{B^2}{2\mu} dV$$

5. 因此計算 Cogging Torque 必須先找出 $L(\theta)$ 、 $R(\theta)$ 、 $B(\theta)$ 、 $H(\theta)$ 。

由上可知，計算 Cogging Torque 的方式很多，如果不採用微觀之 FEM 而採用巨觀之磁能微分，最重要的是要能夠準確的找出每個所需要的參數，如 $L(\theta)$ 、 $R(\theta)$ ……等等，若找出的參數越準確，則計算出來的鈍齒力模型也就越真實。在看過鈍齒力之產生與推導之後，接下來我們將討論如何改善鈍齒效應。

4.3 鈍齒效應改善方法

在永磁式馬達的運動系統中，鈍齒力一直都被視為很重要的輸入干擾(input disturbance)，本文前面也提到，許多學者研究如何改善馬達的鈍齒效應，並提出了許多不同的方法，這些方法可以區分為兩大類：

- 1.) 結構上的改良
- 2.) 控制法則補償

以下將對這兩類的方法做簡單的說明以及介紹。



4.3.1 結構上的改良

前面我們提到了鈍齒效應產生的原因、過程、結果，簡單的說，鈍齒力是由永久磁鐵與鐵心之間的吸引力造成，若更進一步說明，鈍齒力是由於永久磁鐵的磁通經過變動的磁阻，造成永久磁鐵自感的變化進而產生的感應力矩。由(4.13)式我們可以知道當空氣隙磁阻改變即會產生鈍齒力，回顧圖 4.1，我們可以知道當轉子正對著繞線鐵齒時，此時的空氣隙最小，因此空氣隙磁阻最小。當轉子轉動時，因為空氣隙磁阻隨轉動角度而增加，所以產生出鈍齒力欲將轉子拉回 detent position(感應力矩合力為零的位置)。

由上面敘述我們可以知道，空氣隙的分佈不均勻乃是造成永磁馬達鈍齒效應的關鍵。所以從結構上來改善鈍齒效應是治本的方法，不少的學者專家研究出許多的方式來改善鈍齒效應，整理之後將之粗分為兩個類型：

(1) 改變齒槽外型：

此部分是藉著改變鐵齒、槽的外型來減少空氣隙磁阻的改變，如[5]中改變鐵齒的外

型成 T 型或扇形、在齒面增加輔助槽、在槽中增加輔助齒，以減少空氣隙之改變。
或是採用[15]中提到的斜槽方式來平均空氣隙，降低鈍齒力的大小。

(2)改變齒槽配置：

除了改變齒、槽外型之外，另一類型的方法包括：改變內部永久磁鐵的對稱性，來避開鐵齒與永久磁鐵的 alignment，降低鈍齒力的強度[4]；或是改變齒槽比例關係 [3]、[5]來降低鈍齒力簡諧項的大小。藉由此種方式讓鈍齒力相互抵銷已達到改善鈍齒效應的效果。

上面說的這兩種方法，都是經由專門的設計以及製造來達成改善鈍齒效應的效果，整組馬達幾乎需要重新製作，且由於上面提到的方法在製造過程中有相當的難度，因此提高了不少成本，雖然是從根本上來抑制鈍齒效應的產生，但對於馬達控制系統來說，高成本也將成為系統的負擔。所以不少學者從控制的觀點著手，若能以控制的方式補償馬達鈍齒效應，則可以大大提升馬達控制系統的經濟價值。



4.3.2 控制法則補償

因為從結構上去改善鈍齒效應的成本較高，所以藉由控制法則去補償鈍齒效應便成為另一種主流。其中包括建立系統的鈍齒力模型(cogging force model)再透過迴授控制來補償鈍齒力、或使用 robust control [6]、adaptive control [7]線上即時修改控制器參數來補償，或是經由 DOB 架構來估測干擾及雜訊，並且透過迴授控制補償干擾力。不論何種控制補償方式，經由控制法則補償系統鈍齒效應優點是成本較低，因為不必重新製造整組馬達，並且可以套用在任何馬達系統之中，改善馬達的鈍齒效應。

但系統控制的效果與控制架構的干擾抑制能力和 sensor 量測的準確度有絕對的關係。如果控制架構對於干擾的壓抑效果有限或是因為 sensor 量測的資料不準確，則迴授控制的效果將大打折扣，因此在本文中將會實現干擾觀測器的架構來抵抗鈍齒效應，並且在稍後的系統控制架構中將會介紹 DOB 的架構。

第五章 系統控制架構

5.1 DOB(Disturbance Observer)架構

在控制觀點中，主要都是在討論兩個重點，一是系統內部參數的不確定性；再者就是外界給予的干擾、雜訊。許多人研究如何降低系統不確定性(model uncertainties)，也有許多人致力於排除外部的干擾(disturbance)。在馬達運動控制中，馬達的運轉會產生摩擦力、力矩波漣(torque ripple)、鈍齒力、系統參數不確定性以及雜訊，因此馬達實際系統與理想系統有一定的差異。在前面提到我們可以找出鈍齒力模型來補償系統受到鈍齒力的影響，同理我們可以找出摩擦力模型來補償摩擦力的影響，以此類推。但在此節中將介紹另一種可以補償任何干擾的控制架構，而且不需要建干擾模型。

干擾觀測器(DOB, disturbance observer), 顧名思義, 能夠觀察到系統的 equivalent disturbance, 包括摩擦力、鈍齒力、磁阻力、不確定性等等。因此內迴路透過 DOB 估測出系統干擾, 再迴授修正控制輸入, 以消除系統的干擾, 並使系統逼近其 nominal model, 如此一來, 外迴路的控制器設計就比較容易。從數學上看, DOB 在抑制雜訊、排除干擾及降低系統不確定性的方面有著優異的性能。下圖 5.1 為 Umeno 和 Hori 在[8]中所提出的 DOB 架構:

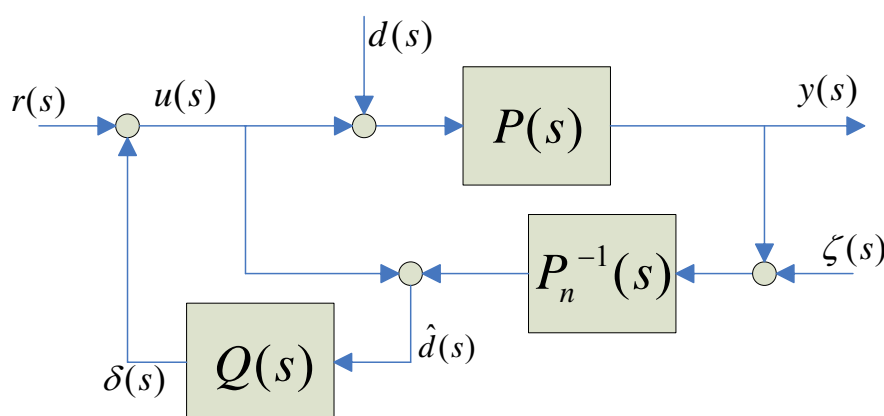


圖 5.1 DOB 方塊圖

其中

$r(s)$: Reference command

$P(s)$: Actual plant

$u(s)$: Control input

$P_n(s)$: Nominal plant

$y(s)$: System output

$Q(s)$: Low-pass filter

$d(s)$: Input disturbance

$\hat{d}(s)$: Estimated disturbance

$\zeta(s)$: Measurement noise

$\delta(s)$: Compensation force

由圖我們可以推導出下列關係式：

$$y(s) = G_{ry}(s)r + G_{dy}(s)d + G_{\zeta y}(s)\zeta \quad (5.1)$$

其中

$$G_{ry}(s) = \frac{P_n(s)P(s)}{P_n(s) + (P(s) - P_n(s))Q(s)}$$

$$G_{dy}(s) = \frac{P_n(s)P(s)(1-Q(s))}{P_n(s) + (P(s) - P_n(s))Q(s)}$$

$$G_{\zeta y}(s) = \frac{-P(s)Q(s)}{P_n(s) + (P(s) - P_n(s))Q(s)}$$

從 I/O 觀點來看，我們希望命令輸入到系統輸出($r \rightarrow y$)的轉移函數 $G_{ry}(s)$ 趨近於期望的系統(nominal plant，即 $P_n(s)$)；從抑制干擾和雜訊的觀點來看，我們希望干擾到輸出($d \rightarrow y$)以及雜訊到輸出($\zeta \rightarrow y$)之轉移函數 $G_{dy}(s)$ 、 $G_{\zeta y}(s)$ 能趨近於零。

若 $Q(s) \approx 1$ ：

$$\text{則 } G_{ry}(s) \approx P_n(s)$$

$$\text{且 } G_{dy}(s) \approx 0$$

若 $Q(s) \approx 0$ ：

$$\text{則 } G_{\zeta y}(s) \approx 0$$

由上可知，我們希望 $Q(s) \approx 1$ 來抑制干擾及消除系統不確定性，但又希望 $Q(s) \approx 0$ 來阻擋雜訊。由於一般機械系統及干擾都是低頻而雜訊多為高頻的影響，因此 Q 設計為一 low-pass filter 來滿足低頻時 $Q(s) \approx 1$ 且高頻時 $Q(s) \approx 0$ 。

承接前面討論， $P_n(s)^{-1}$ 為 nominal inverse plant 通常為 non-proper 函式，在圖 5.1 中是無法被實現的，因此在實現上必須採用等效架構來實現 DOB，圖 5.1 可等效如下圖 5.2：

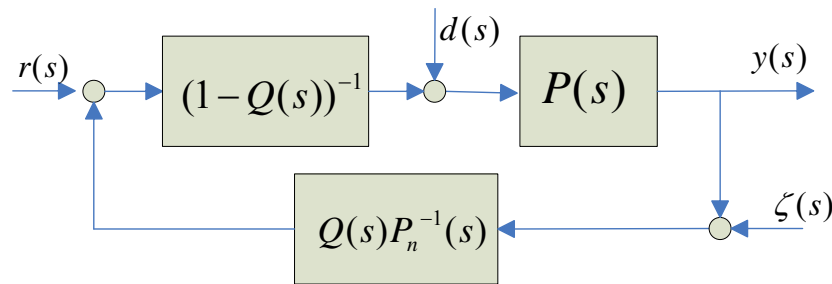


圖 5.2 DOB 等效方塊圖

如此一來便可透過設計 $Q(s)$ ，使得 $Q(s)P_n(s)^{-1}$ 形成 proper 函式，因此在 $Q(s)$ 的設計上又多了一項限制，即 $Q(s)$ 階數至少必須要跟 $P_n(s)$ 一樣(形成 proper 函式)或更高(形成 strict proper 函式)。最後，考慮低通濾波器 $Q(s)$ 及其頻寬的設計：

◆ $Q(s)$ 的設計：

+

在 $Q(s)$ 的設計上，Umeno 及 Hori 在[8]中提出以 Butterworth 形式來設計 $Q(s)$ ，並列出三個基本型式來滿足 $Q(s)$ 一階到三階的設計。之後，Umeno 和 Hori 在[9]提出 N 階 $Q(s)$ 的可以設計為：

$$Q(s) = \frac{1 + \sum_{k=1}^{N-m} a_k (s\tau_N)^k}{1 + \sum_{k=1}^N a_k (s\tau_N)^k} \quad (5.2)$$

其中 N 為 $Q(s)$ 的階數， m 為 $Q(s)$ 與 $P_n(s)$ 的相對階數， τ_N 為 cutoff frequency，而在本文中 $Q(s)$ 將設計為一簡單的二階低通濾波器，如下所示：

$$Q(s) = \frac{1}{(1 + \tau s)^2} \quad (5.3)$$

其中， τ 為 cutoff frequency，根據系統頻寬以及抗干擾需求來選擇適合的 τ 值。

◆ 頻寬的設計：

前面提到 $Q(s)$ 的頻寬至少必須比系統頻寬大，以確保抵抗系統不確定性； $Q(s)$ 頻寬必須大於干擾頻寬，才能透過迴授補償干擾； $Q(s)$ 頻寬必須小於雜訊頻寬，避免雜訊迴授來抑制雜訊。因此 $Q(s)$ 的頻寬設計也就是在『抑制干擾 & 不確定性』以及『抑制雜訊』之間做取捨。

上面提到 DOB 架構的優點，包括排除低頻干擾、高頻雜訊以及保持系統穩健性，在本文中將鈍齒力視為系統輸入干擾(input disturbance, DI)，並以 DOB 的架構來補償鈍齒力的影響。但是傳統 DOB 也有其缺點，整理如下所示：

1. 無法排除高頻干擾
2. 排除不連續的干擾的能力有限
3. 無法應付不確定性太大的系統
4. 無法應付非極小相位系統
5. 不一定能適用在 MIMO 的系統中

若從數學上能上來看，傳統 DOB 無法排除比系統頻寬還高的高頻干擾，在前面推導時也曾提到過若干擾頻寬太高，則 Q 的頻寬必須提高，如此就會有高頻雜訊進入，因此若干擾中包含了大量的高頻部分，則傳統 DOB 對此干擾的排除能力就會下降。除此之外，在[7]中曾提到，若以傳統 DOB 的架構搭配上乘法系統不確定性來進行 small gain theorem 的推導，發現若要滿足 small gain theorem 的話，系統的不確定性不能超過 nominal plant，即 model mismatch 超過兩倍的情況時，傳統 DOB 架構就無法處理。

另外，傳統 DOB 架構中的 inverse plant 部分也會產生幾個問題，一是當系統為 non-minimum phase 時，plant 的零點會落在右半平面，則 inverse plant 就會產生右半平面的極點，導致整個系統不穩定；二是當系統為 non-square MIMO 時，無法直接求得 inverse plant，因此使得傳統 DOB 無法處理此種情況。

因此，針對傳統 DOB 無法處理 non-minimum phase 以及 MIMO 的系統，並且為了加強 DOB 對特定狀況的處理，接下來將提出實驗室發展的一個 Doubly Coprime Factorization DOB 架構，透過此架構，我們可以克服 non-minimum phase 以及 MIMO 所產生的問題，並且可以為了抑制特定的干擾設計特定的控制器，藉此可將傳統 DOB 過於分散的抑制能力集中對抗某些特定干擾且不犧牲太多原來的壓抑性能。



5.2 Doubly Coprime Factorization DOB 架構

因傳統 DOB 的架構無法處理 non-minimum phase 以及 MIMO 的系統，所以實驗室發展出一套 DCF DOB 架構，其中 Doubly Coprime Factorization 簡稱為 DCF，透過此架構，可以處理上面所說的傳統 DOB 無法處理的情況，DCF DOB 架構圖如下所示。

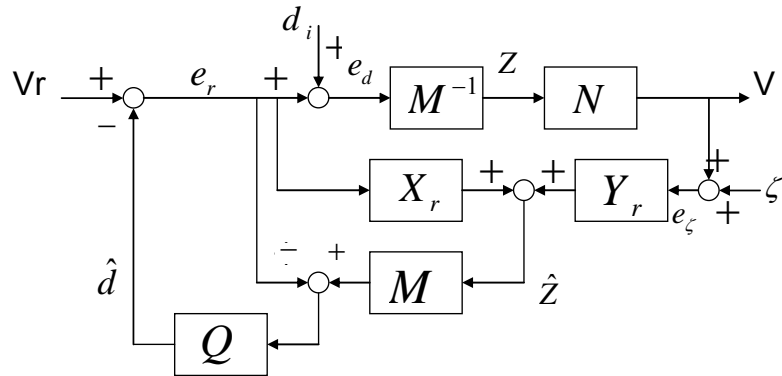


圖 5.3 Doubly Coprime Factorization DOB 架構

下列為圖 5.3 中參數定義。

V_r : 輸入命令(command input) d_i : 輸入干擾(input disturbance)

V : 系統輸出響應(output response) ζ : 高頻雜訊(noise)

\hat{d} : 干擾的估測值(estimated disturbance)

Q : DCF DOB 之控制器

根據[10]中提到，任意一系統轉移函數 P 皆可分解為：

$$P = NM^{-1} = \tilde{M}^{-1}\tilde{N} \quad (5.4)$$

其中 $N, M \in RH_\infty$ ，稱為稱為右互質分解(Right Coprime Factorization)，

$\tilde{N}, \tilde{M} \in RH_\infty$ ，稱為稱為左互質分解(Left Coprime Factorization)。

此外亦可找出一相對應之互質分解 $X_r, Y_r, X_l, Y_l \in RH_\infty$ 必須滿足 Bezout Identity :

$$\begin{cases} X_r M + Y_r N = I \\ Y_l \tilde{N} + X_l \tilde{M} = I \end{cases} \quad (5.5)$$

以及 Doubly Coprime Factorization :

$$\begin{bmatrix} X_r & Y_r \\ -\tilde{N} & \tilde{M} \end{bmatrix} \begin{bmatrix} M & -Y_l \\ N & X_l \end{bmatrix} = \begin{bmatrix} M & -Y_l \\ N & X_l \end{bmatrix} \begin{bmatrix} X_r & Y_r \\ -\tilde{N} & \tilde{M} \end{bmatrix} = I \quad (5.6)$$

而對應的互質分解 X_l 、 X_r 、 Y_l 、 Y_r 、 M 、 \tilde{M} 、 N 、 \tilde{N} 可由下面方程式求得：

$$\begin{bmatrix} M & -Y_l \\ N & X_l \end{bmatrix} = \left[\begin{array}{c|c|c} A+BF & B & -L \\ \hline F & I & 0 \\ \hline C+DF & D & I \end{array} \right], \begin{bmatrix} X_r & Y_r \\ -\tilde{N} & \tilde{M} \end{bmatrix} = \left[\begin{array}{c|c|c} A+LC & -(B+LC) & L \\ \hline F & I & 0 \\ \hline C & -D & I \end{array} \right] \quad (5.7)$$

其中 A 、 B 、 C 、 D 為系統狀態矩陣， F 為控制增益矩陣(control gain matrix)、 L 為觀測增益矩陣(observe gain matrix)。由圖 5.3 架構推導 3x3 系統轉移函數矩陣可得：

$$\begin{bmatrix} e_r \\ e_d \\ e_\zeta \end{bmatrix} = \begin{bmatrix} I & -Q(I-MX_r) & -QMY_r \\ I & I-Q(I-MX_r) & -QMY_r \\ NM^{-1} & NM^{-1}(I-Q(I-MX_r)) & I-NM^{-1}QMY_r \end{bmatrix} \begin{bmatrix} V_r \\ d_i \\ \xi \end{bmatrix} \quad (5.8)$$

假設系統 $P = P_n = NM^{-1} \in RH_\infty$ ，因 $N, M, X_r, Y_r, Q \in RH_\infty$ 則由上式轉移函數矩陣可知整體系統為內部穩定。

考慮此 DCF DOB 架構對於干擾的排除能力，我們由(5.8)式可知 $d_i \rightarrow e_d$ 之轉移函數為 $I-Q(I-MX_r)$ ，因此若能設計 Q 使得 $I-Q(I-MX_r) \approx 0$ ，就可以排除輸入干擾對於系統的影響，因此配合(5.6)式我們知道 $Q = (I-MX_r)^{-1} = Y_l \tilde{N}^{-1}$ 時可達到要求。但為了系統內部穩定性以及可實現性， Q 的設計必須滿足兩個要求：

1. $(I-MX_r)$ 為 invertible，即 $Y_l \tilde{N}^{-1}$ 為 realizable。
2. $(I-MX_r)$ 無右半平面零點，及 $Y_l \tilde{N}^{-1}$ 為 stable。

在[10]中提到：

根據條件 1.，經由推導 $(I - MX_r)$ 為 strictly proper，則 $(I - MX_r)^{-1} = Y_l \tilde{N}^{-1}$ 為 improper 且 unrealizable。

根據條件 2.，若系統含有右半平面零點，則 \tilde{N}^{-1} 也會含有右半平面零點，因此 $Y_l \tilde{N}^{-1}$ 為 unstable。

因此，為了適用在非極小相位、不穩定的系統上， Q 的設計不能為 $Q = Y_l \tilde{N}^{-1}$ ，所以在[10]中分別對於『穩定-極小相位』、『不穩定-極小相位』、『穩定-非極小相位』、『不穩定-非極小相位』四種不同狀況分別提出 Q 的設計方式。

在[10]中經由推導發現，『穩定-極小相位』與『不穩定-極小相位』兩種情況之下，不穩定部分可藉由外迴路的控制達到穩定的效果，因此在極小相位情況之下 Q 的設計方式相同。我們可將 Q 設計為：

$$Q(s) = J \cdot (I - M_n X_r)^{-1} = J \cdot (Y_l \cdot \tilde{N}_n)^{-1} \quad (5.9)$$

其中 J 為 low-pass filter，且因為是極小相位系統，所以 $Y_l \tilde{N}^{-1}$ 並無右半平面之零點，因此此設計方式可符合條件 1.、2.。

至於『穩定-非極小相位』與『不穩定-非極小相位』兩種情況之，不穩定的部分可藉由外迴路的控制達到穩定的效果，非極小相位部分則需使用 Model Matching method 以及 Weighting design 來設計 Q ，在[10]中有詳細推導在此不多加敘述。肯定的是，透過適當的方式設計 Q 的形式，DCF DOB 架構能夠處理 non-minimum phase 等傳統 DOB 架構無法處理的狀況。

在本文中，因為馬達系統為『穩定-極小相位』的情況，因此我們將採用(5.9)式 Q 的設計形式，將系統輸出的方程式表示如下：

$$V = NM^{-1}V_r + (I - Q(I - MX_r))NM^{-1}d_i - QMY_rNM^{-1}\xi \quad (5.10)$$

其中輸入命令 V_r 對輸出 y 的轉移函數為 $G_{V,V}$ ，而輸入干擾對輸出的轉移函數為 $G_{d,V}$ ，雜訊對輸出的轉移函數 $G_{\zeta V}$ 表示如下：

$$\begin{aligned} G_{V,V} &= NM^{-1} \\ G_{d,V} &= (I - Q(I - MX_r))NM^{-1} \\ G_{\zeta V} &= QMY_rNM^{-1} \end{aligned} \quad (5.11)$$

將(5.9)式代入(5.11)式中可以知道：

$$\begin{aligned} G_{V,V} &= NM^{-1} \\ G_{d,V} &= (I - J)NM^{-1} \\ G_{\zeta V} &= J \end{aligned} \quad (5.12)$$



從(5.12)式我們知道，若想要壓低輸入干擾對於輸出的影響， $J(s)$ 需為 1，若想排除雜訊的影響， $J(s)$ 需為 0，但機械系統通常為低頻而雜訊通常為高頻，所以 low-pass filter $J(s)$ 的設計形式考參考前面傳統 DOB 的設計，至於如何設計 $J(s)$ 的頻寬，也必須在排除干擾與抑制雜訊之間做一選擇。

由以上討論我們可以知道當 DCF DOB 應用在 non-minimum phase、SISO 的系統之中，傳統 DOB 擁有的優點 DCF DOB 架構同樣也有，但稍後在 5.5 節之中，將會介紹 DCF DOB 優於傳統 DOB 的地方，也就是對抗特定干擾的能力。

5.3 伺服迴路控制架構

在本文中整個系統的控制架構如圖 5.4 所示，內迴路以 DOB 改善系統的不確定性、排除系統所受到的干擾及雜訊，使系統逼近期望的系統(nominal plant)，以方便外迴路控制器的設計，在速度迴路及位置迴路中由迴授及前饋控制器來改善系統穩定度和追蹤命令的能力。

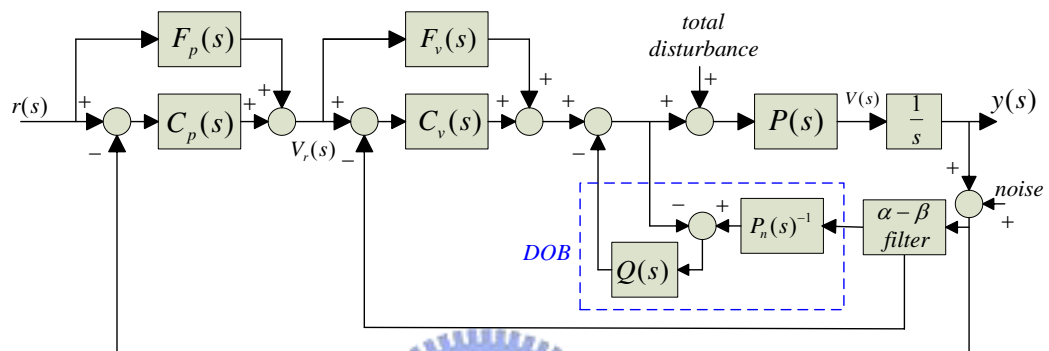


圖 5.4 伺服迴路控制架構

其中：

$P(s)$ 為 真實系統(Actual plant)

$P_n(s)$ 為 期望的系統(Nominal plant)

$Q(s)$ 為 DOB 的低通濾波器(Low-pass filter)

$C_p(s)$ 為 位置迴授控制器(Position feedback controller)

$F_p(s)$ 為 位置前饋控制器(Position feedforward controller)

$C_v(s)$ 為 速度迴授控制器(Velocity feedback controller)

$F_v(s)$ 為 速度前饋控制器(Velocity feedforward controller)

$y(s)$ 為 實際位置響應(Position response)

$v(s)$ 為 實際速度響應(Velocity response)

$r(s)$ 為 位置命令(Position command)

$V_r(s)$ 為 速度命令(Velocity command)

Total Disturbance 為 摩擦力、鈍齒力、磁阻力、系統不確定性、等外界干擾總和

Noise 為 量測時受到的雜訊

在伺服迴路中，我們下達的位置命令或速度命令經過位置迴路控制以及速度迴路控制後輸入到馬達中，經由 Encoder 測量並回傳馬達的響應給控制器進行補償。從數學上看，DOB 架構可將系統逼近成 nominal plant，因此速度迴路以及位置迴路控制器的設計就方便許多，接下來將會對控制架構中的前饋控制以及位置、速度迴授控制做進一步的說明。

5.3.1 Feedforward Control

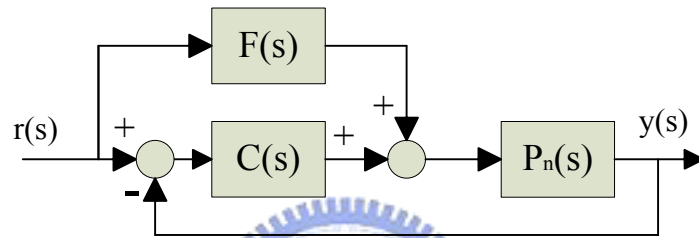


圖 5.5 前饋控制架構

圖 5.5 中表示的控制架構是由一個前饋控制以及一個迴授控制所組成，而其中的 $P_n(s)$ 代表 nominal plant、 $C(s)$ 為迴授控制器、 $F(s)$ 為前饋控制器、 $r(s)$ 表示輸入命令、 $y(s)$ 則代表輸出響應，則系統的轉移函數可表示成：

$$\frac{y(s)}{r(s)} = \frac{F(s)P_n(s) + C(s)P_n(s)}{1 + C(s)P_n(s)} \quad (5.13)$$

在上式中若 $F(s)P_n(s) = 1$ ，則輸入輸出關係將呈現出 $y(s) = r(s)$ ，表示此系統的響應速度快、並且沒有穩態誤差。從另一個角度來看，如果真的能夠讓 $F(s)P_n(s) = 1$ ，我們甚至不需要迴授控制(當 $C(s) = 0$ 時)也能讓系統有良好的響應，但實際上基於干擾、雜訊、系統不確定性的影響，我們仍然需要迴授控制讓系統有較好的追跡響應。

5.3.2 Velocity Loop Control

在速度迴路的設計中，我們同樣以一個迴授控制加上一個前饋控制來進行控制，其中 $P_n(s)$ 代表 nominal plant、 $C_v(s)$ 為迴授控制器、 $F_v(s)$ 為前饋控制器、 $V_r(s)$ 表示速度命令、 $V(s)$ 代示輸出速度響應，因為前饋控制器的設計不影響閉迴路的極點，所以在已設計完成的閉迴路中加上前饋控制器不會影響已設計好的系統之穩定性，換句話說，我們在設計迴授控制器時也不需考慮前饋控制器的影響。

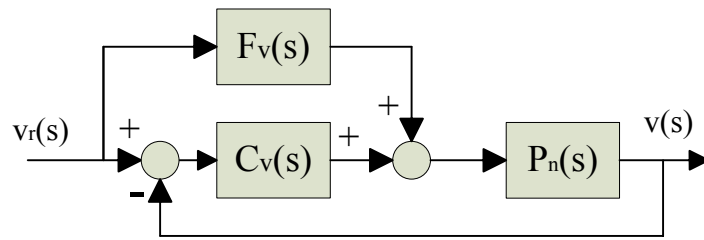


圖 5.6 速度迴路方塊圖

考慮系統為線性系統，馬達系統的 nominal plant $P_n(s)$ 為：

$$P_n(s) = \frac{K_a K_t}{J_n s + B_n} \quad (5.14)$$

其中 K_a 為馬達電流常數、 K_t 為馬達力矩常數、 J_n 為馬達慣量、 B_n 為黏滯摩擦係數。從前一節的討論我們知道前饋控制器 $F(s)$ 可設計為：

$$F_v(s) = \frac{1}{P_n(s)} = \frac{J_n s + B_n}{K_a K_t} \quad (5.15)$$

上式中 $F_v(s)$ 並非 proper function，因此在實現上會需要進行微分的動作，在本文中將以 $\alpha - \beta$ filter 來進行微分的實現，其特色在於低頻時呈現微分的效果，但在高頻時並沒有微分效果，因此與一般的微分器相比，高頻部分雜訊比較不會被放大。承上 $F_v(s)$ 的設計，圖 5.6 的速度閉迴路轉移函數如下：

$$L_v(s) = \frac{F_v(s)P_n(s) + C_v(s)P_n(s)}{1 + C_v(s)P_n(s)} \cong 1 \quad (5.16)$$

若 $F_v(s)$ 不完全等於系統反函數 $P_n(s)^{-1}$ ，此時就需要靠迴授控制器來補償追跡誤差，考慮排除干擾、壓抑雜訊等需求，本文中速度迴路迴授控制器將選擇 PI 控制器，如下：

$$C_v(s) = K_{vp} + \frac{K_{vi}}{s} \quad (5.17)$$

其中 K_{vp} 為比例放大增益， K_{vi} 為積分增益，而 $C_v(s)$ 的設計可以用 pole-placement 的方式來設計，閉迴路轉移函數 $L_v(s)$ 的特性方程式為：

$$J_n s^2 + (B_n + K_a K_t K_{vp})s + K_a K_t K_{vp} = 0 \quad (5.18)$$

考慮標準二階系統的閉迴路特性方程式，以 ξ 表示阻尼係數、 ω_n 表示系統的自然頻率，並且與(5.9)式比較可知：

$$s^2 + \frac{(B_n + K_a K_t K_{vp})}{J_n} s + \frac{K_a K_t K_{vp}}{J_n} = s^2 + 2\xi\omega_n s + \omega_n^2 \quad (5.19)$$

因此藉由設計標準二階系統的 ξ 以及 ω_n (設計標準二階系統的 pole)，經由比較係數後可以得到，迴授控制器的兩項參數：

$$\begin{cases} K_{vp} = \frac{2\xi\omega_n J_n - B_n}{K_a K_t} \\ K_{vi} = \frac{\omega_n^2 J_n}{K_a K_t} \end{cases} \quad (5.20)$$

5.3.3 Position Loop Control

位置迴路之中方塊圖如圖 5.7 所示，其中 $C_p(s)$ 為位置迴路迴授控制器； $F_p(s)$ 為位置迴路前饋控制器； $r(s)$ 以及 $y(s)$ 為位置命令以及系統位置響應。

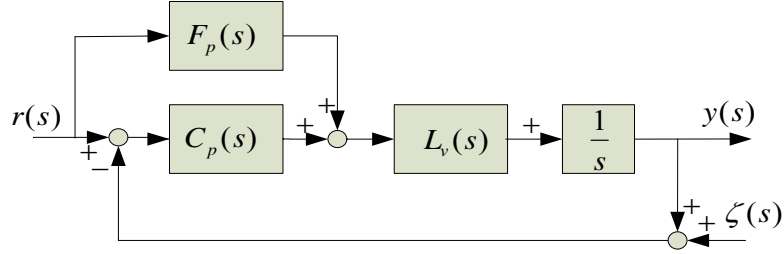


圖 5.7 位置迴路方塊圖

根據前面的推導說明，位置迴路的前饋控制器 $F_p(s)$ 應該設計為 $L_v(s) \cdot \frac{1}{s}$ 的反函數，且速度迴路的頻寬應大於位置迴路許多，假設理想狀況之下速度迴路 $L_v(s) = 1$ ，此時位置迴路前饋控制器 $F_p(s)$ 可設計為：

$$F_p(s) = s \quad (5.21)$$

又，(5.21)式並非 proper function，因此在實現上我們同樣會以 $\alpha - \beta$ filter 來進行微分的實現，而迴授控制器的部分將簡單的選擇比例控制器來設計：

$$C_p(s) = K_{pp} \quad (5.22)$$

其中 K_{pp} 為比例放大增益。接著，推導出位置迴路輸入 $r(s)$ 到輸出 $y(s)$ 的轉移函數為：

$$L_p(s) = \frac{F_p(s)L_v(s) \cdot \frac{1}{s} + C_p(s)L_v(s) \cdot \frac{1}{s}}{1 + C_p(s)L_v(s) \cdot \frac{1}{s}} \cong 1 \quad (5.23)$$

其中 K_{pp} 的選擇在本文中參考[16]的設計，選擇速度迴路之自然頻率之倍數作為 K_{pp} 設計的

依據，在文中選擇 $\frac{1}{8}\omega_n \sim \frac{1}{12}\omega_n$ ，如下：

$$K_{pp} = \frac{1}{8}\omega_n \sim \frac{1}{12}\omega_n \quad (5.24)$$

5.4 系統參數鑑別

在設計伺服迴路控制器參數時，需要使用系統的慣量以及黏滯摩擦係數，因此，在設計控制器之前必須先找出系統本身的參數。在本文中將會使用[17]提到的 DOB 系統鑑別來找出系統的參數，並將鑑定得到的參數用來設計內外迴路控制器以及 DOB 控制器。如下圖 5.8 為 DOB 系統鑑別的架構。

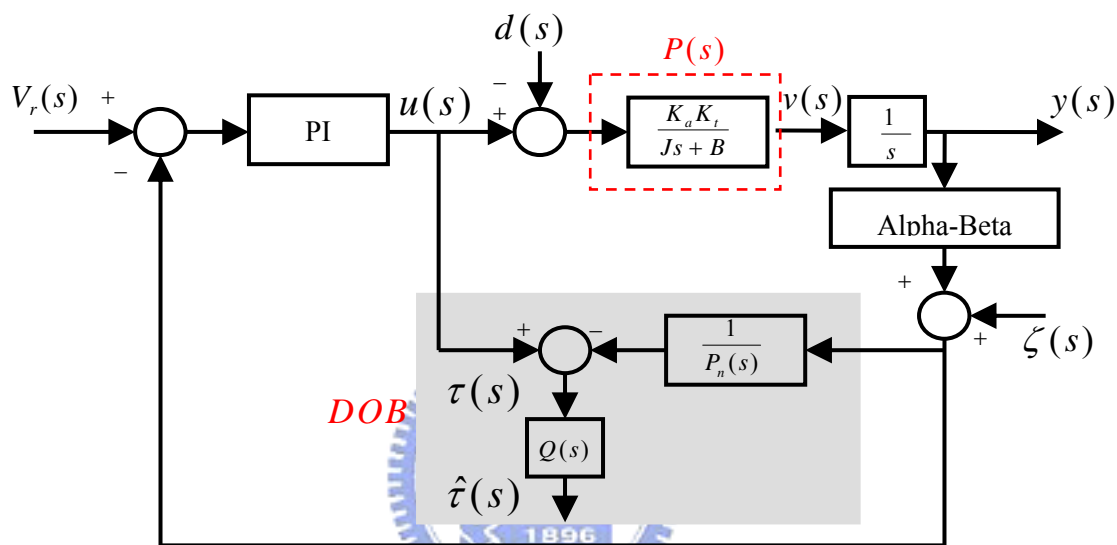


圖 5.8 含 PI 控制器之速度迴路架構圖

其中

$y(s)$ ：機台實際位移

$v(s)$ ：機台實際速度。

$\zeta(s)$ ：估測誤差或量測誤差。

$P(s) = \frac{1}{Js + B}$ ：實際系統轉移函數。

$P_n(s) = \frac{1}{J_n s + B_n}$ ：期望系統轉移函數。

$u(s)$ ：控制器輸入。

$d(s)$ ：干擾輸入。

$Q(s)$ ：低通濾波器。

τ 與 $\hat{\tau}$ 分別為未經過與經過低通濾波器之估測干擾力。

在先不考慮低通濾波器 $Q(s)$ 的作用下，由圖 5.8 我們可得

$$\tau(s) = u(s) - \frac{1}{P_n(s)}(v(s) + \zeta(s)) \quad (5.25)$$

將 $u(s) = d(s) + \frac{1}{P(s)}v(s)$ ，及 $P(s) = \frac{1}{Js + B}$ ， $P_n(s) = \frac{1}{J_n s + B_n}$ 代入式中，可得到

$$\begin{aligned} \tau &= d + J\dot{v} + Bv - J_n\dot{v} - B_nv - (J_n\dot{\zeta} + B_n\zeta) \\ &= d + \Delta J\dot{v} + \Delta Bv - (J_n\dot{\zeta} + B_n\zeta) \end{aligned} \quad (5.26)$$

其中 $J = J_n + \Delta J$ ， $B = B_n + \Delta B$ 。此時將 $Q(s)$ 考慮進來，若能設計低通濾波器 $Q(s)$ 的頻寬介於高頻的雜訊 $\zeta(s)$ 、 $\dot{\zeta}(s)$ 與低頻的外部干擾 d 及運動命令 v_r 之間，如此則可以消除與量測誤差相關項 $J_n\dot{\zeta} + B_n\zeta$ 。因此估測力可以表示為：

$$\hat{\tau} \cong d + \Delta J\dot{v} + \Delta Bv \quad (5.27)$$

以下開始介紹估測慣量的步驟。圖 4.15 為該實驗的架構，圖中速度參考命令為一有偏置

(offset) 之弦波週期函數， $v_r(t) = v_0 + v_1 \sin\left(\frac{2\pi}{T_p}t\right)$ ，週期為 T_p ， $v_0 > v_1 > 0$ ，而且速度 v_r

最低速度不要太接近零，避免進入低速時的非線性摩擦區。利用 PI 速度控制器使速度響應 v_r 在一段時間後除了可忽略的雜訊外與大致命令相同。因為機台沒有發生換向運動，當開始運動一段時間後，其摩擦力脫離了 Stribeck Effect 區，只剩庫倫摩擦力（與速度成正比的黏滯摩擦力則歸類在與系統參數項 Bv 上），而庫倫摩擦力是一個固定常數。將估測力的每一項乘上 \dot{v} 並對之積分一個週期，此一週期必須選擇速度響應已經追上參考命令之時期，如此可寫成下式：

$$\int_{T_p} (\hat{\tau} \dot{v}) dt = \int_{T_p} (\Delta J \dot{v} \dot{v}) dt + \int_{T_p} (\Delta B v \dot{v}) dt + \int_{T_p} (d \dot{v}) dt \quad (5.28)$$

因為 $v > 0$ ，且除摩擦力外並未施加其他外力，則 $d = T_d$ 。而 T_d 、 ΔB 皆為常數，可藉由弦波函數的正交特性，即 $\int_{T_p} (\Delta B v \dot{v}) dt = \int_{T_p} (T_d \dot{v}) dt = 0$ ，因此可計算出

$$\Delta J = \frac{\int_{T_p} (\hat{\tau} \dot{v}) dt}{\int_{T_p} (\dot{v})^2 dt} \approx \frac{\sum_{k=1}^{k=N} \hat{\tau}[k] \dot{v}[k] T}{\sum_{k=1}^{k=N} \dot{v}^2[k] T} = \frac{\sum_N \hat{\tau}[k] \dot{v}[k]}{\sum_N \dot{v}^2[k]} \quad (5.29)$$

其中 N 為一個週期的取樣數， T 為取樣間隔時間。 $\hat{\tau}[k]$ 與 $\dot{v}[k]$ 分別代表一週期中 $\hat{\tau}$ 與 \dot{v} 的第 k 個取樣值。系統慣量估測值於是修正為：

$$\hat{J}_{new} = \hat{J}_{old} + \Delta J \quad (5.30)$$

利用相近的方法也可求 ΔB

$$\Delta B = \frac{\sum_N \hat{\tau}[k]v[k] - T_d \sum_N v[k]}{\sum_N v^2[k]} \quad (5.31)$$

而黏滯阻尼係數 B 的估測變成

$$\hat{B}_{new} = \hat{B}_{old} + \Delta B \quad (5.32)$$

注意到 ΔB 中包含著庫倫摩擦力 T_d ，意即在估測 B 之前需要先求得 T_d 的值。而至於 T_d 的估測，由估測力可知，原本 $\hat{\tau}$ 是用來估測所有的外力干擾，但是估測值會受 ΔJ 與 ΔB 的影響。假如將實驗條件控制到外力只剩庫倫摩擦力 T_d ，當經由實驗使 ΔJ 與 ΔB 均收斂至零時，這時候 $\hat{\tau} = d = T_d$ 。也就是說，理論上當慣量與黏滯摩擦力都能鑑定得很精確時， $\hat{\tau}$ 與時間的關係應為一直線。在實際的操作上可以將各時間的 $\hat{\tau}$ 值取其平均而求得庫倫摩擦力的近似估測值 \hat{T}_d ：

$$\hat{T}_d = \frac{1}{N} \sum_{k=1}^{k=N} \hat{\tau}[k] = \frac{1}{N} \sum_N \hat{\tau}[k] \quad (5.33)$$

將估測力代入上式中：

$$\hat{T}_d = \frac{1}{N} \Delta J \sum_{k=1}^{k=N} \dot{v}[k] + \frac{1}{N} \Delta B \sum_{k=1}^{k=N} v[k] + d \quad (5.34)$$

若取一週期的速度響應結果來進行計算，在一週期內的加速度和為零，因此 \hat{T}_d 只剩下：

$$\hat{T}_d = \frac{1}{N} \Delta B v_0 + d \quad (5.35)$$

其中 v_0 假設為一常數。由上式可看出，當一週期內的取樣數夠多，且經過多次疊代後使得 $\Delta B \rightarrow 0$ ，則估測值 \hat{T}_d 將接近於庫倫摩擦力。

在 DOB 參數鑑定中所推導出來的(5.25)~(5.27)式，其中速度 $v(t)$ 的部分應該是實際速度響應，但為了往後方程式推導的方便，我們必須做一個假設，我們必須假設實際

速度響應 $v(t)$ 逼近於速度命令 $V_r(t)$ ，因此才能使用正弦函數的特性由(5.28)式推導出(5.29)、(5.30)式。因此在進行 DOB 系統鑑別之前，必須先對系統的 PI 控制器進行調整，來滿足 $v(t) = V_r(t)$ 的先決條件。根據實驗架構圖 5.8，並且以線性馬達 XY 平台之 Y 軸為例，下達 $v_r(t) = 30 + 20\sin(5t)$ mm/s 的速度參考命令，利用 PI 控制器，調整增益值以獲得良好的速度追蹤，如下圖 5.9 所示。

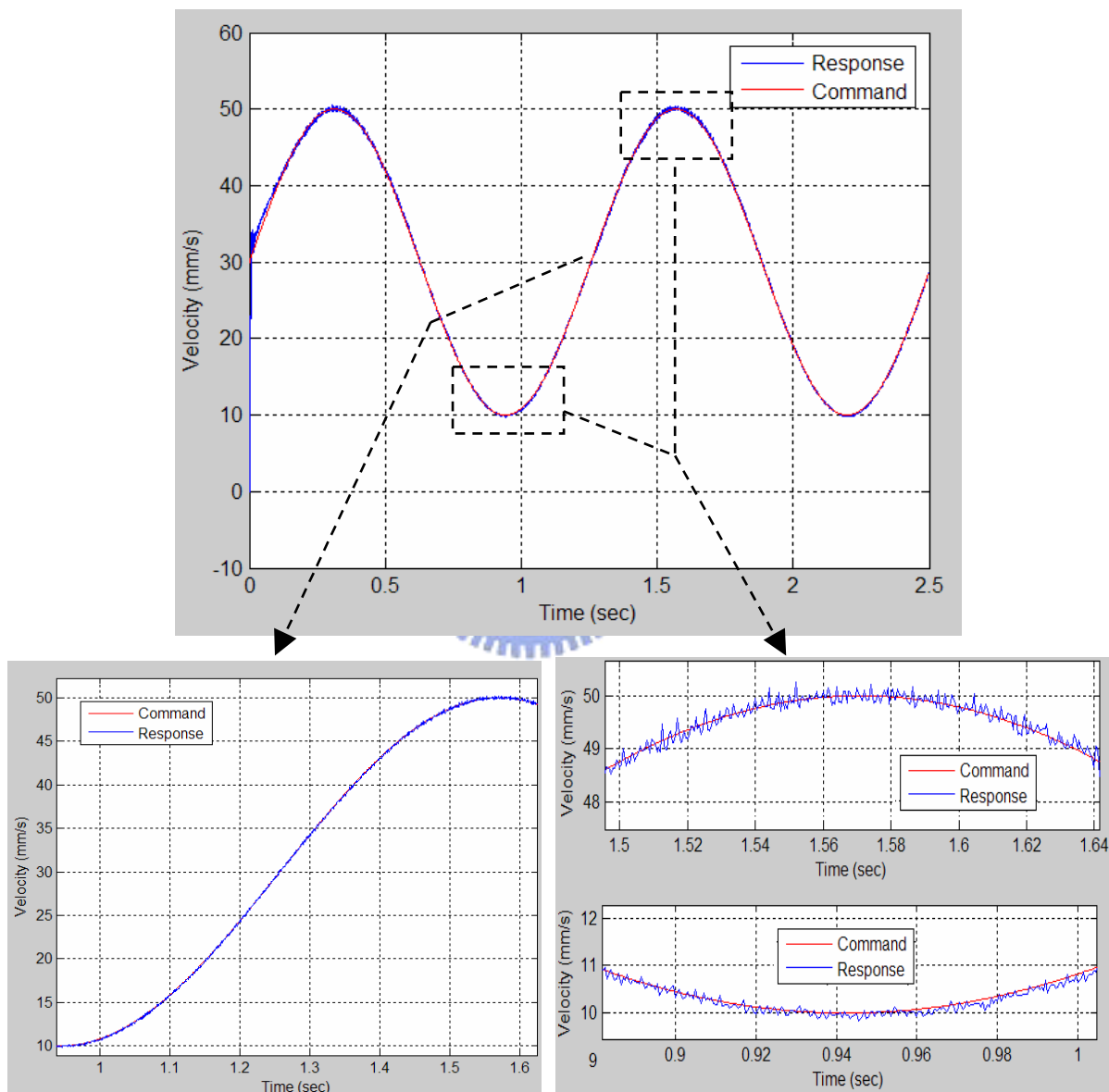


圖 5.9 速度參考命令與輸出響應圖

由圖 5.9 中可知過了暫態響應之後，輸出速度響應與輸入命令非常相近，因此滿足 DOB 鑑別的前提，即 $v(t) = V_r(t)$ 。整個 DOB 系統鑑別的步驟如下所示：

Step 1：調整圖 5.8 速度迴路 PI 控制器，使得系統有良好的速度追蹤響應，滿足

$$v(t) = V_r(t) \text{ 以及 } \dot{v}(t) = \dot{V}_r(t) \text{ 兩個假設條件。}$$

Step 2：以速度命令 $V_r(t)$ 取代實際速度 $v(t)$ ，並以(5.29)式計算 ΔJ 。

$$\Delta J = \frac{\sum_N \hat{\tau}[k] \dot{V}_r[k]}{\sum_N \dot{V}_r^2[k]}$$

Step 3：更新估測的系統慣量。

$$\hat{J}_{new} = \hat{J}_{old} + \Delta J$$

Step 4：以(5.33)式估測庫崙摩擦力。

$$\hat{T}_d = \frac{1}{N} \sum_N \hat{\tau}[k]$$

Step 5：以速度命令 $V_r(t)$ 配合(5.31)以及(5.33)式計算 ΔB 。

$$\Delta B = \frac{\sum_N \hat{\tau}[k] V_r[k] - \hat{T}_d \sum_N V_r[k]}{\sum_N V_r^2[k]}$$

Step 6：更新估測的系統黏滯摩擦係數。

$$\hat{B}_{new} = \hat{B}_{old} + \Delta B$$

Step 7：重複進行 Step 1 ~ Step 6，一直到 ΔJ 、 ΔB 趨近於零，即可停止疊代，

並且記錄所得到的系統參數。DOB 鑑定結果如下表所示：

Symbol and name	Value	Unit
J_n ，系統慣量	2.92733	kg
B_n ，黏滯摩擦係數	15.4386	kg/sec
T_d ，庫崙摩擦力	1.69275	N
K_a ，馬達力矩常數	0.349	N/A
K_t ，馬達電流常數	28.5	A/Volt
T_s ，取樣時間	0.0005	Sec
d_p ，馬達 pole pitch	32.0	mm

表5.1 系統參數表

從上面我們知道了以 DOB 架構來進行系統參數鑑別，但在[17]中也提出了其他系統鑑別的方法，而在本文中也曾經以摩擦力實驗來進行系統參數鑑別，在此我們將進行摩擦力鑑定實驗以及 DOB 系統鑑別的討論。

我們先簡單介紹摩擦力實驗鑑定黏滯摩擦係數、庫崙摩擦力的概念，由[18]我們知道摩擦力可依照接觸面是否有明顯的相對運動分為靜摩擦力(Static Friction)、最大摩擦力(Striction)、庫崙摩擦力(Coulomb Friction)、黏滯摩擦力(Viscous Friction)、以及 Stribeck Friction 等等。以阿姆斯壯摩擦力模型為例，如圖 5.10 所示，從圖中我們可以清楚的得到黏滯摩擦係數以及庫崙摩擦力的參數，因此若我們能夠找出系統的摩擦力曲線，便可以得到系統的黏滯摩擦係數以及庫崙摩擦力。

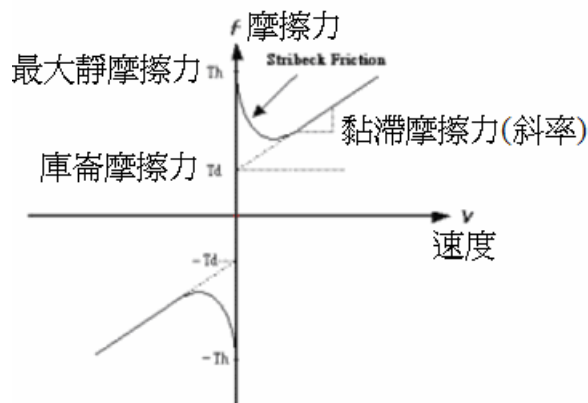


圖 5.10 阿姆斯壯摩擦力模型

在本文中為了增加人工輸出干擾而對系統進行改裝，改裝前後以舊系統、新系統加以分別，新系統的參數鑑定如前面表 5.1 所示，而改裝前的舊系統曾經以[18]中的摩擦力實驗以及 DOB 系統鑑別分別進行系統黏滯摩擦係數的鑑定，透過摩擦力實驗，將舊系統摩擦力曲線繪出如下圖 5.11。由圖可知庫崙摩擦力 $T_d = 1.761$ (N)，黏滯摩擦係數 $B = 41.866$ (kg/sec)，而使用 DOB 系統鑑別所得到之舊系統參數 $T_d = 2.3906$ (N)、 $B = 11.5825$ (kg/sec)，我們發現兩種系統鑑別所得到的黏滯摩擦係數差異很大，因為摩擦力實驗曲線的可信度較高，因此討論可能造成 DOB 系統鑑別失準的原因，認為可能原因有以下幾點：

1. DOB 系統鑑別的推導，皆是假設 $v(t) = V_r(t)$ ，因此以速度命令 $V_r(t)$ 進行運算、疊代，若速度響應與速度命令不夠逼近，則可能會降低 DOB 系統鑑別的準確度。
2. DOB 系統鑑別的方式在疊代黏滯摩擦係數 B 的時候，需要使用到庫崙摩擦力的參數，因此有可能因為庫崙摩擦力估測的不準而影響 B 的準確度。
3. 摩擦力實驗與 DOB 系統鑑別並非在同一天進行，有可能因為外在的因素改變了黏滯摩擦係數，而造成兩實驗結果的差異。
4. 進行摩擦力實驗需要足夠的行程，讓速度響應進入穩態響應，但因為本文中線性平台之 Y 軸不夠長，因此可能造成穩態速度的估測不準，進而影響黏滯摩擦係數(斜率) B 的計算。

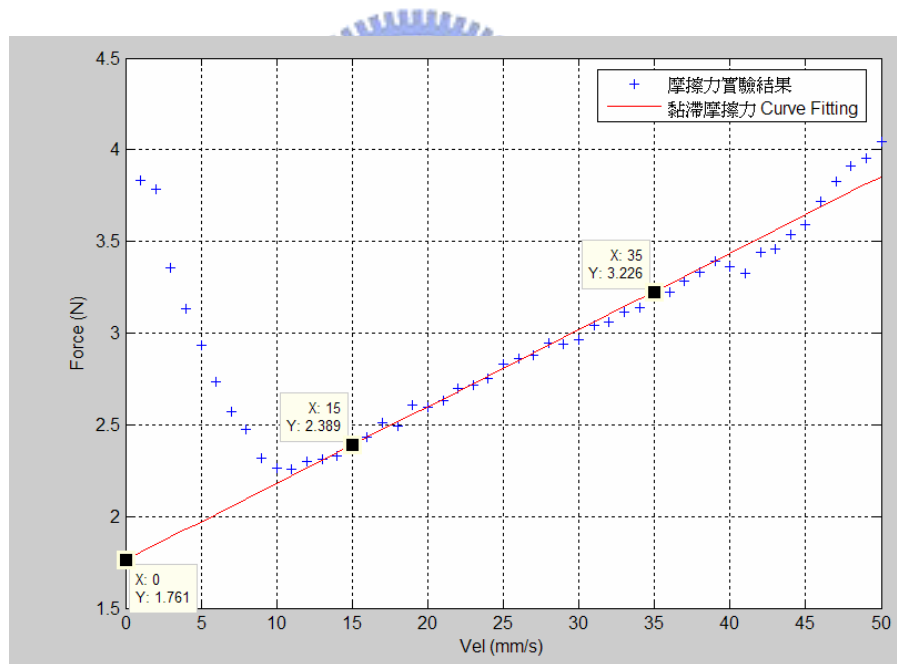


圖 5.11 舊系統的摩擦力實驗曲線

因為兩種鑑別方式都可能有問題存在，因此在新系統的鑑別中，只採用 DOB 系統鑑別來進行系統的參數鑑定，而得到新系統的參數如表 5.1 所示。

5.5 Output Disturbance

輸出干擾(output disturbance)，顧名思義，與輸入干擾不同的地方在於此干擾進入運動控制系統的時機不同。輸入干擾將會隨著運動命令進入 plant 進而影響系統的輸出，然而輸出干擾則不會進入 plant，而是直接影響系統輸出響應。一般來說機械系統的 plant 通常為 low-pass，因此輸入干擾的高頻部分對系統輸出的影響比較小，但輸出干擾沒有經過 plant，所以若組成成分相同的輸入干擾與輸出干擾，輸出干擾對系統響應的影響會比輸入干擾嚴重。為了對抗輸出干擾，我們必須使用閉迴路控制來補償輸出干擾的影響，圖 5.12 為簡單的示意圖。

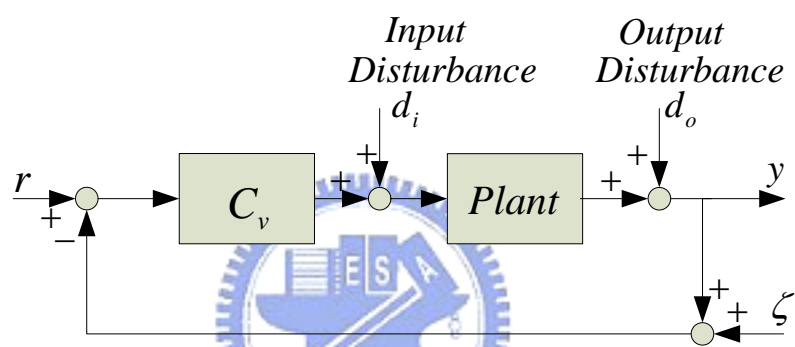


圖 5.12 輸入干擾與輸出干擾

因為 DOB 架構具有排除干擾的特性，我們將傳統 DOB 架構加入圖 5.12 系統之內迴路中，如圖 5.13 所示，而傳統 DOB 架構在數學上能夠排除 DOB 頻寬內之輸入干擾，所以下面將會推導加入輸出干擾之後系統的轉移函數矩陣。

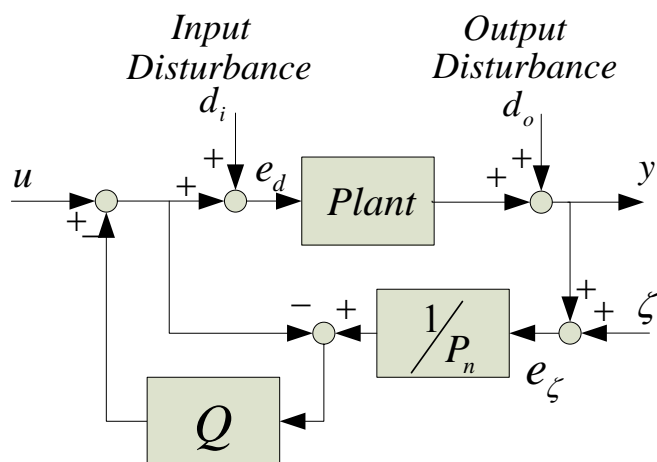


圖 5.13 以 DOB 架構對抗輸出干擾

在圖 5.13 中， u 為輸入、 y 為輸出、 d_i 為輸入干擾、 d_o 為輸出干擾、 ζ 為量測雜訊、 P_n 為 nominal plant、我們可以推導出 u 、 d_i 、 d_o 、 ζ 到 e_r 、 e_d 、 y 、 e_ζ 的 4x4 的轉移函數矩陣如下所示：

$$\begin{bmatrix} e_r \\ e_d \\ y \\ e_\zeta \end{bmatrix} = \frac{P_n}{P_n + (P - P_n)Q} \begin{bmatrix} 1 & -P_n^{-1}PQ & -P_n^{-1}Q & -P_n^{-1}Q \\ 1 & (1-Q) & -P_n^{-1}Q & -P_n^{-1}Q \\ P & P(1-Q) & (1-Q) & -P_n^{-1}PQ \\ P & P(1-Q) & (1-Q) & (1-Q) \end{bmatrix} \begin{bmatrix} u \\ d_i \\ d_o \\ \zeta \end{bmatrix} \quad (5.36)$$

上式中轉移函數矩陣中每一元素皆為穩定，因此圖 5.13 系統為內部穩定，且由上式我們發現輸出干擾 d_o 對系統輸出 y 的轉移函數為：

$$G_{d_o, y}(s) = \frac{P_n(s)(1-Q(s))}{P_n(s) + (P(s) - P_n(s))Q(s)} \quad (5.37)$$

因此從數學上我們知道在 SISO 的系統中，傳統 DOB 架構同樣可以排除輸出干擾，如前幾節所敘述，設計 $Q(s)$ 為一 low-pass filter 在 $Q(s)$ 頻寬之內的輸入干擾、輸出干擾都可以排除，且頻寬以外的雜訊也會被過濾掉。

在前面討論 DOB 架構缺點時曾經提出，DOB 架構對於高頻干擾沒有很好的排除效果，綜合本節前面敘述以及(5.36)式可知，雖然 DOB 無法排除高頻干擾，但高頻的輸入干擾經過 plant 後會被壓低一些，可是高頻的輸出干擾會直接影響系統輸出。因此，我們為了壓抑特定高頻的干擾，又不希望雜訊被放大，傳統的 DOB 無法應付此需求，所以我們在此將展現實驗室發展之 DCF DOB 架構之優點之一，針對某種特定干擾的抑制能力比傳統 DOB 強，回顧圖 5.3 之 DCF DOB 架構並且加入輸出干擾，

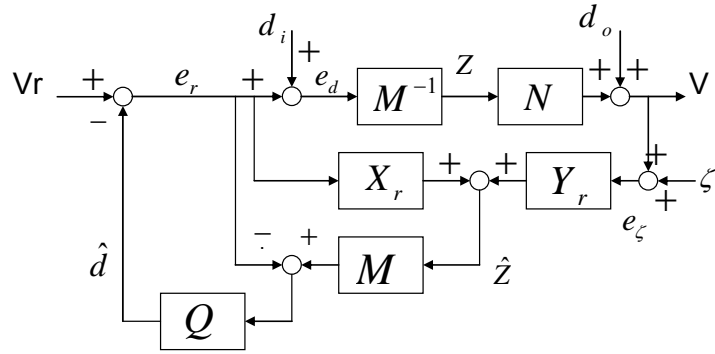
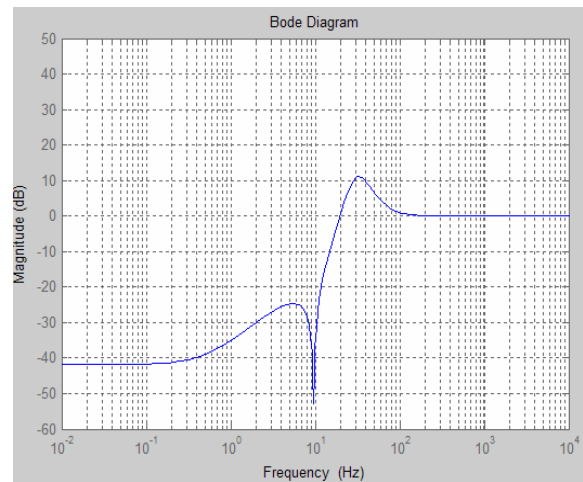
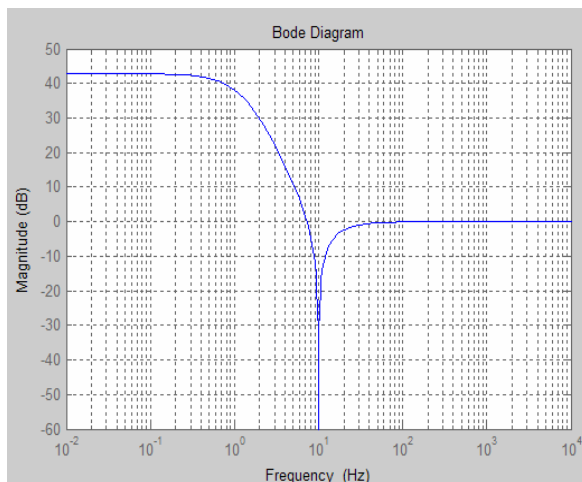


圖 5.14 加入輸出干擾的 DCF DOB 架構

同前面傳統 DOB 架構，推導此系統的轉移函數矩陣，如下所示：

$$\begin{bmatrix} e_r \\ e_d \\ V \\ e_\zeta \end{bmatrix} = \begin{bmatrix} I & -Q(I - MX_r) & -QMY_r & -QMY_r \\ I & I - QY_l\tilde{N} & -QMY_r & -QMY_r \\ NM^{-1} & (I - QY_l\tilde{N})NM^{-1} & (I - QY_l\tilde{N}) & QY_l\tilde{N} \\ NM^{-1} & (I - QY_l\tilde{N})NM^{-1} & (I - QY_l\tilde{N}) & (I - QY_l\tilde{N}) \end{bmatrix} \begin{bmatrix} V_r \\ d_i \\ d_o \\ \xi \end{bmatrix} \quad (5.38)$$

其中 $P_n = NM^{-1}$ ， Q 為 DCF DOB 之控制器，由上式可知，輸入干擾與輸出干擾轉移函數與 $(I - QY_l\tilde{N})$ 相關，若我們設計 Q 使得 d_o 到 V 的轉移函數 $G_{d_o,V}(s) = I - QY_l\tilde{N}$ 之頻率響應形成一個 Notch filter，在特定頻率砍掉輸出干擾，則可以排除高頻輸出干擾對系統的影響又不犧牲雜訊的抑制能力。依據不同的需求，可以設計不同形式的 Notch filter 來進行控制，下圖表示幾種不同的 Notch filter，此部分的實驗將在後面進行討論說明。



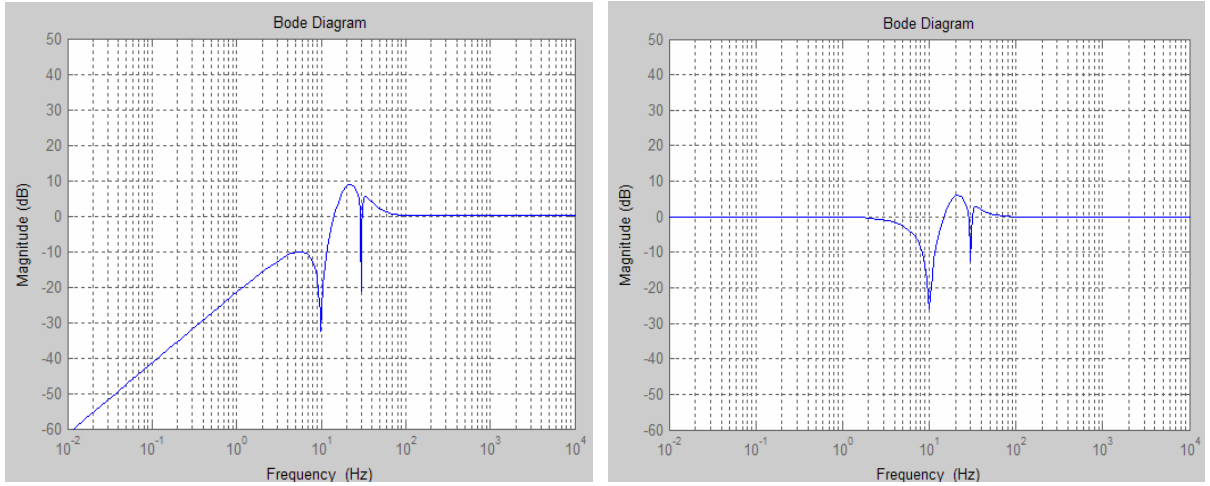


圖 5.15 設計 $I - QY_1\tilde{N}$ 呈現不同形式的 notch

由圖我們可以知道 DCF DOB 可以設計不同的 $Q(s)$ 來形成不同形式的 $G_{d_oV}(s)$ ，接下來將會介紹 DCF DOB 設計成 notch 形式時 $Q(s)$ 的設計方式。

根據[10]中提到的設計方式，在『穩定-極小相位』中 $Q(s)$ 的設計方式如(5.9)式所表示的 $Q(s) = J \cdot (Y_l \cdot \tilde{N}_n)^{-1}$ ，所以為了對 $Q(s)$ 進行特殊的設計，我們必須將 $Q(s)$ 強制轉變成『非極小相位』系統，然後依照[10]中『穩定-非極小相位』系統的設計步驟設計 $Q(s)$ 。

Step 1. Model Matching 的設計方式來設計 $Q(s)$ ：

由上面可知 d_o 到 V 的轉移函數 $G_{d_oV}(s) = I - QY_1\tilde{N} = I - Q(I - MX_r)$ ，在此架構中我們的設計目標如下

$$\min_{Q \in RH_\infty} \|I - Q(I - M_n X_r)\|_\infty < \gamma \quad (5.39)$$

即是壓抑轉移函數 $G_{d_oV}(s)$ 至某個範圍 γ 以下，經由推導我們可以得知，

$$\min_{Q \in RH_\infty} \|I - Q(I - M_n X_r)\|_\infty = \min_{Q \in RH_\infty} \|X_r M_n + \hat{Q} Y_r N_n\|_\infty = \min_{Q \in RH_\infty} \|T_1 + T_2 \hat{Q} T_3\|_\infty \quad (5.40)$$

其中 $T_1 = X_r M_n \in RH_\infty$ 、 $T_2 = I$ 、 $T_3 = Y_r N_n \in RH_\infty$ 、 $\hat{Q} = (I - Q) \in RH_\infty$ ，形成 model matching 的設計問題。

Step 2. 求解 Nehari Problem :

由 Step 1 的 model matching 問題再經過推導、拆解 outer-inner factorization

$T_3 = T_{3o} \cdot T_{3i}$ ，其中 T_{3o} 為 outer function、 T_{3i} 為 inner function、並且滿足

$$\begin{cases} T_{3i}^* \cdot T_{3i} = I, & T_{3i}^* = T_{3i}^T(-s) \\ T_{3o}^{-1} \in RH_\infty \end{cases}$$

因此，由(5.39)式推導出 Nehari Problem

$$\begin{aligned} \|T_1 + \hat{Q}T_3\|_\infty &= \|T_1 + \hat{Q}T_{3o}T_{3i}\|_\infty \\ &= \|T_1T_{3i}^* + \hat{Q}T_{3o}\|_\infty \\ &= \|R - \tilde{Q}\|_\infty \end{aligned} \quad (5.41)$$

接下來依照[10]中的方式求解 Nehari Problem。

Step 3. 設計 Weighting Function :

由上述得知，經由求解 Nehari Problem，可求得 $\min_{Q \in RH_\infty} \|I - Q(I - M_n X_r)\|_\infty < \gamma$ ，

但對於一般系統而言，DOB 首要目標為抑制相對低頻之外來干擾，因此在[10]

中藉由加入一權重函數 $W(s)$ 來強調相對低頻域之干擾訊號，使得設計目標為

$$\min_{Q \in RH_\infty} \|W - Q(I - M_n X_r)W\|_\infty < \gamma \leq 1 \quad (5.42)$$

定義 $G_{d,v}(s) = (I - Q(I - M_n X_r)) = S_i$ ，則(5.42)式可推導成

$$\begin{aligned} \min_{Q \in RH_\infty} \|W - Q(I - M_n X_r)W\|_\infty &= \|S_i \cdot W\|_\infty < \gamma \leq 1 \\ &\Leftrightarrow |S_i(j\omega)| \leq |W^{-1}(j\omega)| \end{aligned} \quad (5.43)$$

然而權重函數的設計並非任意給定，當系統為非極小相位系統時，sensitivity function 必須滿足下面兩點：

1. 水床效應

當系統為非極小相位系統時，sensitivity function 必須滿足下面的積分式

$$\int_0^z \ln |S_i(j\omega)| d\omega = \int_0^z \ln |I - Q(j\omega)(I - M_n X_r(j\omega))| d\omega \approx 0 \quad (5.44)$$

(for $|S_i| \approx 1$ at $\omega > z$)

其中 z 為右半平面的零點，由(5.44)式可之，滿足此積分式則表示若低頻抑制能力越好，則高頻放大越多，就像水床一樣無法全部壓低。

2. 右半平面零點位置限制

右半平面之零點亦會限制干擾抑制頻寬。根據[10]所述，當系統存在右半平面實數零點 (Real RHP-Zero) 時，Sensitivity Function 約在 $\frac{z}{2}$ rad/sec 頻率處跨越 0 dB。當為虛數零點時約在

$$\begin{cases} |z|/4 & \text{Re}(z) \gg \text{Im}(z) \\ |z|/2.8 & \text{Re}(z) = \text{Im}(z) \\ |z| & \text{Re}(z) \ll \text{Im}(z) \end{cases} \text{ rad/sec 頻率處跨越 0 dB。}$$

因為上面兩個限制條件，參考[10]中提出的權重函數設計方法，我們可以使用 matlab 的設計工具加以輔助，以極零點拉繪出我們需要的權重函數，在重點處加強權重函數的成分，並且觀看響應是否達到我們需求，多測試不同的函數來進行調整，最後選出最符合需求的權重函數進行設計。

在[10]中對於整個設計的流程、步驟有更完整的推導以及說明，在此處我們引用幾個設計的方向、設計步驟、以及設計概念來進行說明，我們由上面步驟所設計出來的 $Q(s)$ 如下頁所示，皆是針對抑制 10 Hz 的干擾而設計，因此由圖 5.16~5.19 可以看出在 10 Hz 處都會有 notch 的效果。

$$Q_1(s) = \frac{-286.2s - 8518}{s + 10} \quad (5.46)$$

$$Q_2(s) = \frac{2738659.1839s(s + 9999)(s + 10)(s + 8)(s^2 + 108s + 7743)}{s(s + 10000)(s + 182.2)(s^2 + 316.9s + 39180)(s^2 + 113.2s + 37960)} \quad (5.47)$$

$$Q_3(s) = \frac{636260.3281(s + 10)(s + 8)(s^2 + 48.41s + 2852)(s^2 - 11.85s + 33900)}{(s + 99.7)(s^2 + 189.7s + 13230)(s^2 + 86.36s + 16530)(s^2 + 19.19s + 36350)} \quad (5.48)$$

$$Q_4(s) = \frac{215506.7175s(s + 165.9)(s + 10)(s + 8)(s^2 - 32.46s + 33380)}{(s + 103.7)(s^2 + 197.8s + 14290)(s^2 + 92.95s + 17660)(s^2 + 24.72s + 36910)} \quad (5.49)$$

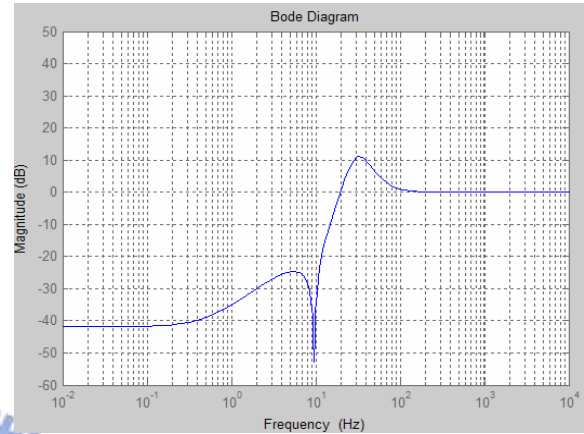
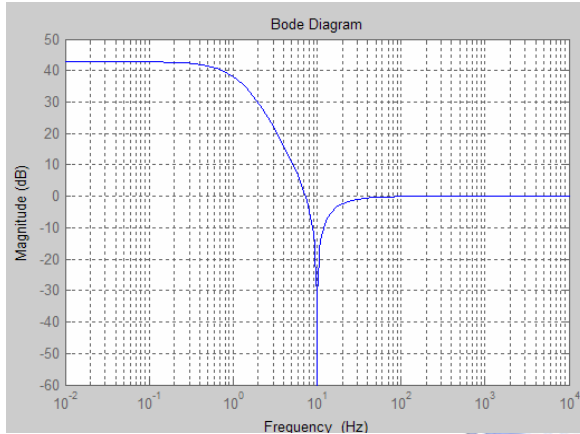


圖 5.16 設計為 $Q_1(s)$ 時 $G_{d_oV}(s)$ 頻率響應

圖 5.17 設計為 $Q_2(s)$ 時 $G_{d_oV}(s)$ 頻率響應

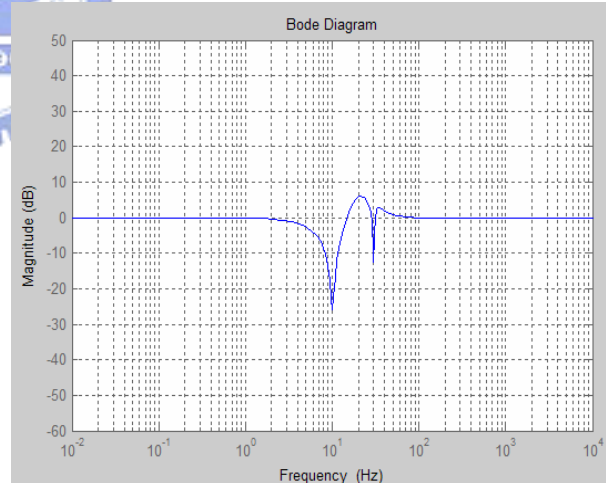
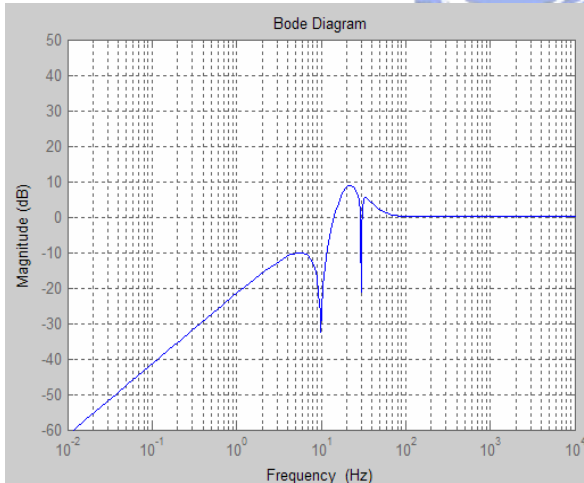


圖 5.18 設計為 $Q_3(s)$ 時 $G_{d_oV}(s)$ 頻率響應

圖 5.19 設計為 $Q_4(s)$ 時 $G_{d_oV}(s)$ 頻率響應

由上面討論，我們可以知道 DCF DOB 架構除了可以應付非極小相位系統以及不穩定系統外，還比傳統 DOB 的設計更有彈性、應用更廣泛，接下來我們將開始對模擬與實驗的結果進行比較討論。

第六章 實驗結果討論

前面提到移植 DSP 即時多工控制系統，但因硬體尚未通過測試，因此本文實驗部分仍採用 PC-bases 方式實現控制系統。本文的實驗過程中主要分為兩部分：第一部份為考慮輸入干擾的情況下系統的響應；第二部分為考慮輸入干擾以及人工加入輸出干擾的情況下系統的響應。其中將會比較傳統 PID 迴授控制以及 DOB 架構對於輸入干擾、輸出干擾的抑制能力，並且將提出實驗過程中遇到的問題以及討論。實驗中將使用三種控制架構來進行速度控制並比較其結果。

架構一：為單純 PI 速度迴授控制，如圖 6.1 所表示，以此架構作的控制效果作為基準，來比較加入 DOB 架構之後是否能讓系統擁有更佳的響應。

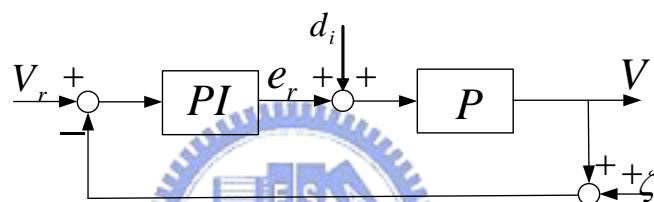


圖 6.1 PI 速度迴授控制

架構二：是在 PI 控制內迴路中加入傳統 DOB 架構，比較加入 DOB 架構之後對於干擾的排除能力是否提升，並希望在不確定性實驗時能夠逼近系統至期望系統，如圖 6.2。

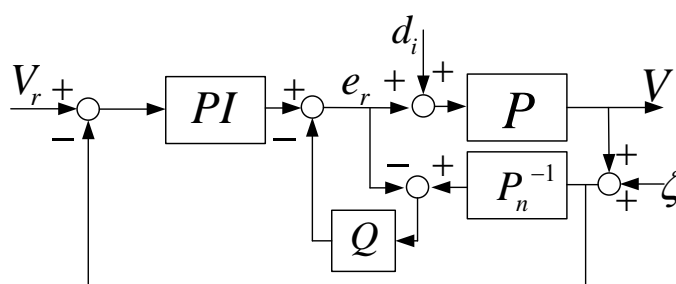


圖 6.2 傳統 DOB +PI 速度迴授控制

架構三：為 PI 控制加上 DCF DOB 架構，如下圖 6.3，由前面的敘述我們已經知道

在馬達系統中， Q 若設計成(5.9)式則其性能與傳統 DOB 相同，但在輸出干擾的實驗時設計成 Notch 形式則將呈現與傳統 DOB 不同的控制效果。

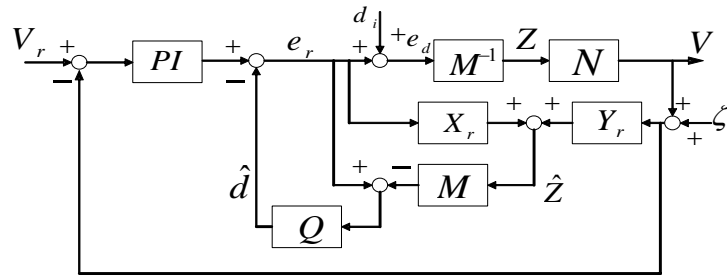


圖 6.3 DCF DOB+PI 速度迴授控制

6.1 考慮輸入干擾

在只有考慮輸入干擾的實驗中我們以上述之架構一及架構二進行實驗，我們將針對鈍齒力、摩擦力等輸入干擾比較不同的控制架構的控制效果，另外，本文將會在馬達系統中增加、減少質量來改變馬達系統的參數，以此比較不同架構對於系統不確定性的抵抗能力。

實驗一：開迴路定電壓速度響應

實驗一	開迴路
電壓命令	0.4 V

表6.1 實驗一相關資料

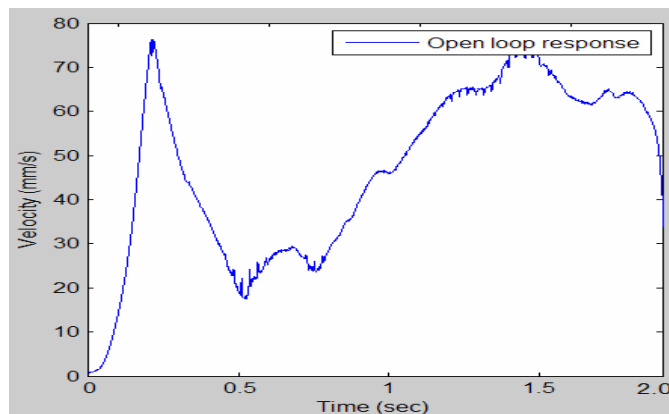


圖 6.4 定電壓開迴路速度響應

討論一：

實驗一所進行的是開迴路電壓輸入的測試，而經過多次測試後選出此組實驗資料，我們以 0.4V 電壓輸入進行開迴路測試，選擇 0.4V 的原因是因為線性馬達的行程太短，太大的電壓會使得平台太快走到極限，而太小的電壓又無法通過靜摩擦力的影響。由圖 6.4 我們可以知道，速度響應再通過靜摩擦力影響後，摩擦力隨著速度提升而減小，但又因為受到 cogging torque 以及其他輸入干擾的影響，導致無法維持等速前進。

6.1.1 速度追蹤實驗

我們將以兩種速度命令來進行實驗，第一種為等速速度追蹤實驗，第二種為正弦速度追蹤，實驗資料以及響應如下所示：

實驗二：無輸出干擾、等速速度追蹤響應

實驗二	架構一：純 PI 迴授控制	架構二：傳統 DOB+PI 迴授控制	
等速命令	$V_r = 20 \text{ mm/s}$	$V_r = 20 \text{ mm/s}$	
控制器參數	$PI \begin{cases} K_p = 100 \\ K_i = 2000 \end{cases}$	$PI \begin{cases} K_p = 100 \\ K_i = 2000 \end{cases}$ $DOB \{ Q = 10 \text{ Hz} \}$	$PI \begin{cases} K_p = 100 \\ K_i = 2000 \end{cases}$ $DOB \{ Q = 60 \text{ Hz} \}$
穩態均方根誤差	0.1195066279 mm/s (100%)	0.1153541891 mm/s (96.5253485%)	0.1254716357 mm/s (104.9913614 %)

表6.2 實驗二相關資料

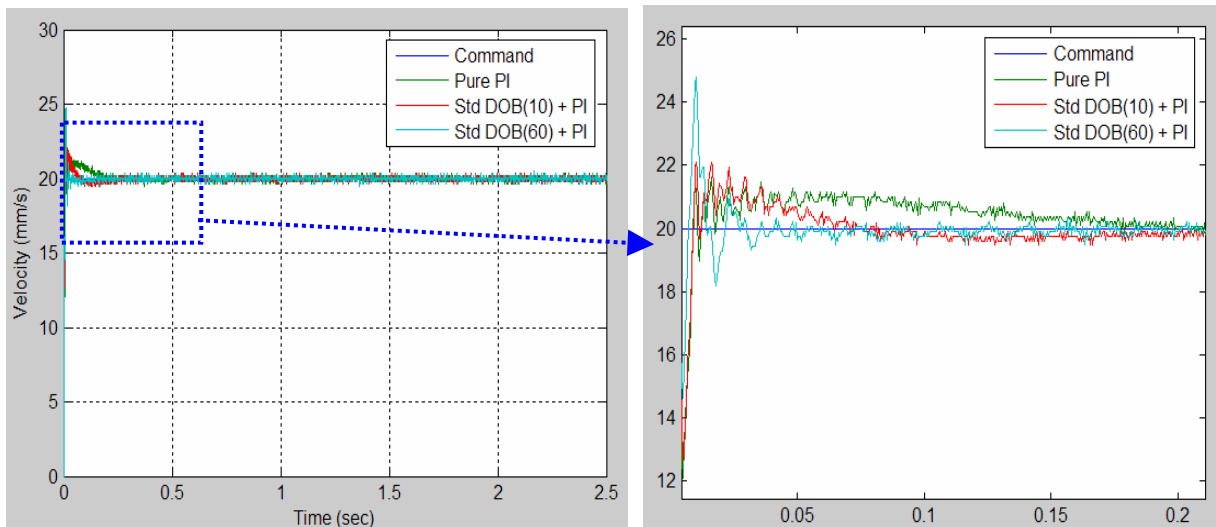


圖 6.5 等速實驗速度響應

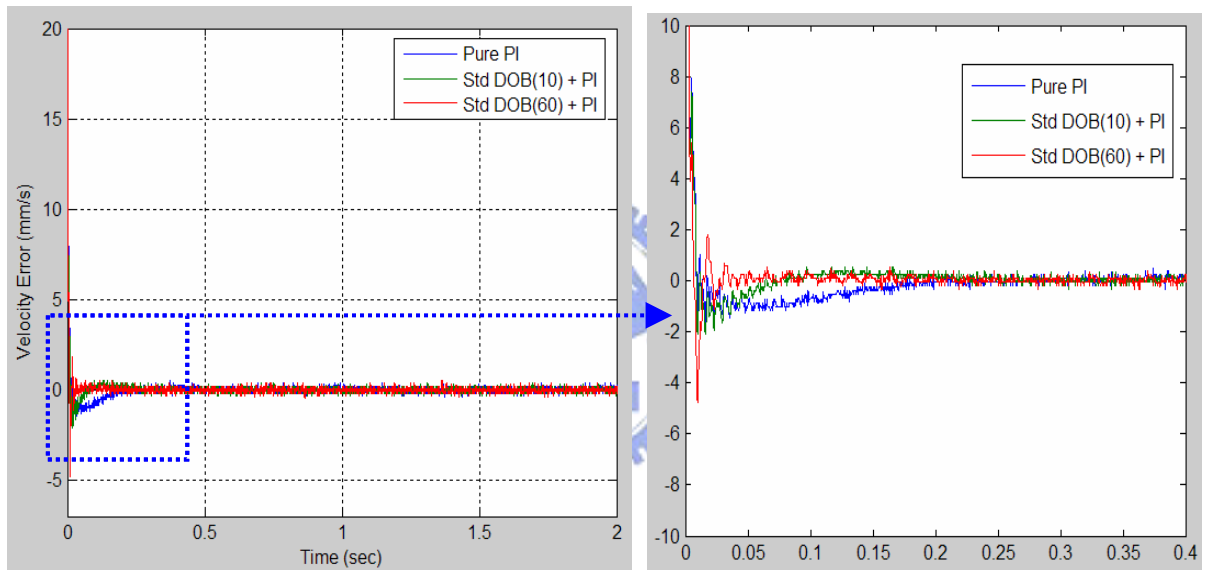


圖 6.6 速度響應誤差

討論二：

由圖 6.5、圖 6.6 可知道在馬達 SISO 系統的等速實驗之中，若能調整適當的 K_p 與 K_i 值，單純 PI 迴授控制架構也有不錯的響應。圖中我們可以看出加入 DOB 架構後，不同的 Q 頻寬會有不同的控制效果，值得討論的是， Q 頻寬為 10Hz 時與 60Hz 時，系統暫態響應明顯優於純 PI 控制(收斂到穩態速度快)，其收斂速度 DOB 60Hz 最快、10Hz 其次、純 PI 最慢，但穩態之後的方均根誤差 60Hz 的時候卻比 DOB 10Hz 以及純 PI 響應差，合理的推測可能是因為 Q 頻寬太大，雖然加快了系統的反應，但同時導致雜訊進入系統並造成誤差。

實驗三：無輸出干擾、Sin wave 速度追蹤響應

實驗三	架構一：純 PI 迴授控制	架構二：傳統 DOB+PI 迴授控制	
正弦命令	$V_r = 30 + 20\sin 5t \text{ mm/s}$	$V_r = 30 + 20\sin 5t \text{ mm/s}$	
控制器參數	$PI \begin{cases} K_p = 100 \\ K_i = 2000 \end{cases}$	$PI \begin{cases} K_p = 100 \\ K_i = 2000 \end{cases}$ $DOB \{ Q = 10 \text{ Hz} \}$	$PI \begin{cases} K_p = 100 \\ K_i = 2000 \end{cases}$ $DOB \{ Q = 60 \text{ Hz} \}$
穩態均方根誤差	0.1852405802 mm/s (100%)	0.1743031910 mm/s (94.0955760%)	0.1773648309 mm/s (95.7483671%)

表6.3 實驗三相關資料

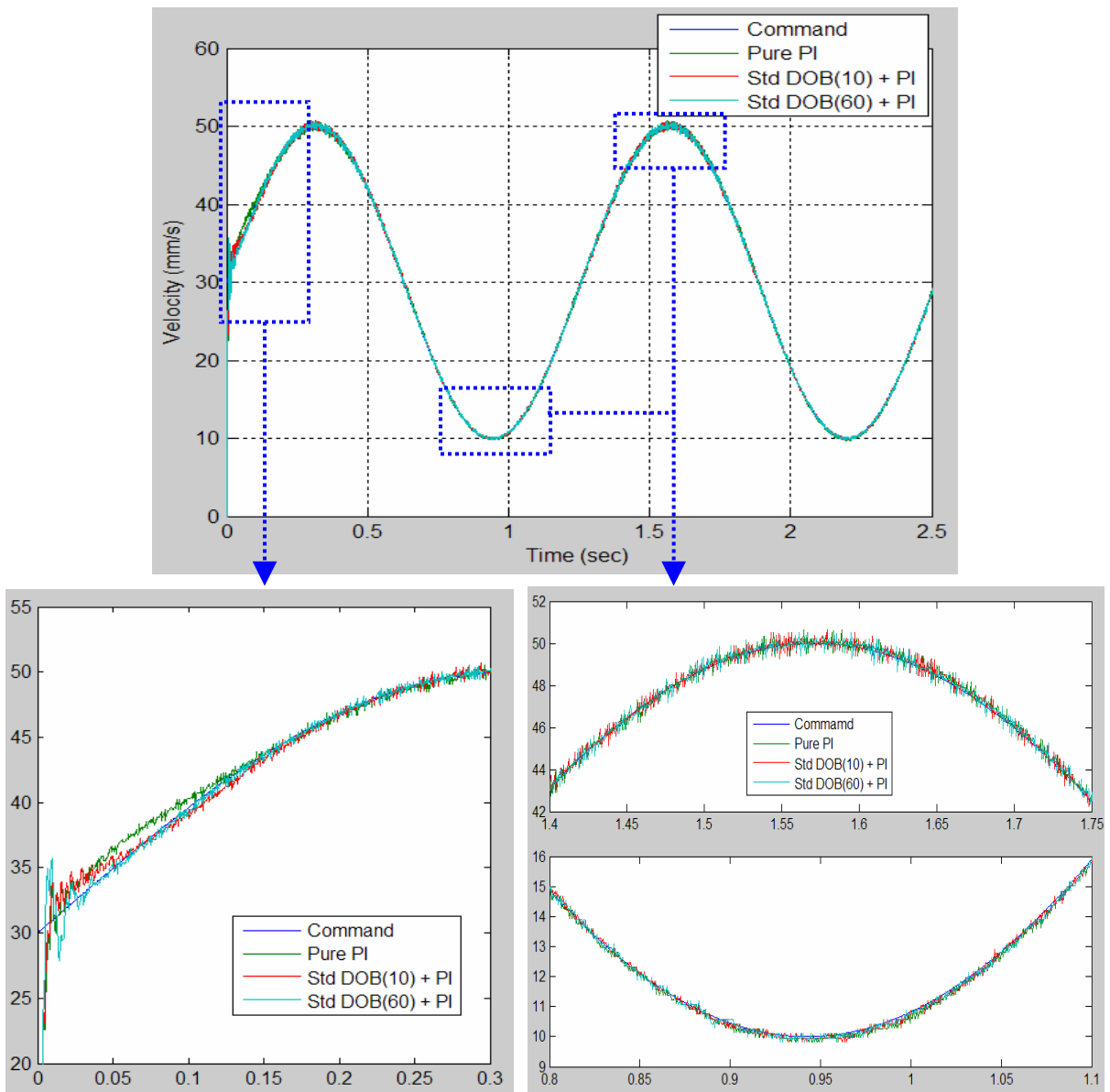


圖 6.7 正弦速度響應

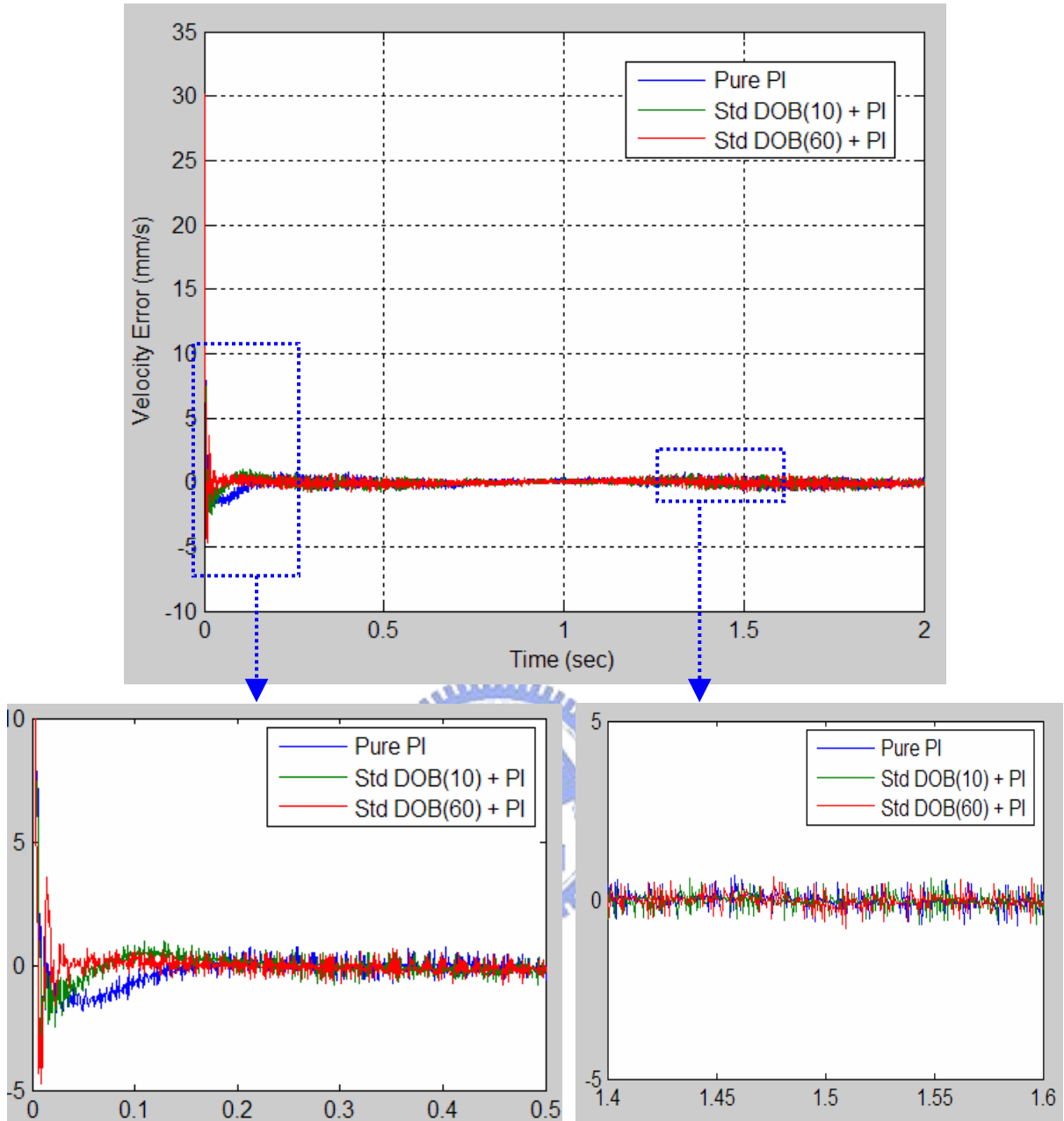


圖 6.8 正弦速度響應誤差

討論三：

由圖 6.7、圖 6.8 可知，若速度命令不再是簡單的等速命令，而是正弦速度命令時，加入 DOB 架構後對於系統的控制效果有較明顯的改善。此點我們可以從表 6.3 中穩態方均根誤差得知，加入 DOB 之後不但在暫態收斂的更快，在穩態時的誤差也較純 PI 控制架構更小，因此我們可以知道，當命令越複雜時，DOB 架構的效果也就越能顯現。但與前面實驗相同，DOB 頻寬 60Hz 的響應並沒有 10 Hz 理想，應該是頻寬太大造成。

6.1.2 改變不確定性實驗

將 1.5 Kg 的質量塊加裝到馬達系統中，作為馬達系統慣量 J_n 的不確定性，進行加質量的正弦速度追蹤實驗(實驗四)，並且重新鑑定此系統為 nominal plant，重新設計 PI 後再將質量塊移除進行減質量的正弦速度追蹤實驗(實驗五)。並且由前面實驗我們知道 DOB 頻寬 60 Hz 有過大的嫌疑，所以此部分實驗結果只呈現 DOB 10 Hz 的結果。

實驗四：無輸出干擾、加質量、Sin wave 速度追蹤響應

實驗四	架構一：純 PI 迴授控制	架構二：傳統 DOB+PI 迴授控制
系統參數	$\begin{cases} J_n = 2.92733 \\ B_n = 15.4386 \end{cases}$	$\begin{cases} J_n = 2.92733 \\ B_n = 15.4386 \end{cases}$
動作	+1.5 Kg	+1.5 Kg
正弦命令	$V_r = 30+20\sin 5t \text{ mm/s}$	$V_r = 30+20\sin 5t \text{ mm/s}$
控制器參數	$PI \begin{cases} K_p = 100 \\ K_i = 2000 \end{cases}$	$PI \begin{cases} K_p = 100 \\ K_i = 2000 \end{cases}$ $DOB \{ Q = 10 \text{ Hz} \}$
加質量前 穩態均方根誤差	0.185240580296 mm/s (100%)	0.174303191057 mm/s (94.0955760%)
加質量後 穩態均方根誤差	0.192766387032 mm/s (100%)	0.169597335085 mm/s (87.9807%)

表6.4 實驗四相關資料

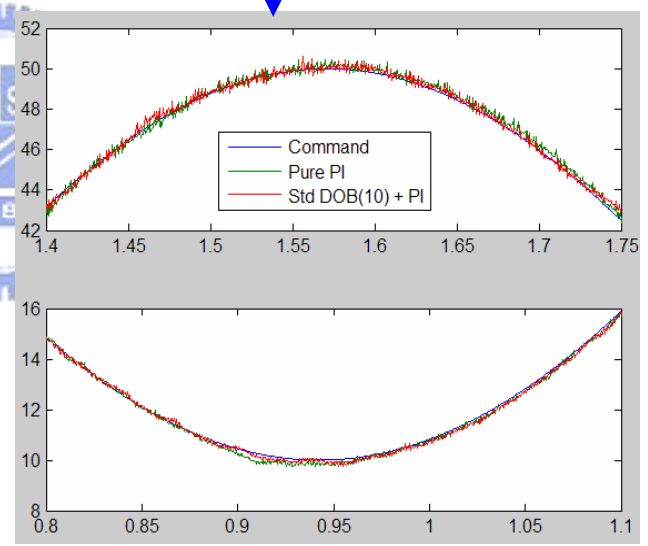
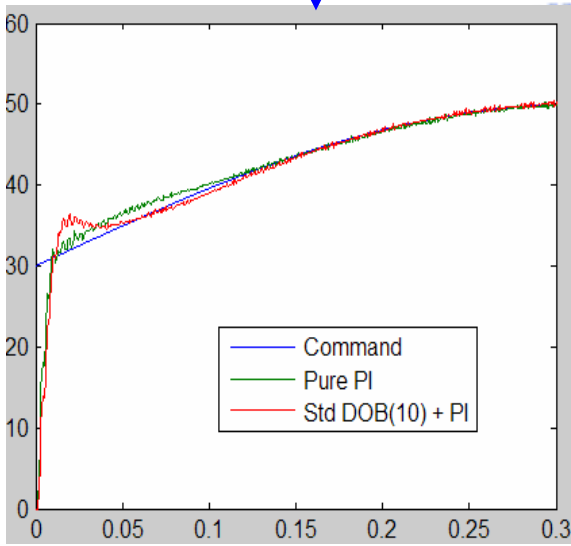
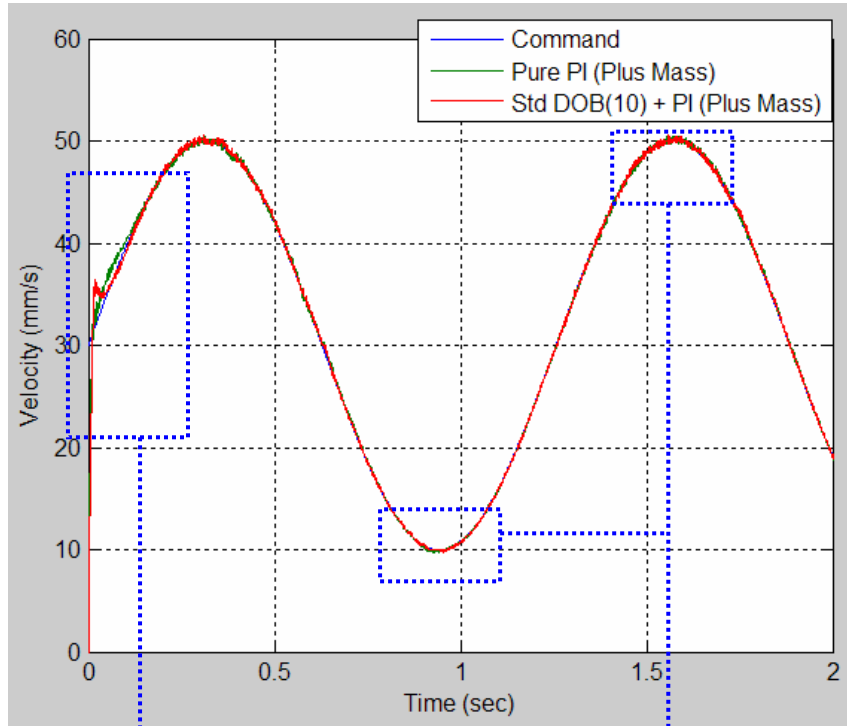


圖 6.9 加質量後正弦速度響應

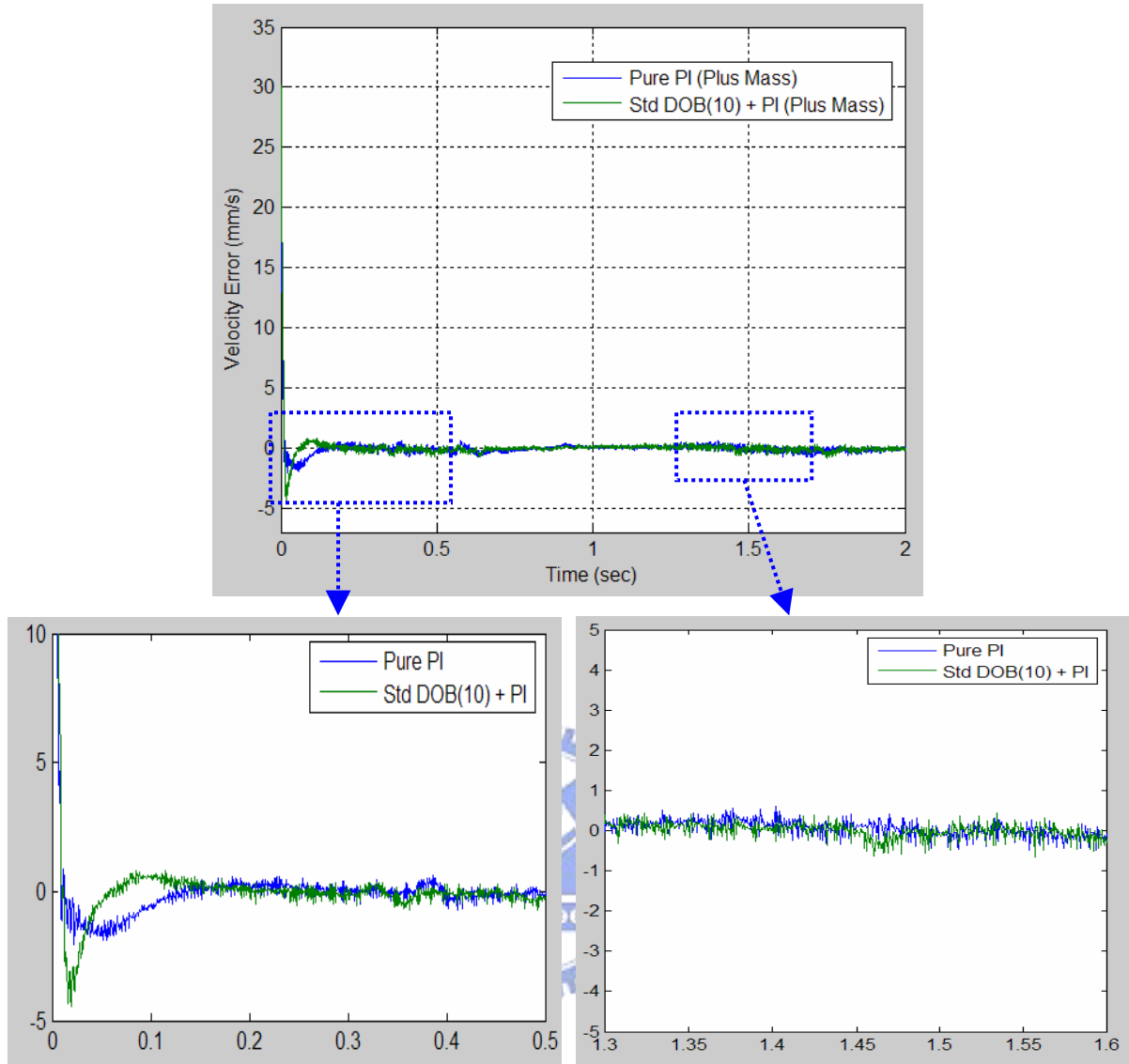


圖 6.10 加質量後正弦速度響應誤差

討論四：

加上 1.5 Kg 的質量塊當作是系統慣量參數的不確定性，由相同的控制器參數做控制，藉此比較兩種架構對於不確定性的抵抗能力。由表 6.4 我們知道加入質量塊之後，兩架構的實驗誤差 RMSE 分別增加了 4.06272%以及降低了 8.445%，可見得加了不確定性後，純 PI 受影響而變差而 DOB 將有更好的效果。由圖 6.9、圖 6.10 可知如我們所預期，加入 DOB 架構的實驗架構擁有較好的響應，若以加入質量塊之後兩架構的響應相比，加入 DOB 架構的 RMSE 比純 PI 的 RMSE 降低 12.01923%，可知 DOB 架構確實能提升抵抗不確定性的能力。

實驗五：無輸出干擾、減質量、Sin wave 速度追蹤響應

實驗五	架構一：純 PI 迴授控制	架構二：傳統 DOB+PI 迴授控制
系統參數	$\begin{cases} J_n = 4.1450 \\ B_n = 22.4026 \end{cases}$	$\begin{cases} J_n = 4.1450 \\ B_n = 22.4026 \end{cases}$
動作	-1.5kg	-1.5kg
正弦命令	$V_r = 30+20\sin 5t \text{ mm/s}$	$V_r = 30+20\sin 5t \text{ mm/s}$
控制器參數	$PI \begin{cases} K_p = 120 \\ K_i = 3000 \end{cases}$	$PI \begin{cases} K_p = 120 \\ K_i = 3000 \end{cases}$ $DOB \{ Q = 10 \text{ Hz} \}$
減質量前 穩態均方根誤差	0.135764768434 mm/s (100%)	0.133127676661 mm/s (98.05760227 %)
減質量後 穩態均方根誤差	0.141750654893 mm/s (100%)	0.139750050013 mm/s (98.5886%)

表6.5 實驗五相關資料

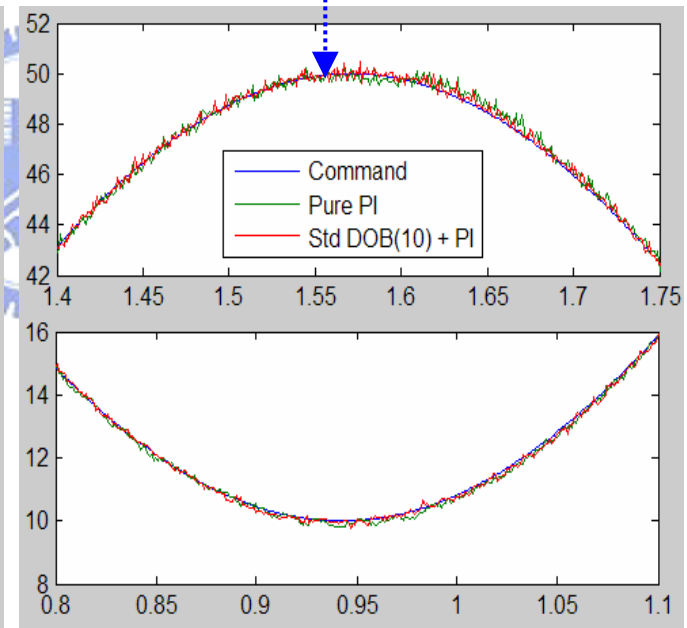
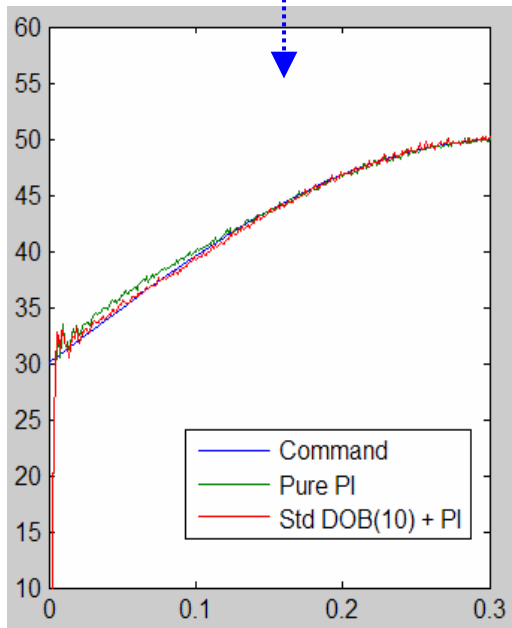
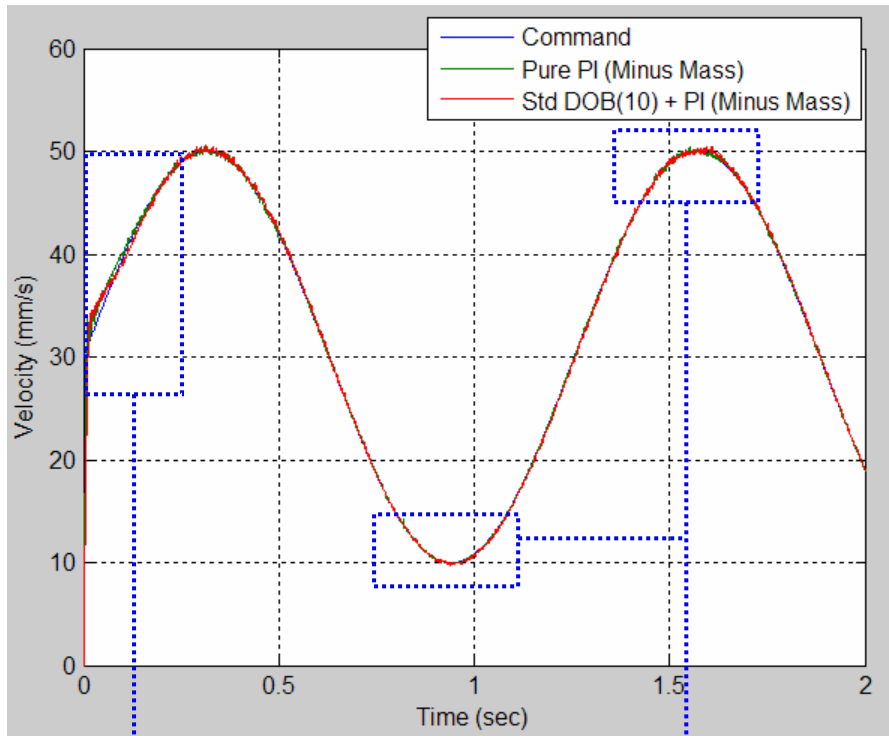


圖 6.11 減質量後正弦速度響應

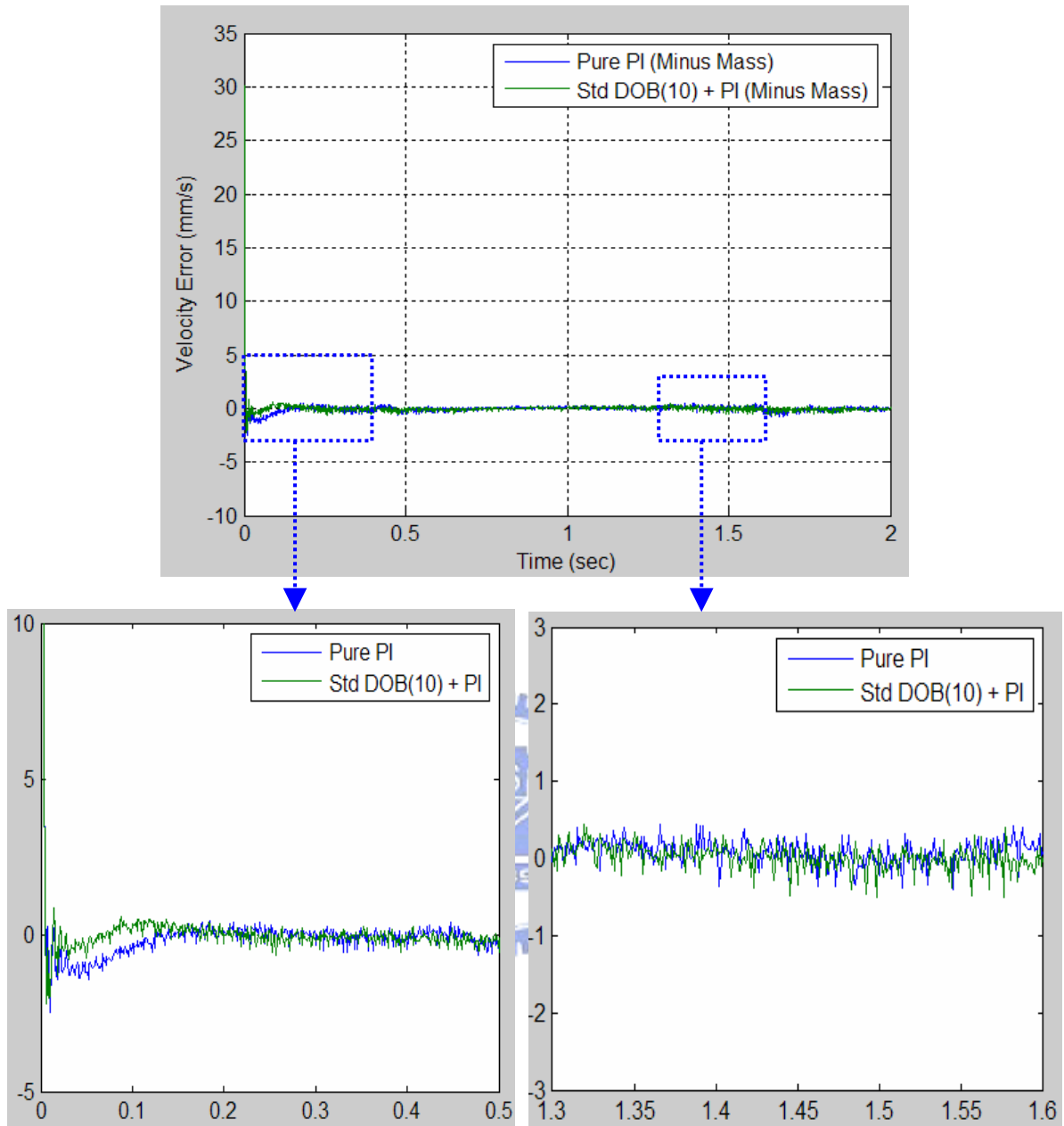


圖 6.12 減質量後正弦速度響應誤差

討論五：

進行此部分實驗實驗之前，須要將加了質量的系統作為 *nimnal plant*，經由系統鑑別鑑別出新系統的參數，如表 6.5， $J_n = 4.1450$ 、 $B_n = 22.4026$ ，以系統調整 PI 控制器參數得到 $K_p = 120$ 、 $K_i = 3000$ ，並且進行正弦速度追蹤實驗作為比較基準，兩架構在減去質量之前的均方根誤差 RMSE 相差 1.943%。在減去質量塊之後，實驗結果如圖 6.11、

6.12 所表示，雖然兩架構的 RMSE 都有增加，純 PI 增加了 4.409% 而加入 DOB 架構後增加了 2.935%，但我們仍看得出加入 DOB 架構後之響應如前面幾個實驗相同，在暫態方面表現較為優良，而在穩態方均根誤差也略勝一籌(1.41135%)，因此我們能肯定在內迴路加入 DOB 架構可以增加系統抵抗干擾、雜訊、不確定性的性能。

經由前面幾個實驗，我們可以知道在馬達控制系統中加入傳統 DOB 的架構，不但不影響外迴路 PI 的設計，並且能夠提升整體控制的性能，還可以提升系統對於不確定性的抵抗能力，最重要的是傳統 DOB 的架構簡單，對於數位實現而言是一大優點。接下來，我們將會討論加入輸出干擾之後，這些架構的抵抗能力。



6.2 考慮輸入干擾、輸出干擾

加入了輸出干擾之後，我們將採用前面所說的三種架構，分別是純 PI 迴授控制、傳統 DOB 配合 PI 迴授控制、DCF DOB 配合 PI 迴授控制架構來進行實驗，並且比較結果提出討論。而我們加入的人工輸出干擾來自於裝了 laser encoder 鏡頭的線性陶瓷壓電馬達，如圖 2.2 所示，透過簡單的 PI 閉迴路控制器來控制壓電馬達的響應，藉以作為系統的輸出干擾。

此部分將進行的實驗，主要為幾種不同的輸出干擾對正弦速度響應的影響，包括 Step、Ramp 形式的輸出干擾，不同頻率之 Sin Wave 來回震盪的輸出干擾，並且在最後提出設計成 Notch 形式的 DCF DOB 來排除特定頻率干擾，以表現出 DCF DOB 更為彈性的設計及寬廣的應用範圍。

實驗六：Step 輸出干擾、Sin wave 速度追蹤響應

實驗六	架構一：純 PI 迴授控制	架構二：傳統 DOB+PI 迴授控制
正弦速度命令	$V_r = 30+20\sin 5t \text{ mm/s}$	$V_r = 30+20\sin 5t \text{ mm/s}$
Step 輸出干擾 1	$d_o = 5 \text{ mm/s}$	$d_o = 5 \text{ mm/s}$
Step 輸出干擾 2	$d_o = 10 \text{ mm/s}$	$d_o = 10 \text{ mm/s}$
控制器參數	$PI \begin{cases} K_p = 100 \\ K_i = 2000 \end{cases}$	$PI \begin{cases} K_p = 100 \\ K_i = 2000 \end{cases}$ $DOB \{ Q = 10 \text{ Hz} \}$
未加 DO 之前 穩態均方根誤差	0.185240580296 mm/s (100%)	0.174303191057 mm/s (94.0955760%)
加入 DO1 之後 穩態均方根誤差	0.245156135552 mm/s (100%)	0.2309875687758 mm/s (94.2205%)
加入 DO2 之後 穩態均方根誤差	0.261724118043 mm/s (100%)	0.2249019555312 mm/s (85.9309%)

表6.6 實驗六相關資料

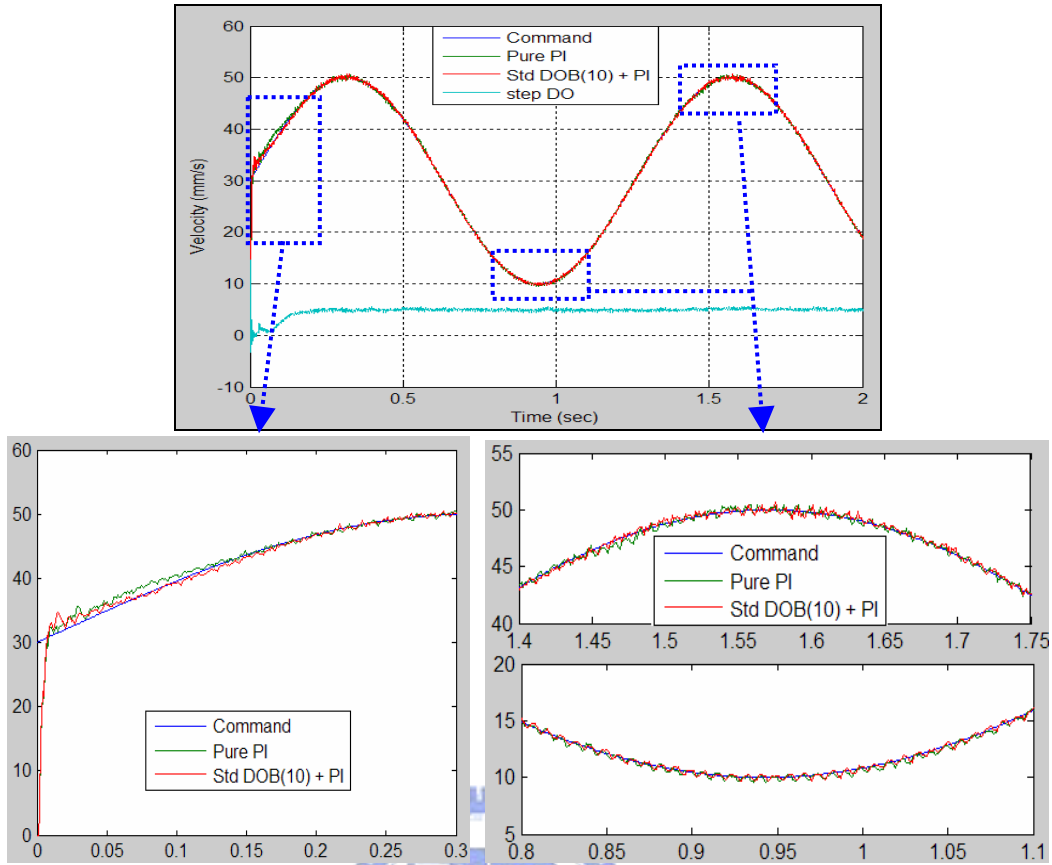


圖 6.13 Step DO(1)與正弦速度響應

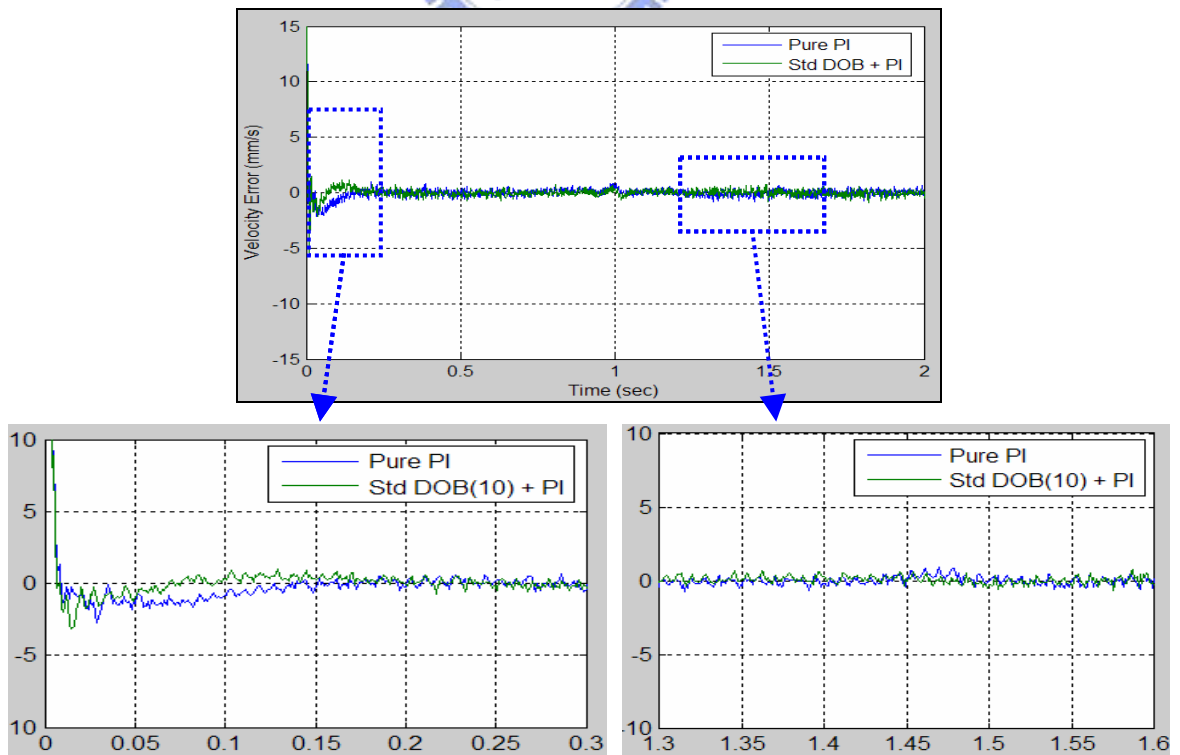


圖 6.14 Step DO(1)之正弦速度響應誤差

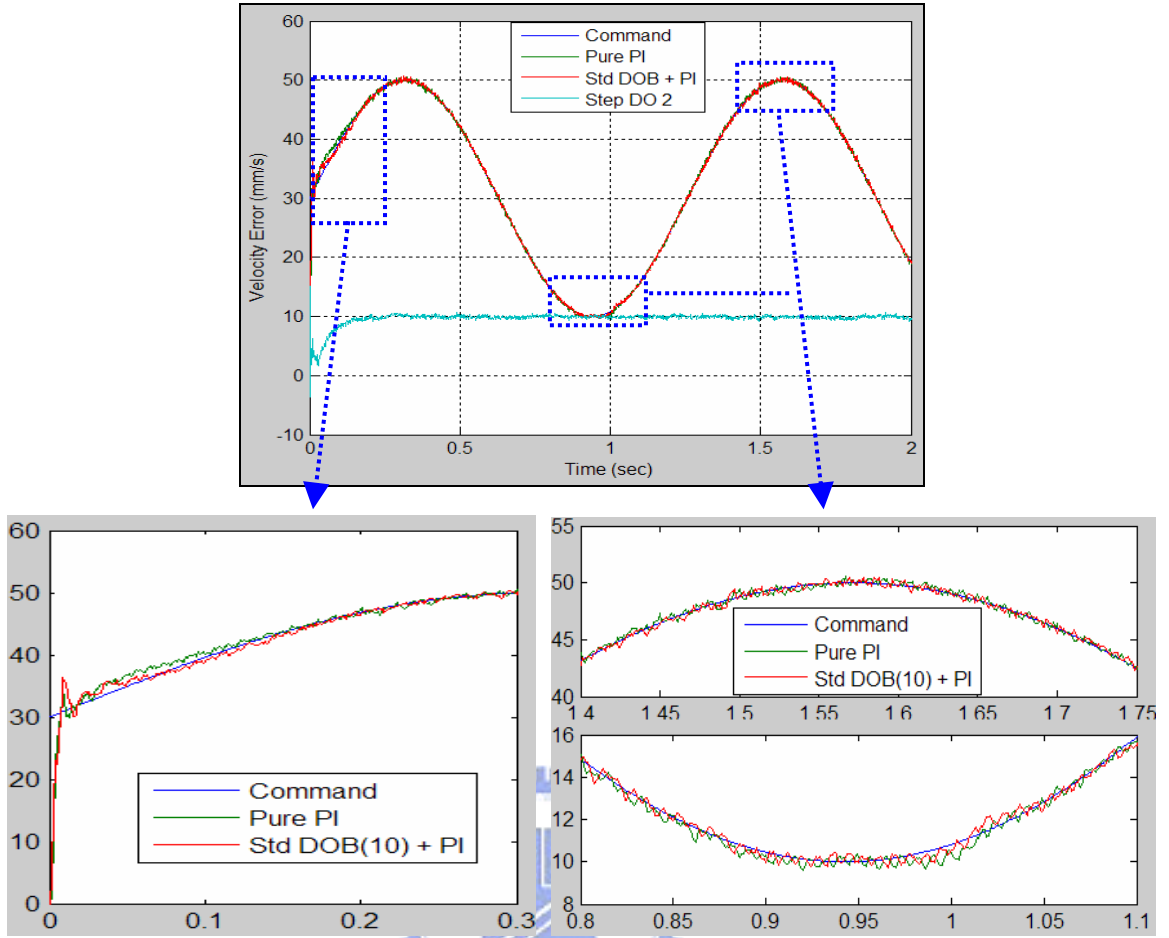


圖 6.15 Step DO(2)與正弦速度響應

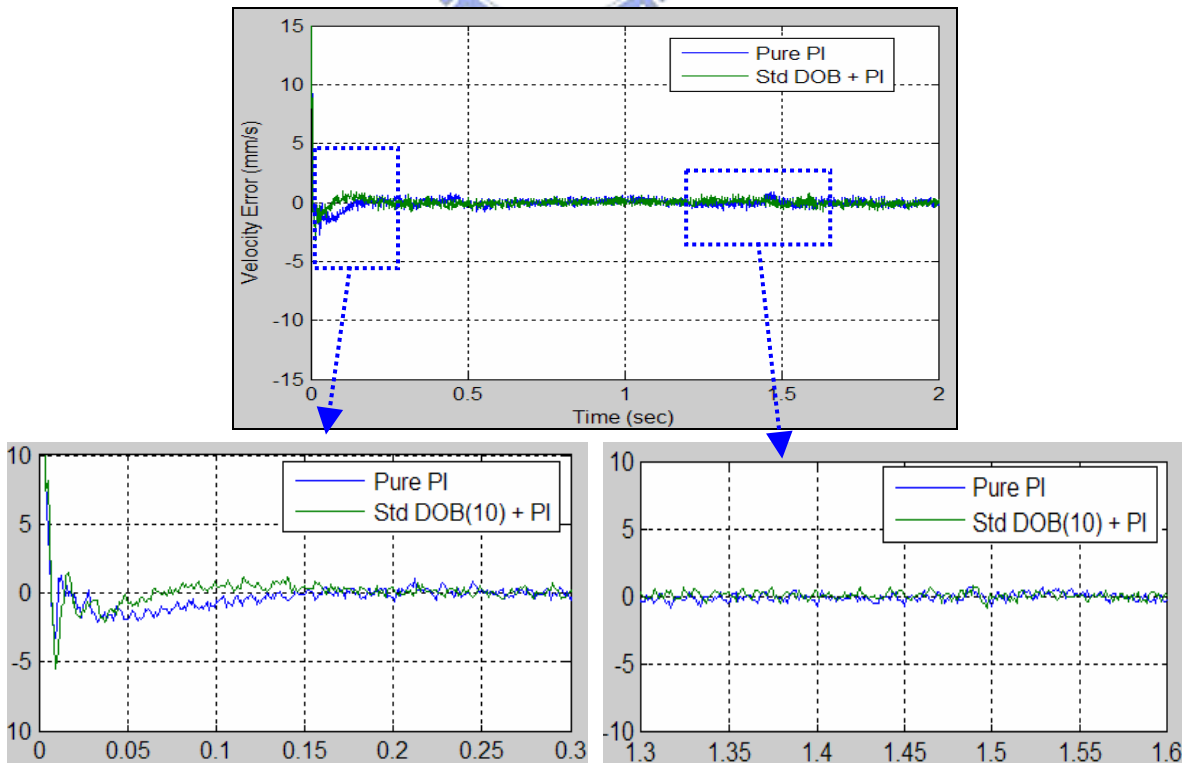


圖 6.16 Step DO(2)之正弦速度響應誤差

討論六：

在此組實驗中，我們主要測試輸出干擾對於系統的影響，因此使用小馬達製造出簡單的 Step disturbance，DO1 為大小 5 mm/s 的步階速度，其實際響應可從圖 6.13 中觀察到，而 DO2 為大小 10 mm/s 之步階速度，如圖 6.15 中所表示。由實驗結果可以知道，當系統受到步階輸出干擾時，系統響應因受到干擾而變差，受到越大的干擾準確度也就越差，從表 6.6 我們知道當加入 5 mm/s 的 DO 時，DOB 組之穩態 RMSE 為純 PI 組之 94.2205%，改善了 5.7795% 的穩態響應，而當加入 10 mm/s 的 DO 時，DOB 組的 RMSE 為 PI 組的 85.9309%，改善了 14.0691%，因此，加入 DOB 的系統還是較純 PI 迴授擁有較好的響應，不但暫態時的收斂速度較快，穩態誤差也較小。

實驗七：Ramp 輸出干擾、Sin wave 速度追蹤響應

實驗七	架構一：純 PI 迴授控制	架構二：傳統 DOB+PI 迴授控制
正弦速度命令	$V_r = 30 + 20\sin 5t \text{ mm/s}$	$V_r = 30 + 20\sin 5t \text{ mm/s}$
Ramp 輸出干擾 1	$d_o = 3 \text{ mm/s}^2$	$d_o = 3 \text{ mm/s}^2$
Ramp 輸出干擾 2	$d_o = 5 \text{ mm/s}^2$	$d_o = 5 \text{ mm/s}^2$
控制器參數	$PI \begin{cases} K_p = 100 \\ K_i = 2000 \end{cases}$	$PI \begin{cases} K_p = 100 \\ K_i = 2000 \end{cases}$ $DOB \{ Q = 10 \text{ Hz} \}$
未加 DO 之前 穩態均方根誤差	0.185240580296 mm/s (100%)	0.174303191057 mm/s (94.0955760%)
加入 DO1 之後 穩態均方根誤差	0.248177351028 mm/s (100%)	0.228188426857 mm/s (91.9457%)
加入 DO2 之後 穩態均方根誤差	0.254008117270 mm/s (100%)	0.241499695762 mm/s (95.0755%)

表6.7 實驗七相關資料

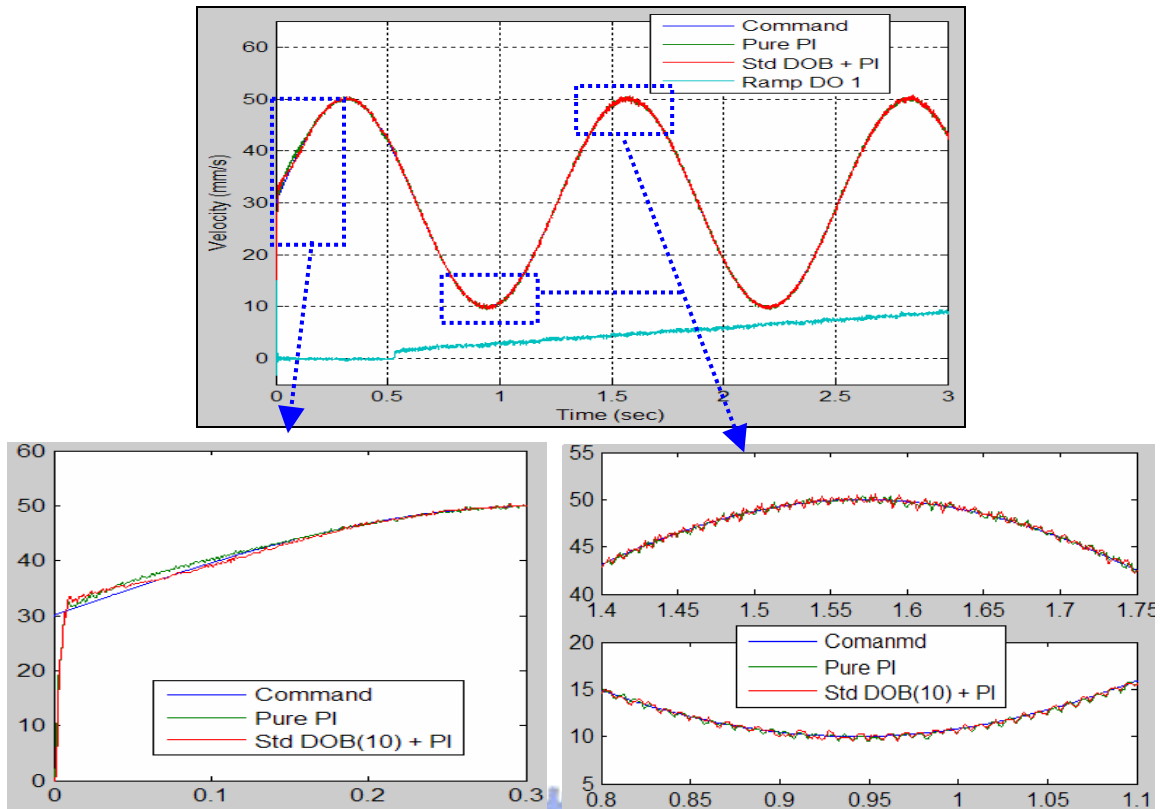


圖 6.17 Ramp DO(1)與正弦速度響應

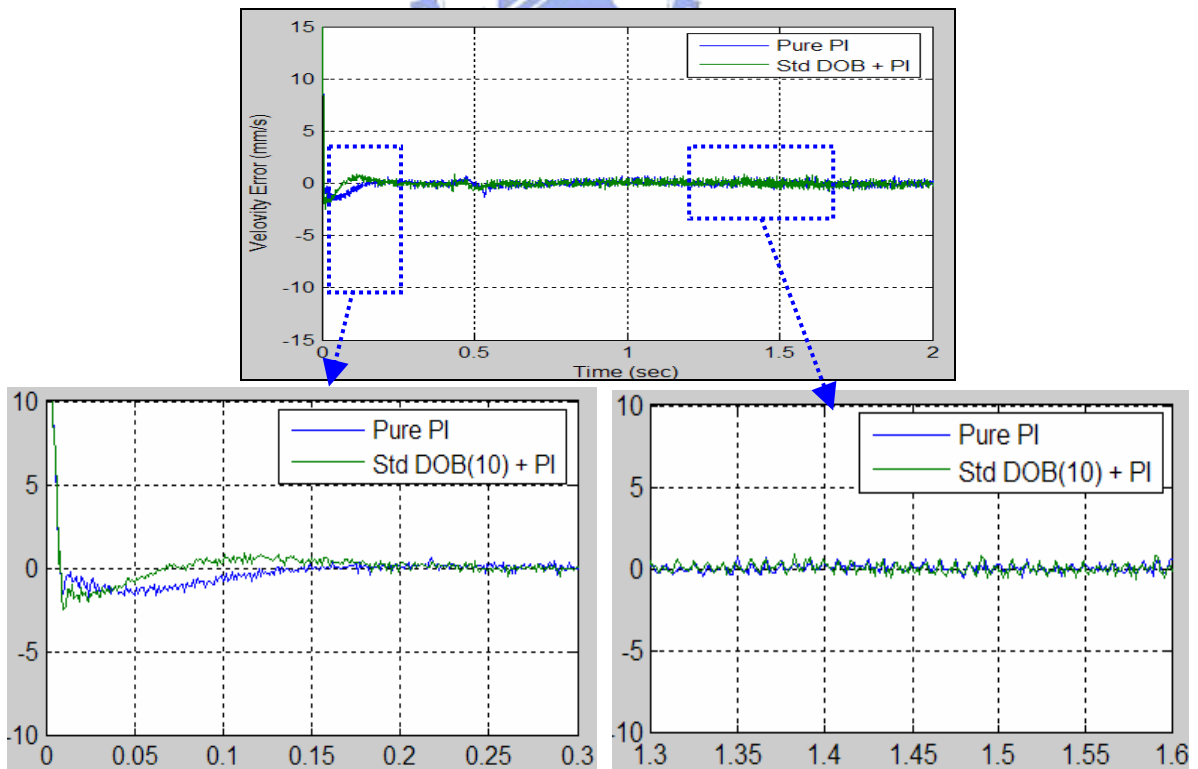


圖 6.18 Ramp DO(1)之正弦速度響應誤差

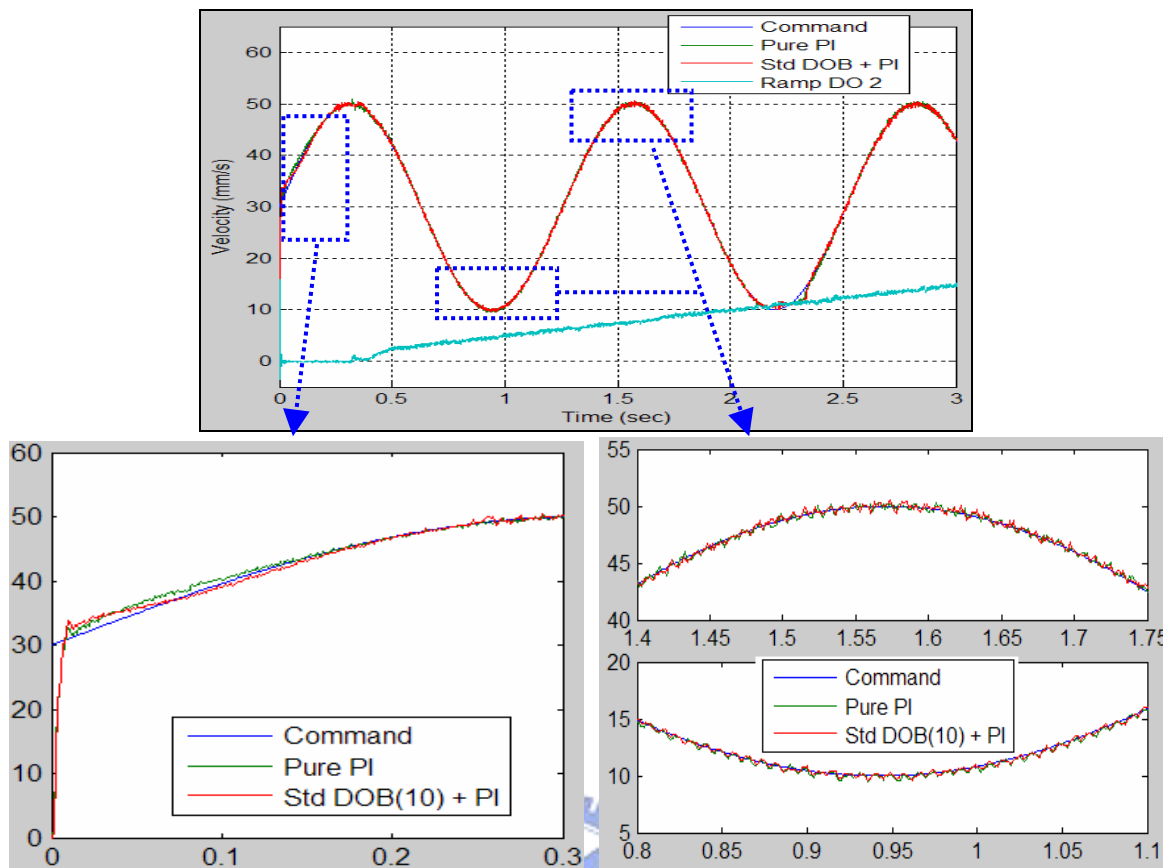


圖 6.19 Step DO(2)與正弦速度響應

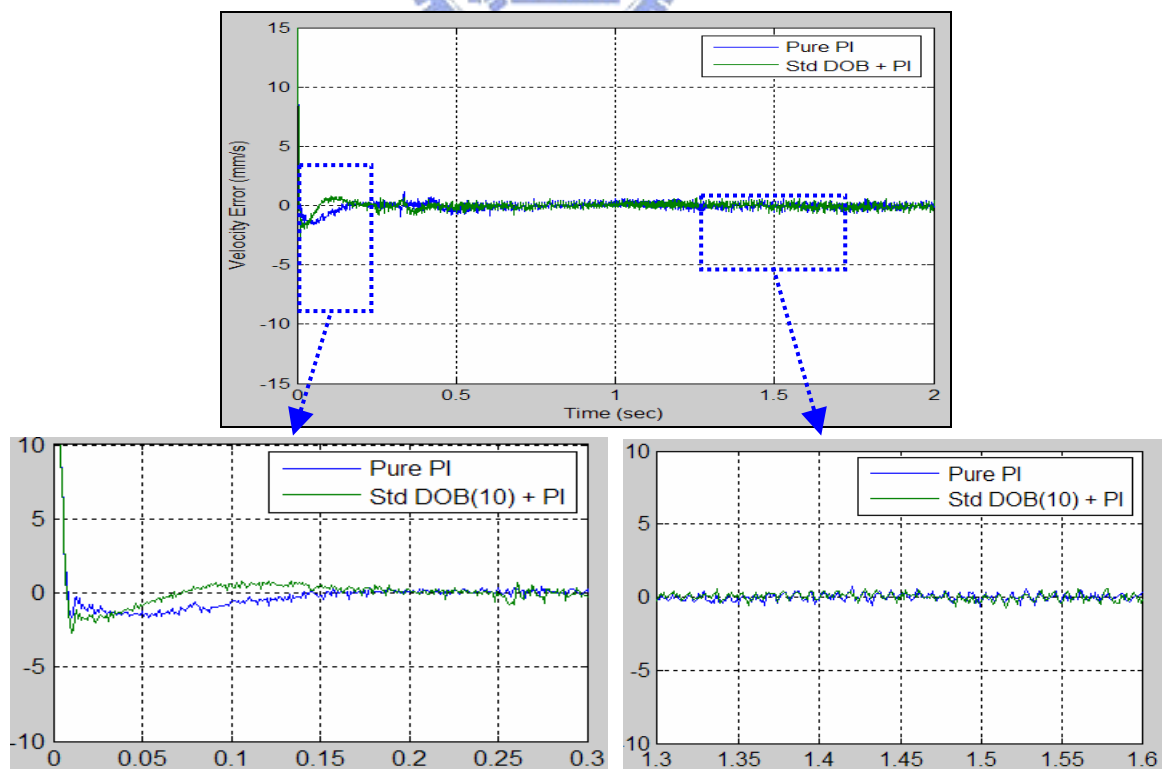


圖 6.20 Step DO(2)之速度響應誤差

討論七：

加入了 Ramp DO 之後，會遇到啟動時摩擦力太大而導致延遲的情況，此點我們可從圖 6.17、6.19 中 Ramp DO 線段看出。由實驗結果可知，加入 Ramp DO 的大小會直接影響系統響應的好壞，此外，加入 DOB 架構的系統與純 PI 控制系統相比，在兩種不同大小的 Ramp DO 下其 RMSE 分別為純 PI 系統的 91.9457%與 95.0755%，因此我們再次驗證加入 DOB 架構之後，能夠提升系統抵抗干擾的能力。

實驗八：Sin wave 輸出干擾、加質量塊、Sin wave 速度追蹤響應

實驗八	架構一：純 PI 迴授控制	架構二：傳統 DOB+PI 迴授控制
正弦速度命令	$V_r = 30 + 20\sin 5t \text{ mm/s}$	$V_r = 30 + 20\sin 5t \text{ mm/s}$
加質量塊	+1.5 Kg	+1.5 Kg
Sin wave 輸出干擾	$d_o = 10\sin(5t) \text{ mm/s}$	$d_o = 10\sin(5t) \text{ mm/s}$
控制器參數	$PI \begin{cases} K_p = 100 \\ K_i = 2000 \end{cases}$	$PI \begin{cases} K_p = 100 \\ K_i = 2000 \end{cases}$ $DOB \{ Q = 10 \text{ Hz} \}$
加 Do 及 Mass 之前 穩態均方根誤差	0.185240580296 mm/s (100%)	0.174303191057 mm/s (94.0955760%)
加 DO 及 Mass 之後 穩態均方根誤差	0.480303733197 mm/s (100%)	0.440864640942 mm/s (91.7887183%)

表6.8 實驗八相關資料

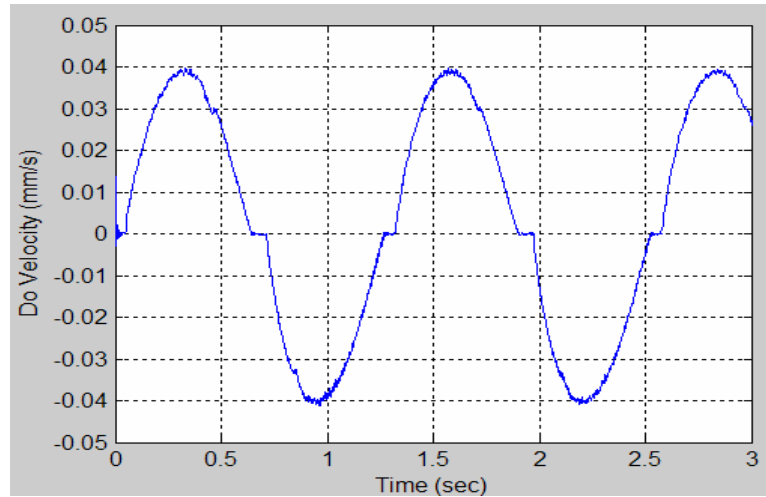


圖 6.21 人工 DO 響應

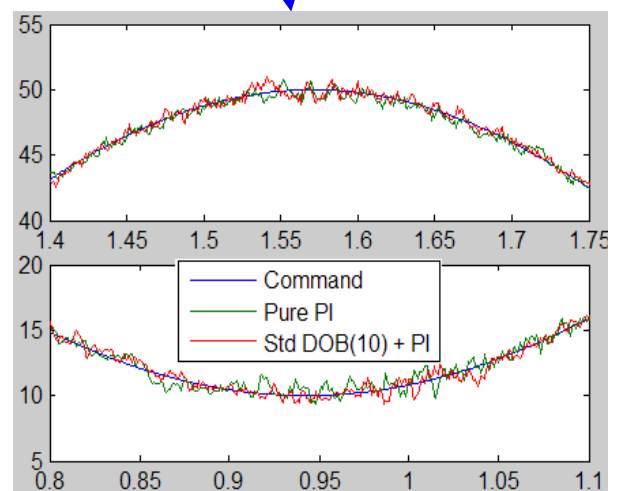
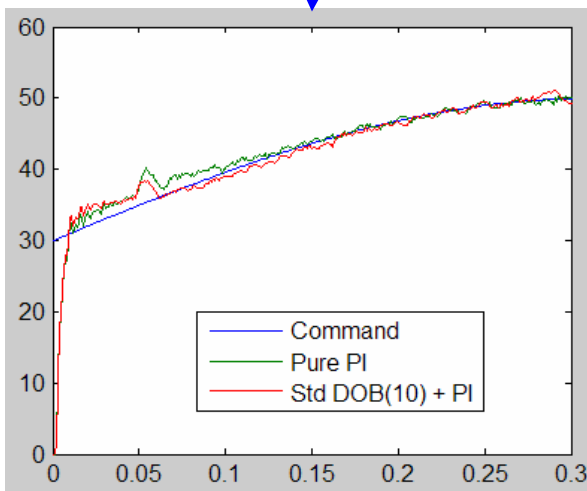
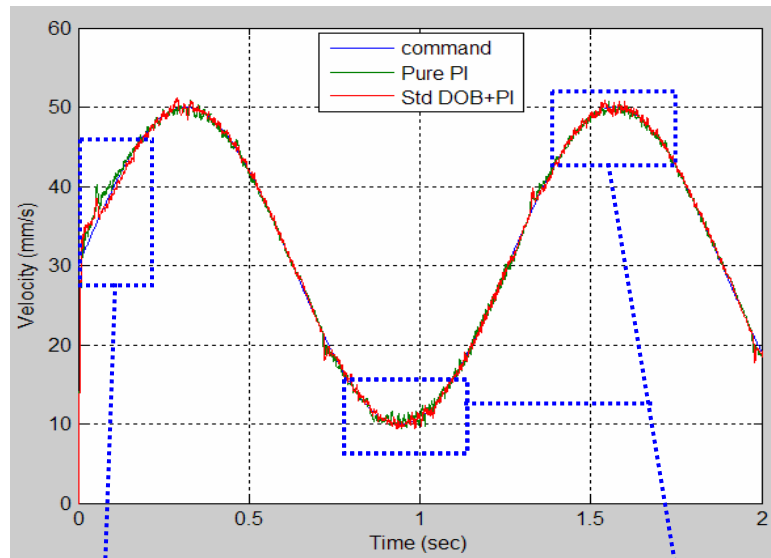


圖 6.22 實驗八結果響應

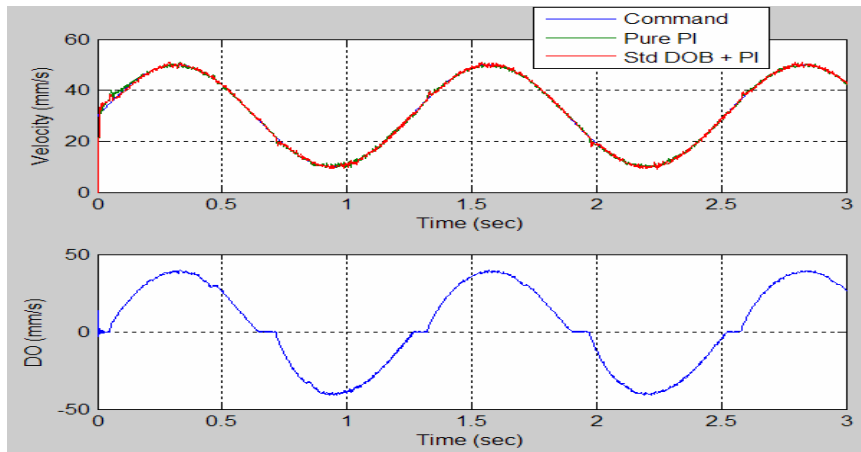


圖 6.23 實驗八結果與 DO 之相對關係

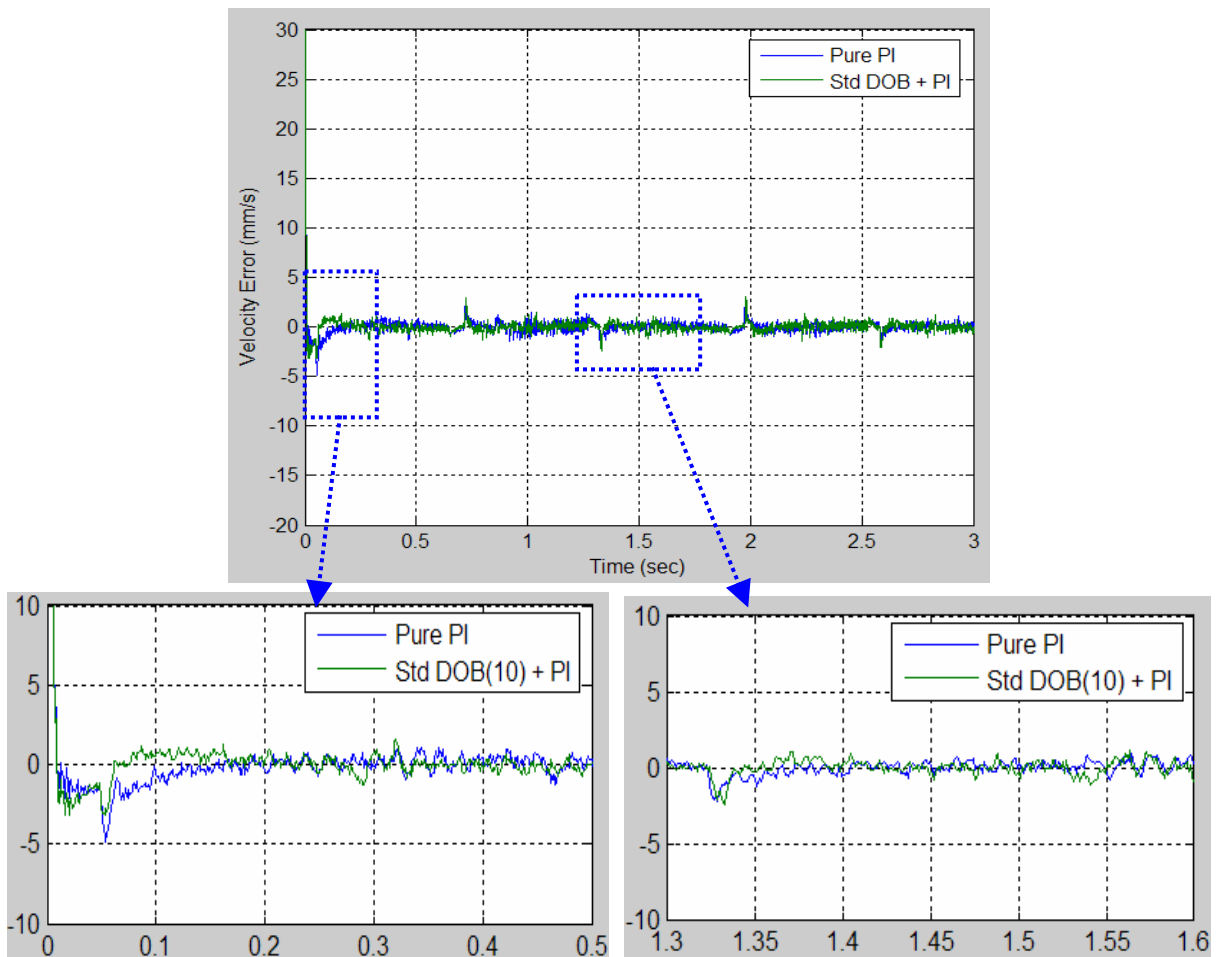


圖 6.24 實驗八之速度誤差

討論八：

在此實驗中，我們將加入人工週期輸出干擾(0.791 Hz)，如圖 6.21 所示，我們知道因為此馬達有較大的遲滯效應，所以 DO 產生的並不是很順暢，在低速時會因為摩擦力的影響而停止。此外，我們增加不確定性的影響(加入質量塊)，結果如圖 6.22、6.23 所示，我們可以看出在 DO 從停止啟動時，系統受到的影響較大，之後就能追上速度命令。而在加入 DO 以及不確定的情況下，兩架構的 RMSE 均比前面實驗提高不少，表示在此情況之下兩架構的控制已稍顯吃力。

實驗九：Sin wave 輸出干擾、減質量塊、Sin wave 速度追蹤響應

實驗九	架構一：純 PI 迴授控制	架構二：傳統 DOB+PI 迴授控制
正弦速度命令	$V_r = 30 + 20 \sin 5t \text{ mm/s}$	$V_r = 30 + 20 \sin 5t \text{ mm/s}$
減質量塊	-1.5 Kg	-1.5 Kg
Sin wave 輸出干擾	$d_o = 10 \sin(5t) \text{ mm/s}$	$d_o = 10 \sin(5t) \text{ mm/s}$
控制器參數	$PI \begin{cases} K_p = 120 \\ K_i = 3000 \end{cases}$	$PI \begin{cases} K_p = 120 \\ K_i = 3000 \end{cases}$ $DOB \{ Q = 10 \text{ Hz} \}$
加 Do 及 Mass 之前 穩態均方根誤差	0.185240580296 mm/s (100%)	0.174303191057 mm/s (94.0955760%)
加 DO 及 Mass 之後 穩態均方根誤差	0.437963858185 mm/s (100%)	0.444087639238 mm/s (101.3982389%)

表6.9 實驗九相關資料

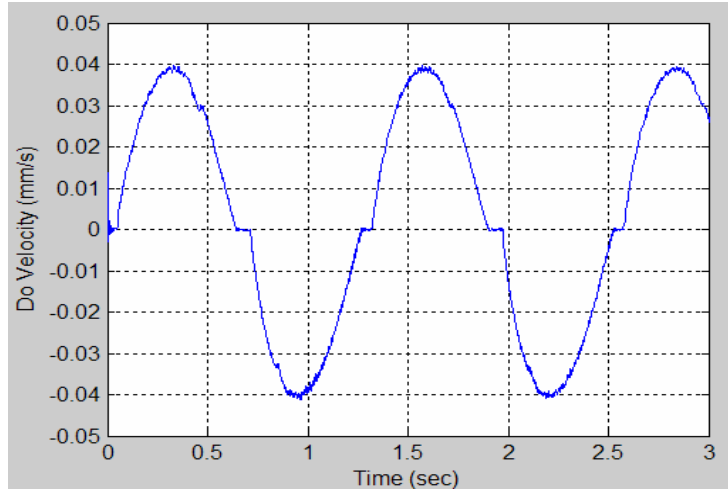


圖 6.25 人工 DO 響應

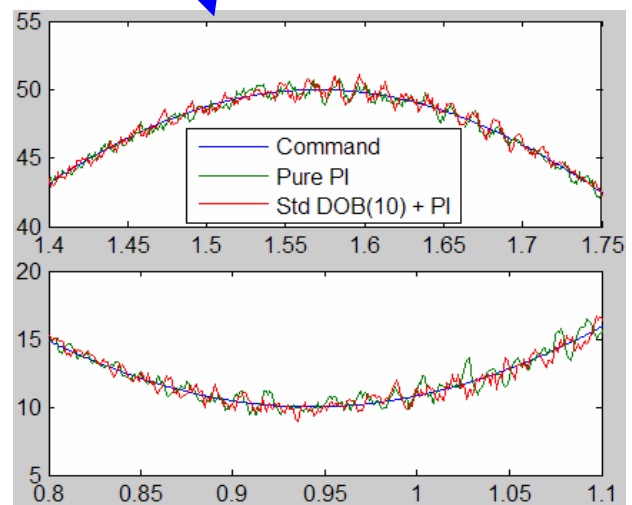
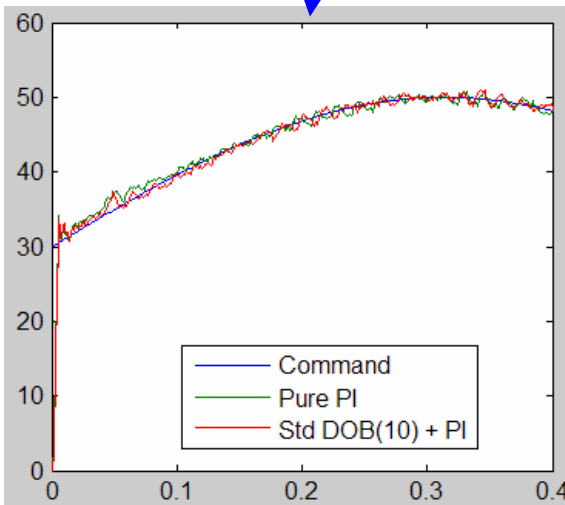
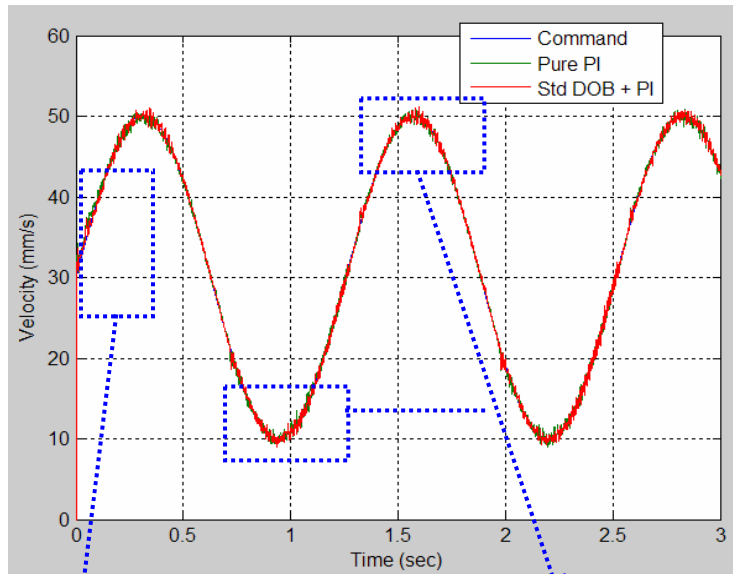


圖 6.26 實驗九結果響應

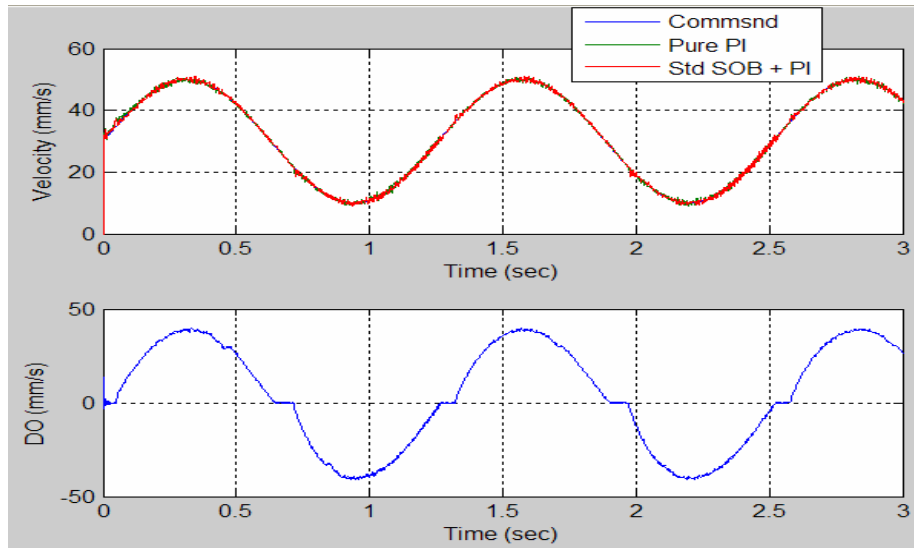


圖 6.27 實驗九結果與 DO 之相對關係

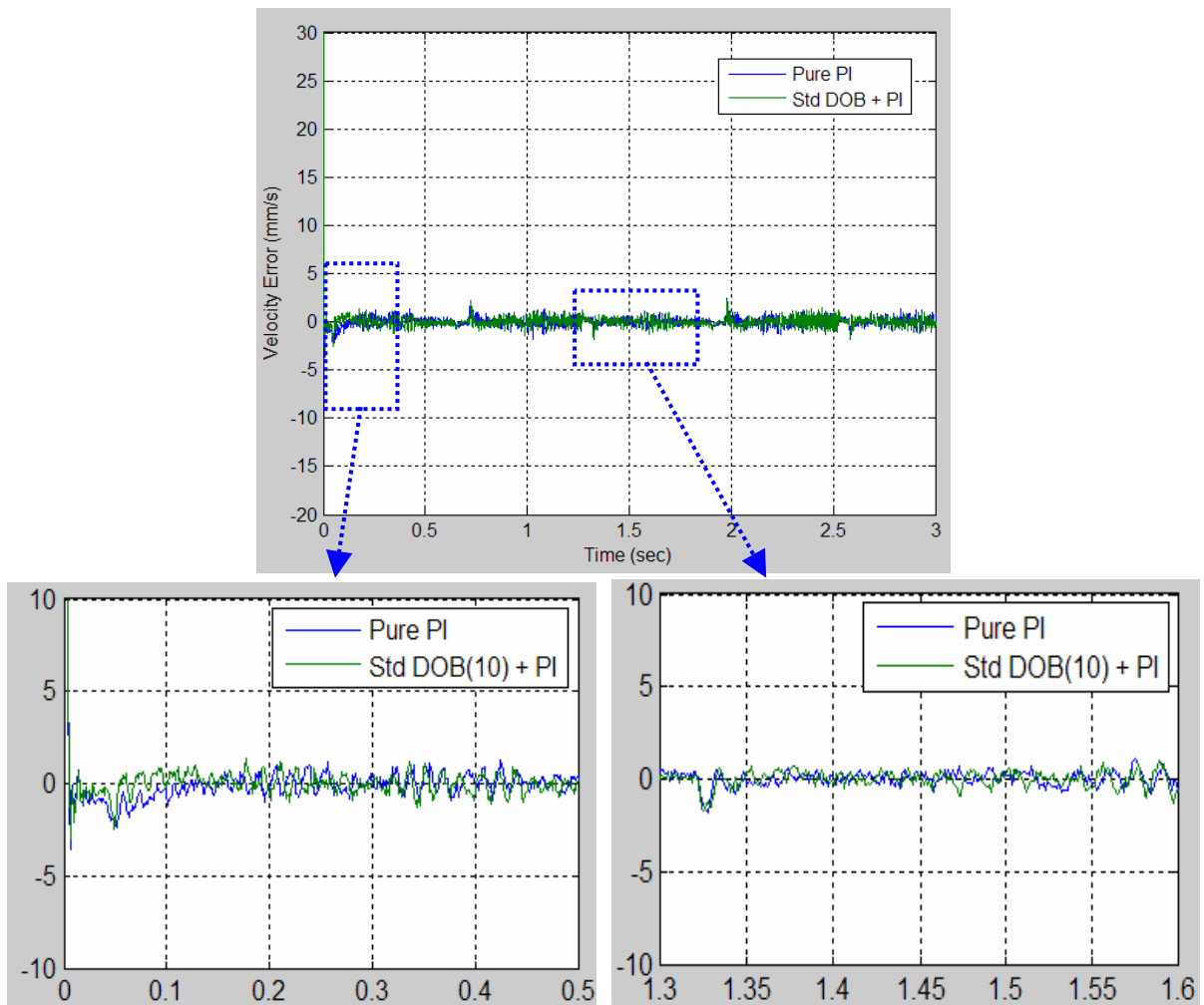


圖 6.28 實驗九之速度誤差

討論九：

在實驗九中，人工 DO 如圖 6.25 所示，並且以減去質量塊作為不確定性的影響，結果如圖 6.26、6.27 所示，由圖 6.28 我們可以知道，加入 DOB 架構之後，暫態的收斂速度雖然比純 PI 快，但在穩態時響應比純 PI 控制稍差，推測可能是因為 DO 成分有高频存在(速度轉折點)，而傳統 DOB 對於高频的輸出干擾抑制能力不佳，讓我們從數學上回顧傳統 DOB 對抗輸出干擾的能力，如(5.37)式中所表示的 $d_o \rightarrow y$ 轉移函數為：

$$G_{d_o,y}(s) = \frac{P_n(s)(1-Q(s))}{P_n(s) + (P(s) - P_n(s))Q(s)}$$

若假設 $P(s) - P_n(s) \cong 0$ ，則 $G_{d_o,y}(s) \cong (1-Q(s))$ ，因此若 $Q(s)$ 設計為 low-pass filter 時，在低頻對抗輸出干擾的效果不錯，但因為 $Q(s)$ 為 low-pass filter，所以 $1-Q(s)$ 為 high-pass filter(如圖 6.29)，因此高频的輸出干擾將會直接表現到輸出響應上。但若將 $Q(s)$ 的頻寬提高來容納高频輸出干擾，則又會讓高频雜訊進入系統，故傳統 DOB 無法同時應付高频輸出干擾及高频雜訊的缺點便在此展現出來

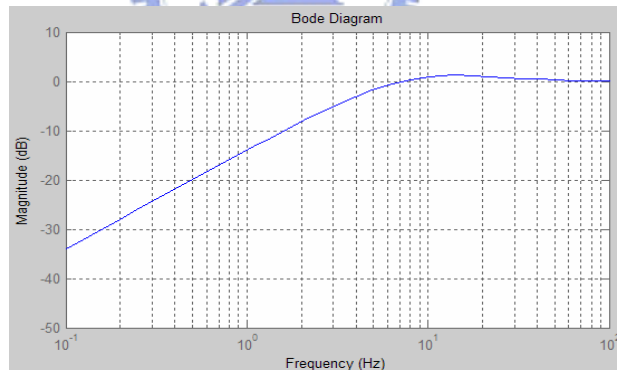


圖 6.29 傳統 DOB $d_o \rightarrow V$ 之頻率響應

實驗十：Sin wave 輸出干擾、零速度追蹤響應

實驗十	架構一： 純 PI	架構二： 傳統 DOB+PI	架構三： DCF DOB+PI
速度命令	$V_r = 0 \text{ mm/s}$	$V_r = 0 \text{ mm/s}$	$V_r = 0 \text{ mm/s}$
輸出干擾	$d_o = 15\sin(20\pi t) \text{ mm/s}$	$d_o = 15\sin(20\pi t) \text{ mm/s}$	$d_o = 15\sin(20\pi t) \text{ mm/s}$
控制器參數	$PI \begin{cases} K_p = 100 \\ K_i = 2000 \end{cases}$	$PI \begin{cases} K_p = 100 \\ K_i = 2000 \end{cases}$ Std DOB $\{Q = 30 \text{ Hz}\}$	$PI \begin{cases} K_p = 100 \\ K_i = 2000 \end{cases}$ DCF DOB $\begin{cases} Q_1 \text{ 為(5.46)式} \\ Q_2 \text{ 為(5.47)式} \\ Q_3 \text{ 為(5.48)式} \\ Q_4 \text{ 為(5.49)式} \end{cases}$
穩態均方根 誤差	4.82062949 mm/s (100%)	4.19739426 mm/s (87.07149%)	$\begin{cases} Q_1 = 3.617561 (75.0433\%) \\ Q_2 = 9.580171 (198.732\%) \\ Q_3 = 2.236188 (46.3878\%) \\ Q_4 = 1.976329 (40.9973\%) \end{cases}$ mm/s

表6.10 實驗十相關資料

在此實驗中，我們將引入之前所說明的 DCF DOB 進行抑制輸出干擾的實驗，由於在 SISO 系統中，若 DCF DOB 之 Q 設計成一般 low-pass filter 則效果等同於傳統 DOB，因此在前面實驗中，均沒有出現 DCF DOB 的比較。但在此實驗中，我們將 DCF DOB 中的 Q 設計成(5.46)~(5.49)，使得 d_o 到 V 的轉移函數 $G_{d_o V}(s) = I - QY_l\tilde{N}$ 之頻率響應形成一個 Notch filter，在特定頻率(10 Hz)砍掉輸出干擾，並且與傳統 DOB 以及純 PI 控制架構的進行比較。

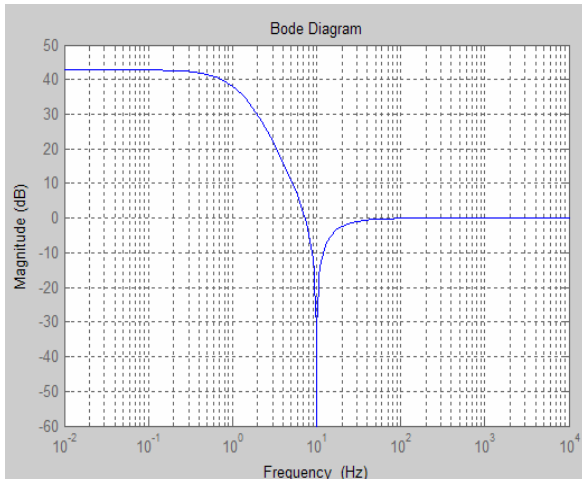


圖 6.30 Q_1 形式 $d_o \rightarrow V$ 之頻率響應

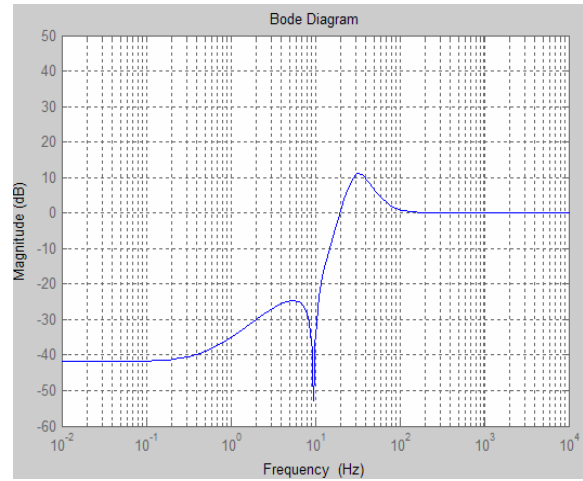


圖 6.31 Q_2 形式 $d_o \rightarrow V$ 之頻率響應

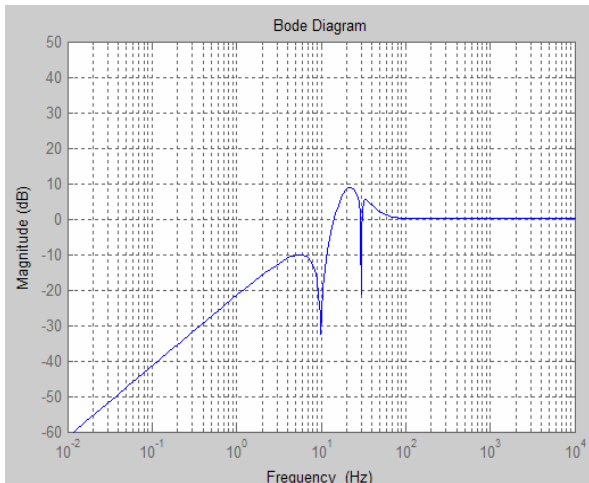


圖 6.32 Q_3 形式 $d_o \rightarrow V$ 之頻率響應

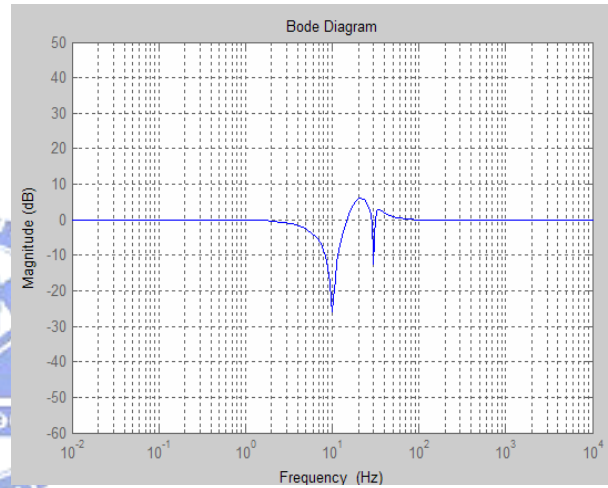


圖 6.33 Q_4 形式 $d_o \rightarrow V$ 之頻率響應

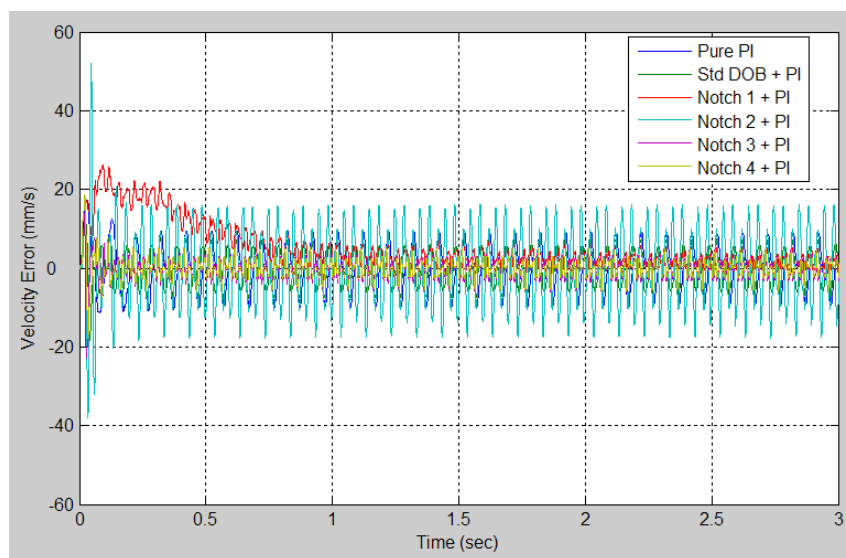


圖 6.34 實驗十速度響應誤差

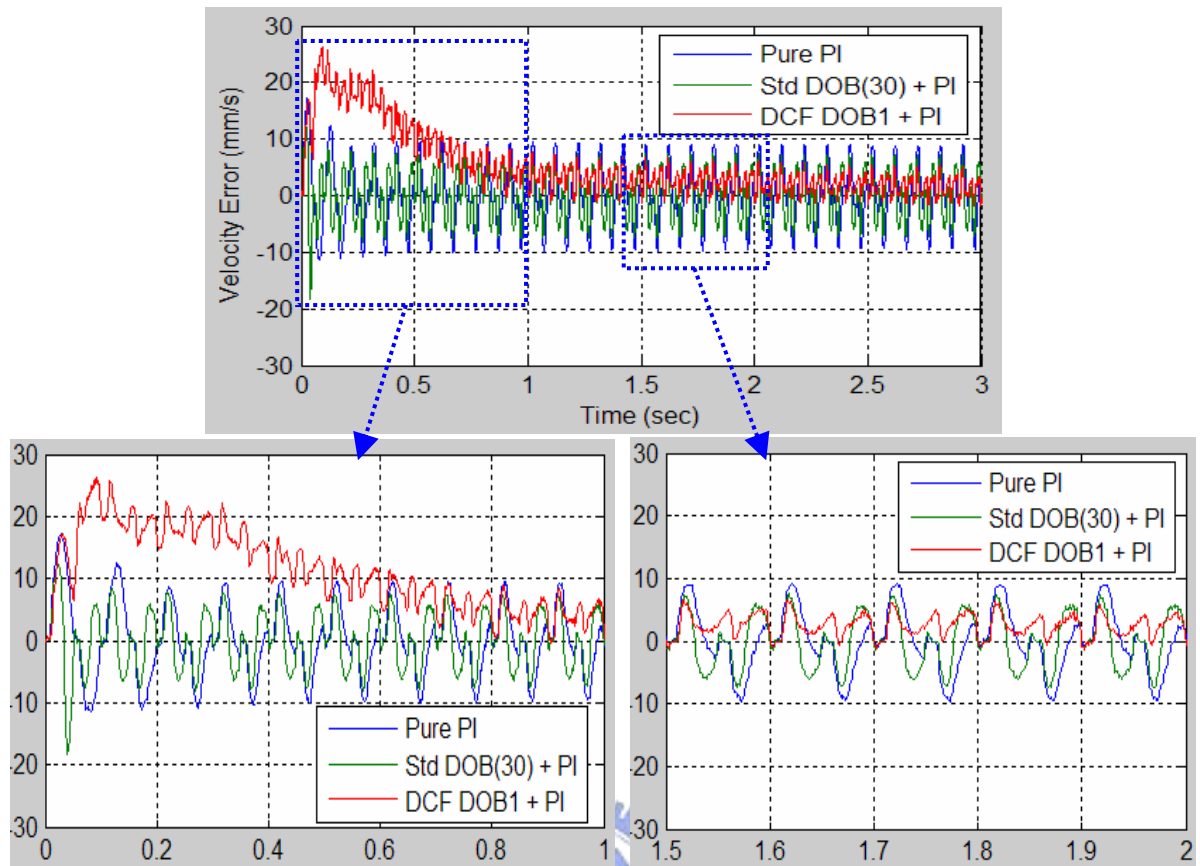


圖 6.35 $notch Q_1$ 之速度誤差響應

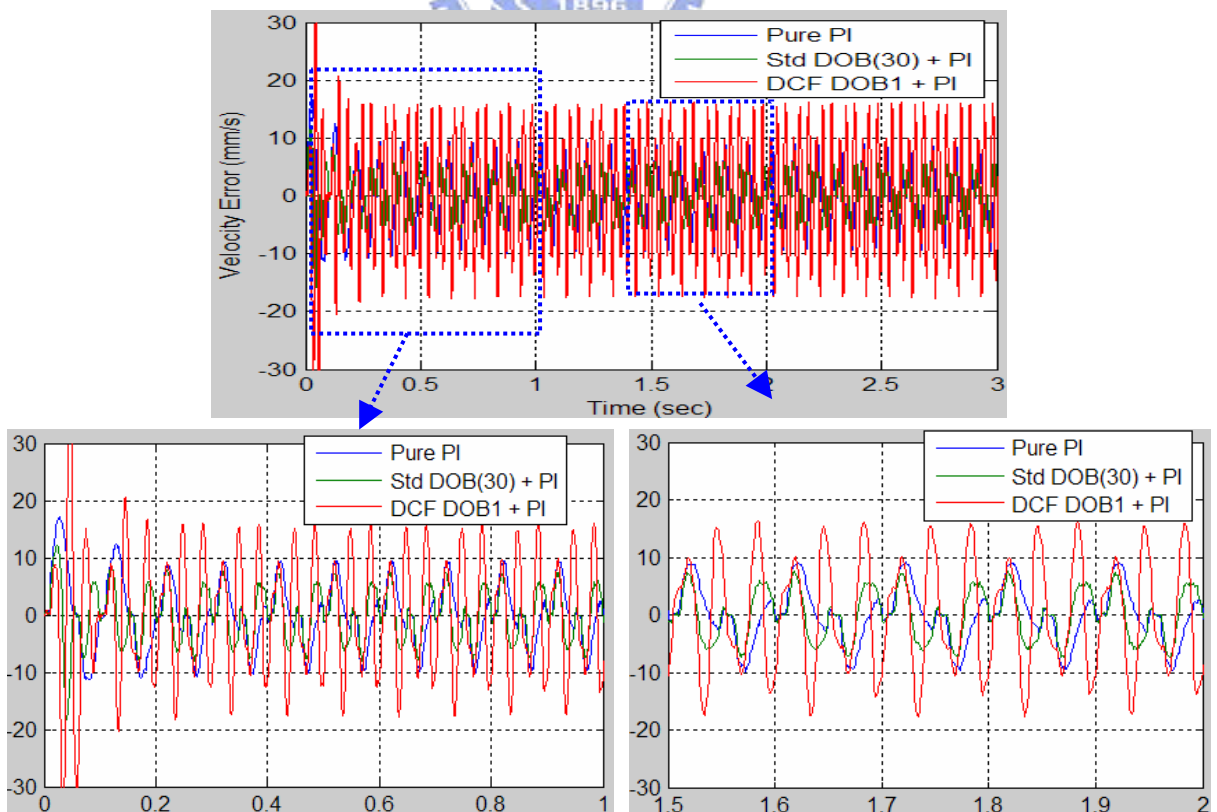


圖 6.36 $notch Q_2$ 之速度誤差響應

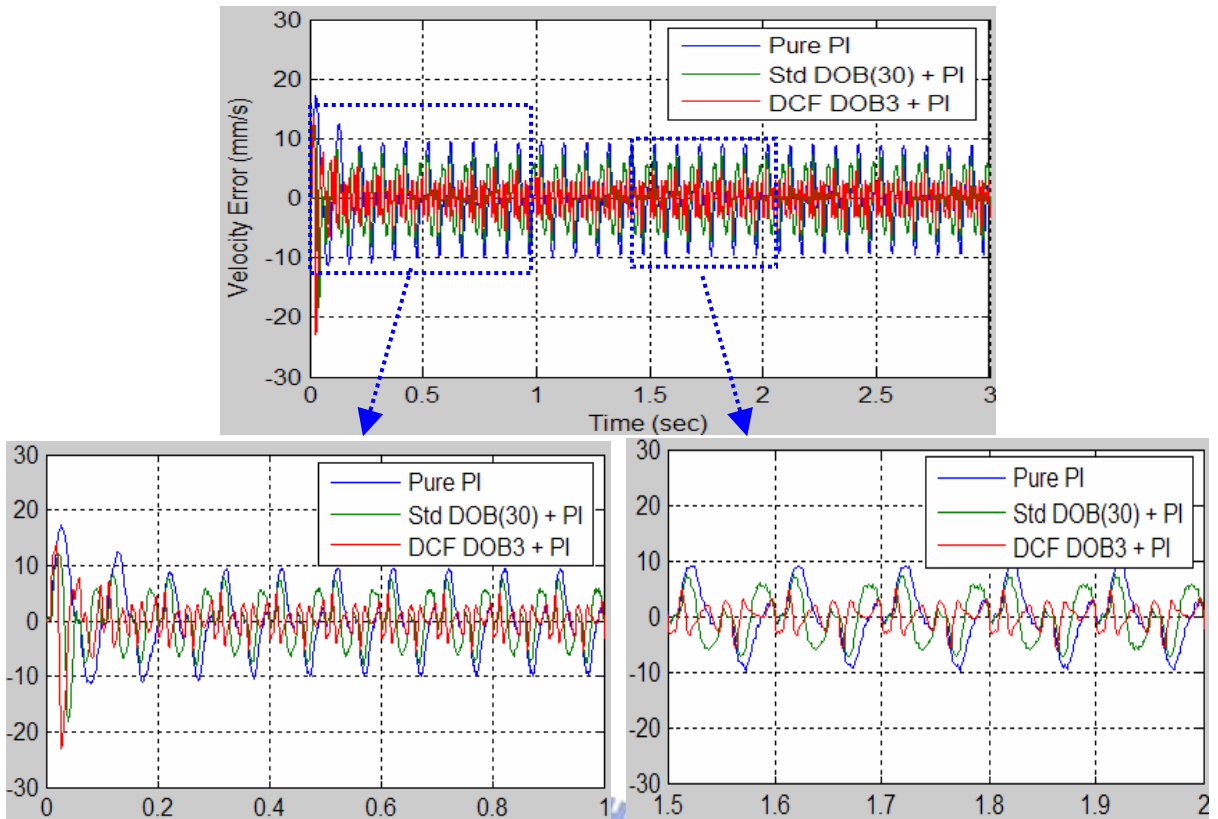


圖 6.37 notch Q_3 之速度誤差響應

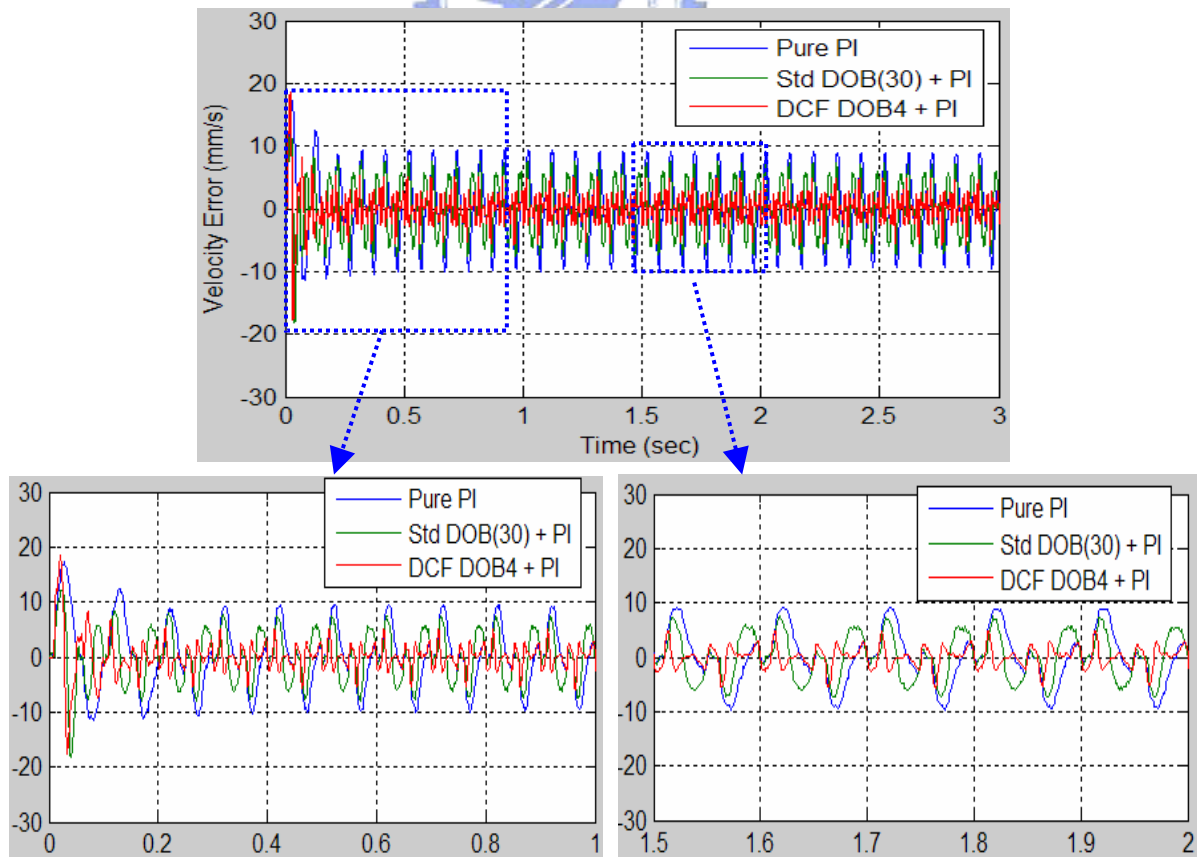


圖 6.38 notch Q_4 之速度誤差響應

討論十：

由上個實驗的討論我們知道傳統 DOB 對於高頻的輸出干擾(頻率超過 DOB 頻寬)壓制效果有限，因此在此實驗中選定 30Hz 為傳統 DOB 頻寬，比輸出干擾 10Hz 更高，故理論上能夠排除輸出干擾的影響，但有可能導致雜訊進入系統。

如圖 6.30~6.33，我們設計了不同類型的 Notch DCF DOB，其中每一次的設計都是由上一次設計的經驗以及我們的需求加以修改，圖 6.30 為第一次設計的 notch，在 10Hz 處壓低，但是低頻卻因為水床效應而浮起，由表 6.10 以及圖 6.34、6.35 可知設計成 $Q_1(s)$ 時，系統在穩態時的方均根誤差(75.0433%)比傳統 DOB (87.07149%)以及純 PI 架構(100%)更加優良，但由於低頻放大太多(high gain)，因此若系統有初始值，則可能導致機台的飄移，如圖 6.35 中 DCF DOB 1 的響應曲線應該就是機台本身的飄移被放大所造成的。

重新設計 $Q_2(s)$ 如圖 6.31 所表示，為了避免低頻為 high gain，因此壓低低頻，可是卻導致 30Hz 左右翹起，由實驗結果(圖 6.36)可看出此組響應結果非常糟糕，連穩態誤差幾乎是純 PI 的 2 倍，為了探討設計成 $Q_2(s)$ 形式後，系統響應變差的原因，我們將輸出干擾找出，如圖 6.39，並且對此輸出干擾進行 FFT 如圖 6.40。

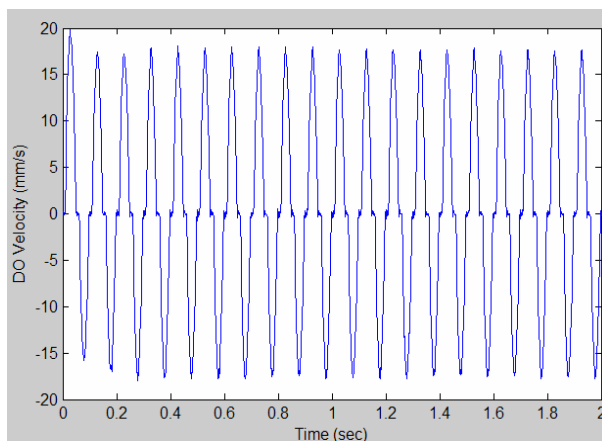


圖 6.39 輸出干擾(10Hz)之響應

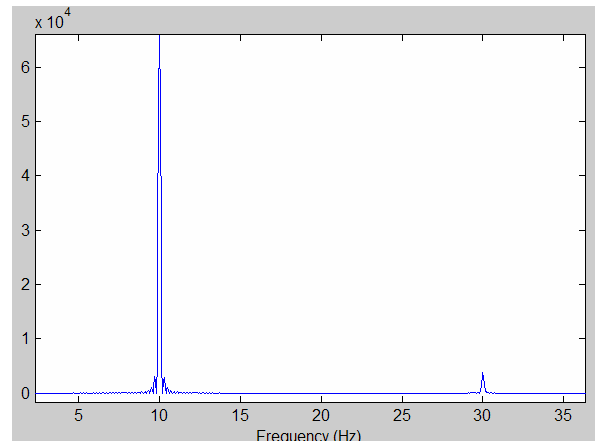


圖 6.40 輸出干擾之 FFT 分析

由 FFT 圖我們可以知道，此輸出干擾在 10Hz 為其主頻，在 30Hz、50Hz、70Hz....都

有其簡諧項，而且由圖 6.31 $d_o \rightarrow V$ 之 bode 圖可知正好在 30Hz 處 $notch Q_2$ 不但沒有壓制干擾，反而將干擾放大，因此造成此組不理想的響應。

根據上面提到輸出干擾的 FFT 分析，我們知道此干擾所有的能量幾乎集中在前三個簡諧頻率之中，而第一個副頻位在 30Hz 處，因此設計 $notch Q_3$ 為雙頻 notch filter，如圖 6.32，在 10Hz 與 30Hz 分別壓低大小，雖然在 20Hz 處則因水床效應而浮起，但由 FFT 分析我們可以知道干擾在 20Hz 左右並沒有太大的成分，因此將這組設計實現並做實驗，由實驗結果(圖 6.37 及表 6.10)可知， $notch Q_3$ 的設計表現出非常好的控制效果，其穩態 RMSE 降低為純 PI 迴授控制的 46.3878%、降低為加入傳統 DOB 架構控制的 53.2786%，展現出 DCF DOB 優越的一面。

然而，在 $notch Q_4$ 的設計中又更進一步的將原本 $notch Q_3$ 的 20Hz 翹起部分加以壓低，犧牲低頻的壓制效果(低頻為 0 dB)以換取較低的最大峰值，如圖 6.33，由實驗結果(圖 6.38、表 6.10)可知， $notch Q_4$ 的設計使系統的響應更勝 $notch Q_3$ ，已將穩態 RMSE 將低為純 PI 控制架構的 40.9973%、降低為傳統 DOB 架構的 47.0846%。

以上四組 DCF DOB 以及傳統 DOB 和純 PI 控制架構對於 10Hz 輸出干擾的抵抗能力，雖然經由上面的討論及實驗結果已經明瞭，但在此將會以另一種角度，FFT 分析，來評斷上述六種控制性能。我們將以上六組實驗的結果進行 FFT 分析，並與圖 6.40 進行比較，如下頁圖所示：

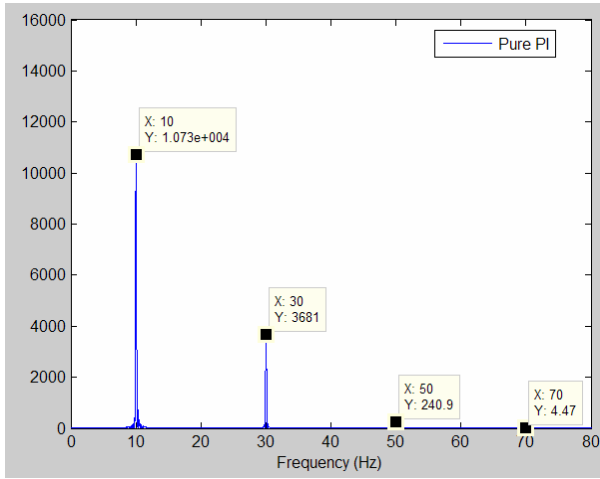


圖 6.41 純 PI 響應之 FFT 分析

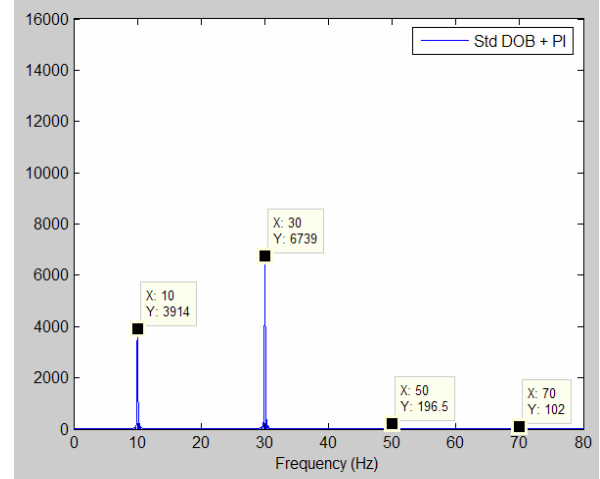


圖 6.42 傳統 DOB+PI 響應之 FFT 分析

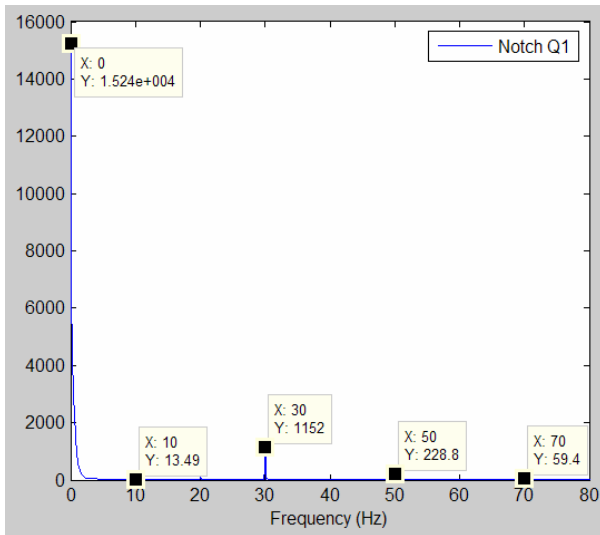


圖 6.43 Notch1+PI 響應之 FFT 分析

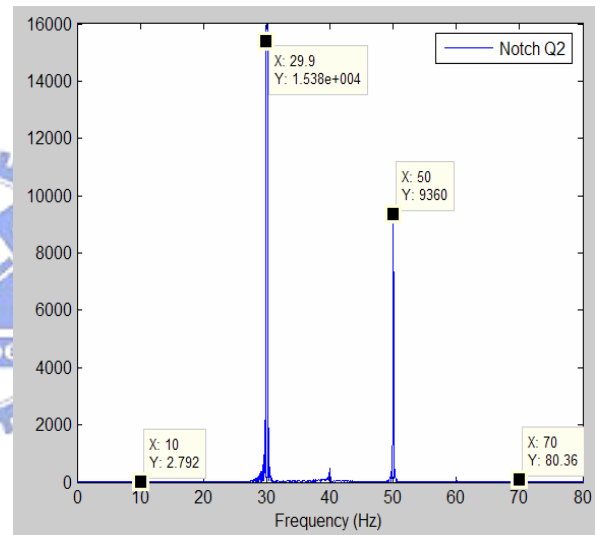


圖 6.44 Notch2+PI 響應之 FFT 分析

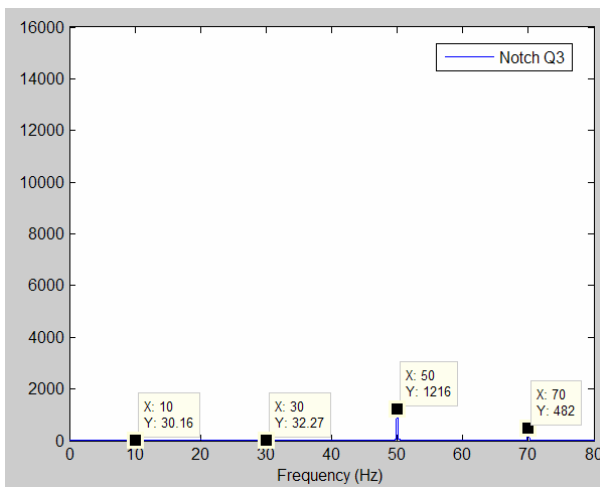


圖 6.45 Notch3+PI 響應之 FFT 分析

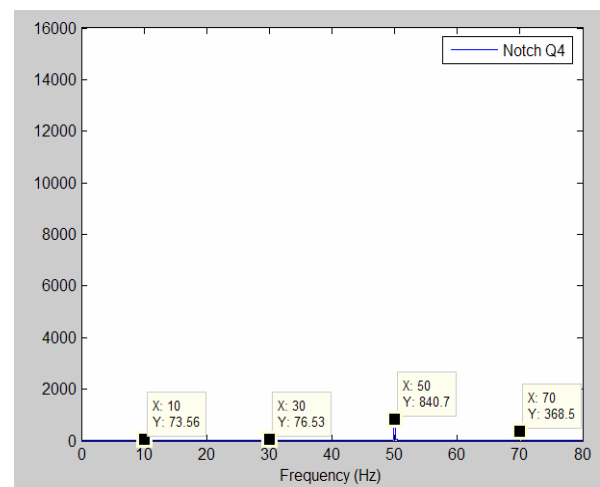


圖 6.46 Notch4+PI 響應之 FFT 分析

由圖 6.41~6.46 我們可以清楚的知道，正如我們前面所說的，傳統 DOB 已經將 30Hz 頻寬以內的干擾成分壓低，但 30Hz 的簡諧項則無法很好的壓制；至於 Notch 1 的效果其實不差，只是因為低頻時 high gain 導致有使用上的限制；而 Notch 2 可以從 FFT 分析中清楚的看到 30Hz 以及 50Hz 被放大許多，因此最後響應才會如此糟糕；Notch 3 以及 Notch 4 則是在高低頻互有優劣，因為 Notch 4 放棄了少部分低頻，因此從圖 6.46 中可以看出低頻影響比 Notch 3 大，但高頻部分則優於 Notch 3。經由 FFT 分析的觀點，我們得以再次驗證前面討論的結果。

綜合以上實驗結果，我們可以知道傳統 DOB 對於系統的輸入干擾以及輸出干擾都能發揮不錯的抑制效果(實驗二~實驗七)，但傳統 DOB 遇上高頻的輸入、輸出干擾時就顯得力不從心，而 DCF DOB 在 SISO 系統中雖然難以表現其許多優點，但是經由彈性的設計 Q ，我們可以讓 DCF DOB 解決高頻的干擾而不犧牲太多的控制性能，因此，我們在馬達控制系統之中，驗證了干擾觀測器的現在，並期待著干擾觀測器的未來。



6.3 實驗問題討論

在本節中實驗問題討論中，將會提出在實驗過程中所遇到的問題，並且將之提出討論以及心得，本文實驗過程中遇到的問題包括 $\alpha-\beta$ filter 頻寬設計問題、驅動器飽和問題、控制器實現問題，接下來我們將分別討論這些問題。

6.3.1 $\alpha-\beta$ filter 頻寬問題

實驗過程中，由於速度資訊無法直接得到，因此皆是採用位置資訊的微分，而在本文前面也提到，普通的微分器會導致高頻雜訊的放大，所以通常採用 $\alpha-\beta$ filter 來進行速度的估測。而 $\alpha-\beta$ filter 頻寬的選擇也將導致系統速度響應的估測。

一般來說， $\alpha-\beta$ filter 的頻寬太小時(與速度迴路頻寬相等)則會造成速度估測的太慢，而控制架構加強補償而導致系統暴衝，如圖 6.47 所示，而 $\alpha-\beta$ filter 頻寬為 2 倍速度迴路的頻寬時，速度估測的效果也不甚理想，如圖 6.48 所示：

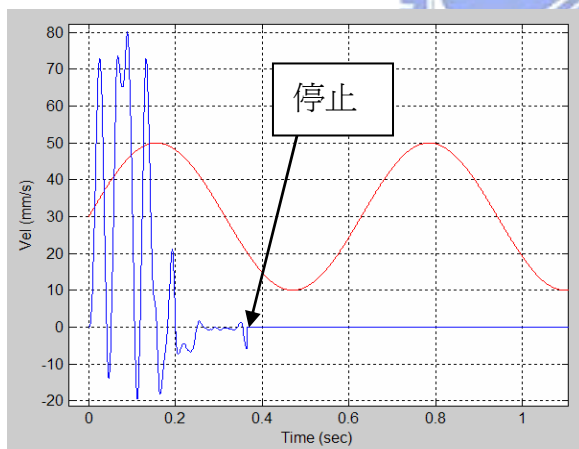


圖 6.47 $\alpha-\beta$ filter 為 1 倍系統頻寬

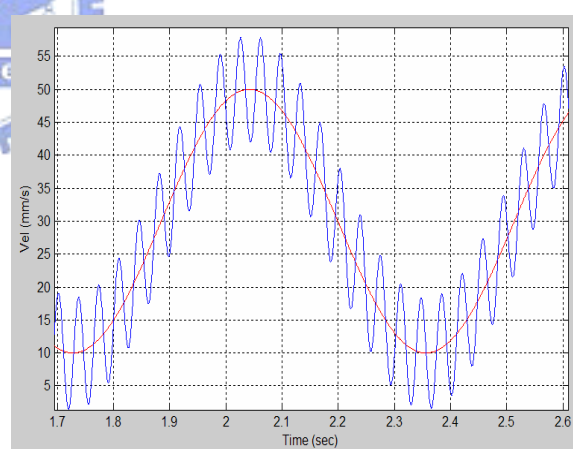


圖 6.48 $\alpha-\beta$ filter 為 2 倍系統頻寬

我們可以很直觀的發覺，隨著將 $\alpha-\beta$ filter 頻寬越大， $\alpha-\beta$ filter 估測速度的能力也就越強，但當 $\alpha-\beta$ filter 頻寬太大時，此時雜訊又會引想系統速度的估測，如圖 6.49 表示 $\alpha-\beta$ filter 頻寬為 7、8、9 倍(294Hz、336Hz、360Hz)系統頻寬時，速度估測結果。由圖可以發現雜訊影響非常嚴重，震盪的很厲害。

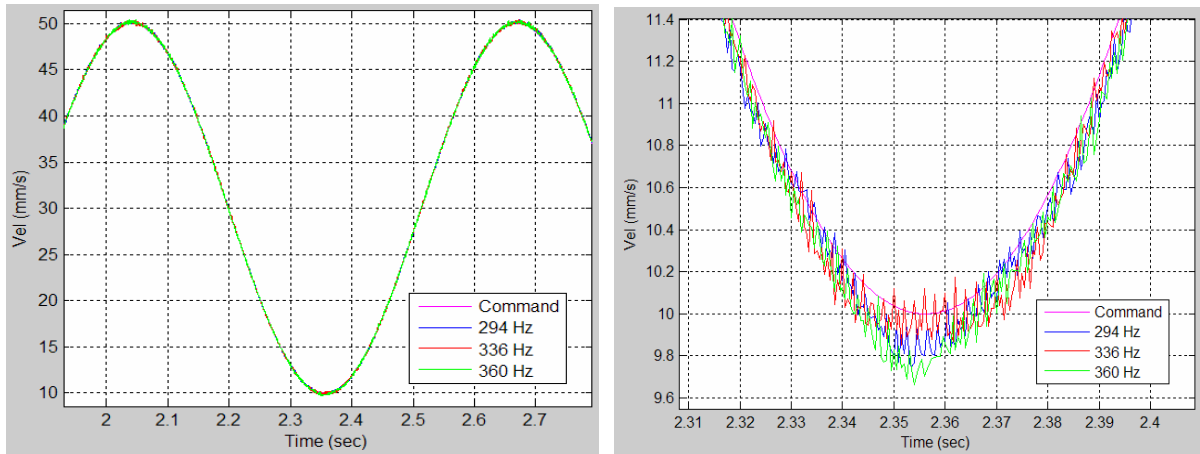


圖 6.49 $\alpha - \beta$ filter 為 7、8、9 倍系統頻寬

所以在本文中對於 $\alpha - \beta$ filter 頻寬的選擇，則是選擇介於 3 倍~5 倍系統頻寬的頻寬作為 $\alpha - \beta$ filter 的頻寬，如下圖表示 3、4、5 倍(126Hz、168Hz、210Hz)系統頻寬。

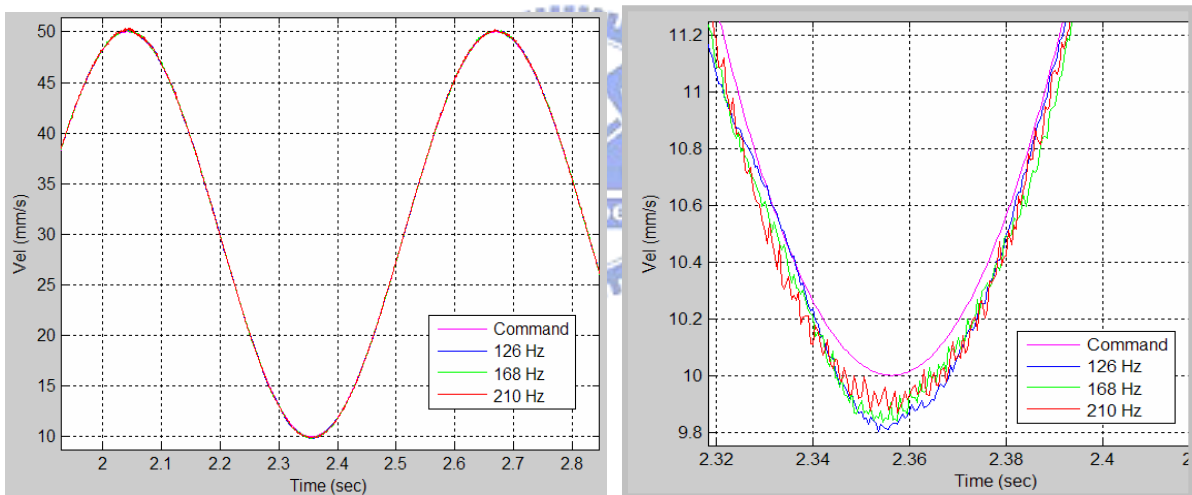


圖 6.50 $\alpha - \beta$ filter 為 3、4、5 倍系統頻寬

由圖 6.50 我們可以知道，當 $\alpha - \beta$ filter 頻寬為 5 倍系統頻寬時，能夠得到不錯的速度估測結果，也不置於讓太多高頻雜訊進入系統，因此本文中便採用 5 倍系統頻寬為 $\alpha - \beta$ filter 的頻寬。

6.3.2 驅動器飽和問題

當我們討論輸出干擾的時候，曾經提到輸出干擾並未經過 plant，因此輸出干擾的

大小將會直接影響輸出響應，同理，輸出干擾的大小也將會直接影響控制架構的補償，因此，在本文實驗中所發現的另一個問題就是，輸出干擾大會導致 PI 迴授控制以及 DOB 架構對於輸出干擾的補償很大，如圖 6.51 所示：

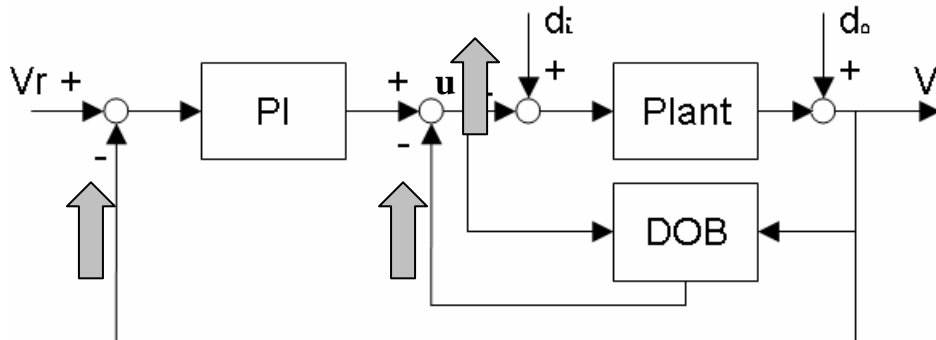


圖 6.51 d_o 對於控制架構的影響

由圖 6.51 可知，系統的控制器輸出 u 可能在一瞬間變的很大，導致 D/A 輸出的電壓很大，到達馬達驅動器的飽和電壓，造成馬達驅動器飽和。因此為了降低控制架構補償的強度，我們對控制器參數進行調整並且將電壓輸出記錄下來，如下圖 6.52 所示：

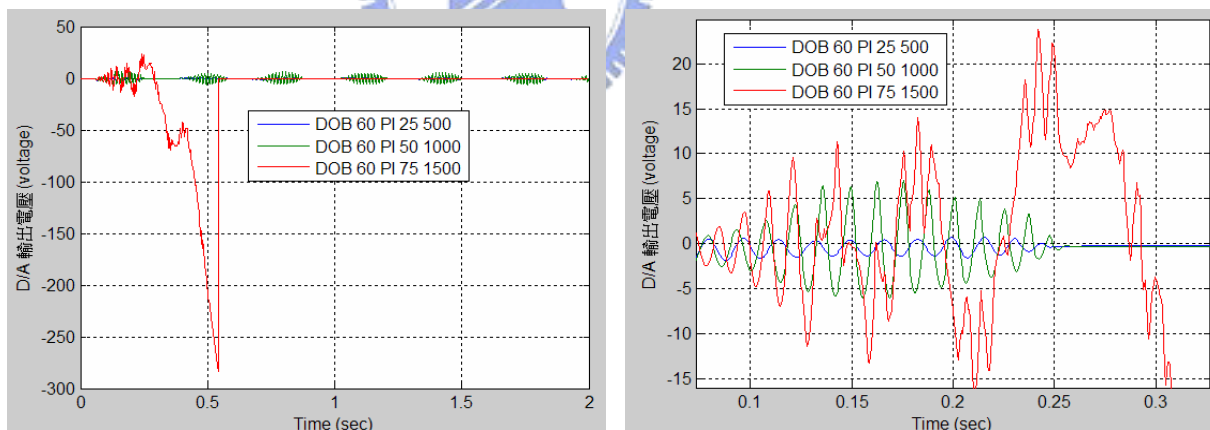


圖 6.52 不同控制參數的 D/A 電壓輸出

在本文中，馬達驅動器的飽和電壓為 $\pm 10V$ ，圖 6.52 中清楚表示出若 PI 控制器參數大，則輸出電壓震盪較大，並且只要 D/A 輸出電壓到達驅動器馬達的飽和電壓，即造成驅動器飽和，導致實驗失敗。

為了解決此問題，在設計控制器時必須考慮此因素，必須稍微降低 PI 或 DOB 架構其中一項的補償能力，才不會再次導致驅動器飽和，經過多次實驗後，發現降低 DOB

頻寬、提高 PI 參數可以有不錯的效果，因為 DOB 頻寬只要能夠包容干擾就夠了，太大的 DOB 頻寬可能造成高頻雜訊以及高頻干擾的放大，影響系統的響應。

6.3.3 控制器實現問題

在控制器的實現部分，我們先在連續時間下設計系統的控制器，並且將設計完成的連續時間控制器進行數位化，成為離散時間的差分方程式，數位化的方式在[19]中也提到許多種不同的方法。在本文中採用 3-D 的方式進行數位化實現，並且確定控制器的設計、數位化、程式撰寫均無錯誤，但在實驗過程中卻發現系統產生不穩定的現象。

經過檢查、討論後發現，雖然程式撰寫、數位化方式、控制器設計都沒有問題，但仍有可導致系統的不穩定，以(5.46)~(5.49)式為例：

$$Q_1(s) = \frac{-286.2s - 8518}{s + 10} \quad (5.46)$$

$$Q_2(s) = \frac{2738659.1839s(s + 9999)(s + 10)(s + 8)(s^2 + 108s + 7743)}{s(s + 10000)(s + 182.2)(s^2 + 316.9s + 39180)(s^2 + 113.2s + 37960)} \quad (5.47)$$

$$Q_3(s) = \frac{636260.3281(s + 10)(s + 8)(s^2 + 48.41s + 2852)(s^2 - 11.85s + 33900)}{(s + 99.7)(s^2 + 189.7s + 13230)(s^2 + 86.36s + 16530)(s^2 + 19.19s + 36350)} \quad (5.48)$$

$$Q_4(s) = \frac{215506.7175s(s + 165.9)(s + 10)(s + 8)(s^2 - 32.46s + 33380)}{(s + 103.7)(s^2 + 197.8s + 14290)(s^2 + 92.95s + 17660)(s^2 + 24.72s + 36910)} \quad (5.49)$$

在實現之前，我們將內迴路 DOB 控制架構方塊圖簡化，由圖 6.53 我們知道原本控制架構，若直接進行數位化實現則可能會因為太多層次而造成 delay，因此將架構簡化如後：

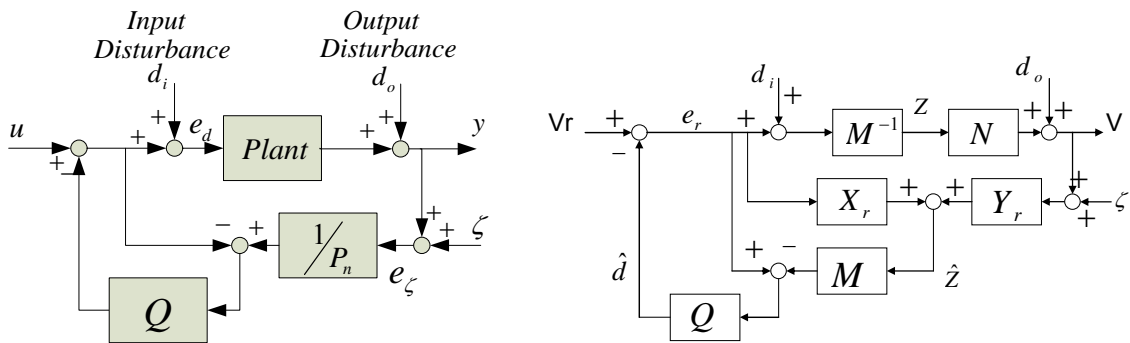


圖 6.53 傳統 DOB 以及 DCF DOB 之方塊圖

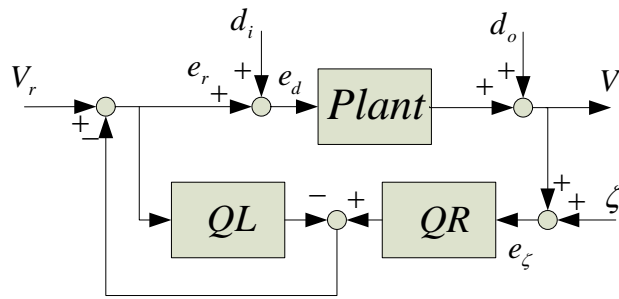


圖 6.54 控制架構實現之方塊圖

除了 $Q_1(s)$ 以外的 $Q_2(s)$ 、 $Q_3(s)$ 、 $Q_4(s)$ 經過簡化包裝成 QL 與 QR 後，在經過 z 轉換時 QL 與 QR 的分母、分子階數很高，分別為分子 8 階、分母 9 階 ($\frac{8\text{階}}{9\text{階}}$)，因此導致係數敏感度很高，如果係數擷取的不夠精確(位數不夠)，很可能導致系統的極、零點稍微偏移。檢查後發現，在實現時控制器會有很接近單位圓的極點(z-domain)，因此若極、零點發生偏移移出單位圓之外時，就會導致系統不穩定。如下圖 6.55 為 $Q_3(s)$ 理想的數位化實現在 z-domain 之下的 pole-zero map。

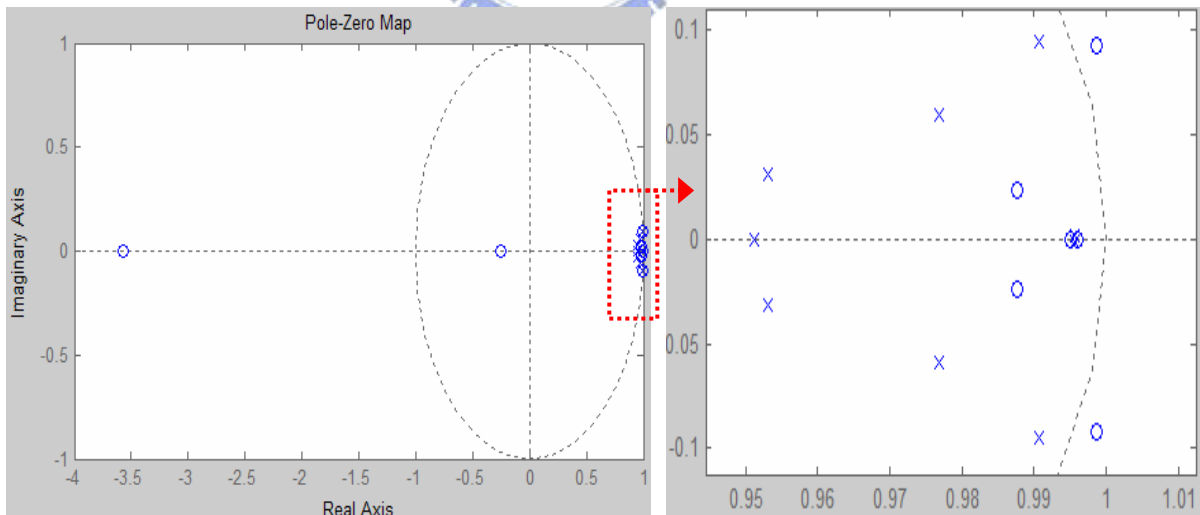


圖 6.55 理想數位化實現之 pole-zero map

由圖 6.55 我們可以知道有幾組極點位置很接近單位圓，但都還是落在單位圓內，因此系統還是維持穩定的狀態。但若是我們數位化的過程中無法完整的擷取參數時(或參數擷取的不夠精確時)，圖 6.55 的系統將變成下圖 6.56、6.57 所示：

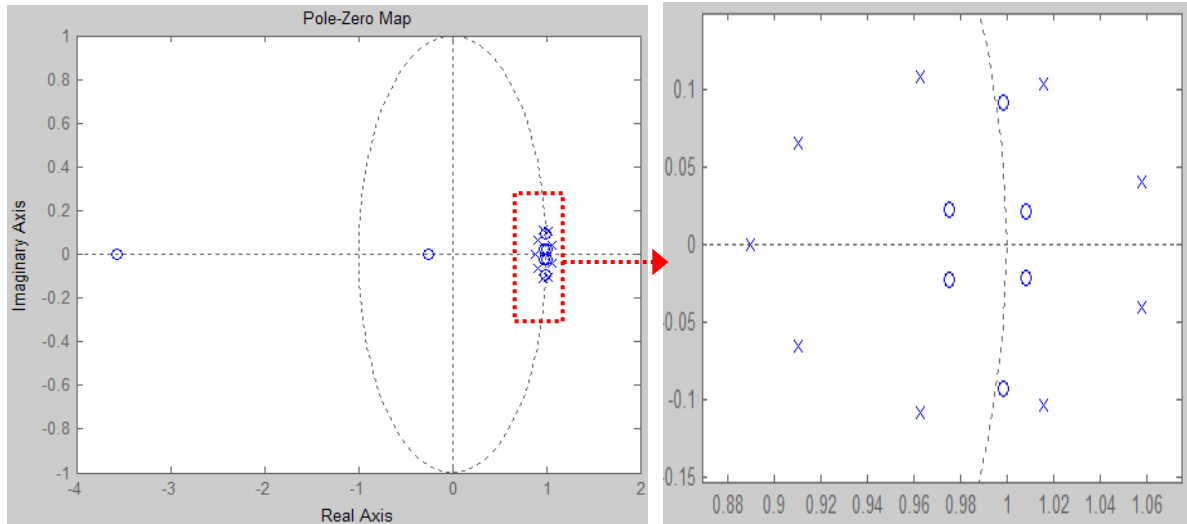


圖 6.56 實際數位化實現之 pole-zero map (參數取 9 位數)

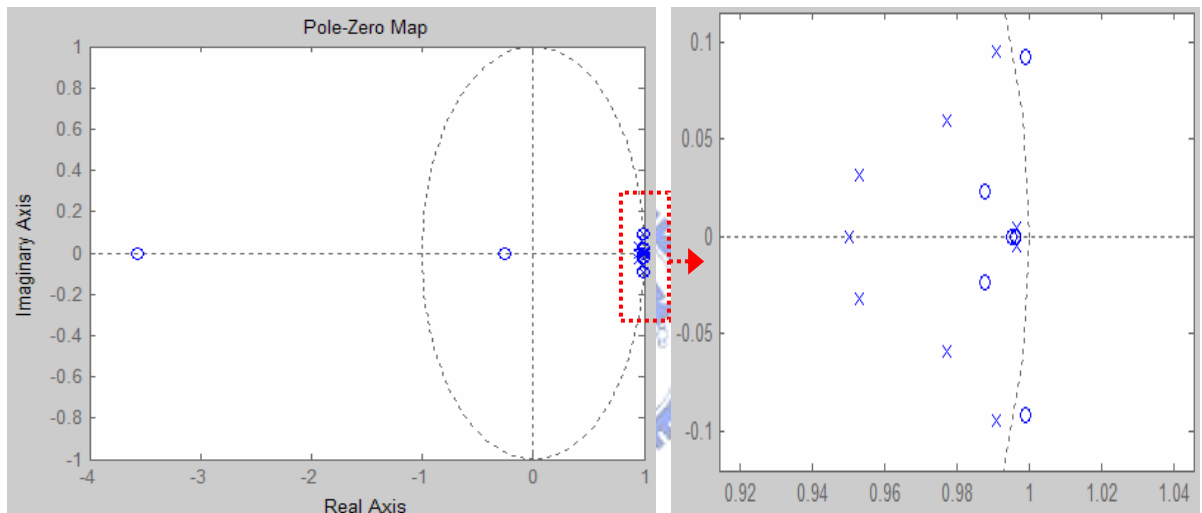


圖 6.57 實際數位化實現之 pole-zero map (參數取 15 位數)

由圖 6.56 可知，當參數只取 9 位數的時候，因為原本的極點非常靠近單位圓，因此一有飄移就跑出單位圓之外，導致系統不穩定。而圖 6.57 中我們也可以看出，參數取 15 位數時，雖然極點還保留在單位圓之內，但是已經從原來的位置(圖 6.55 所示)偏移掉，因此響應也無法達到理想數位化實現時的效果。

由上面的圖以及討論可知，我們在實現數位控制器時，一定會遇到這方面的問題，所以，解決此問題的方式有以下幾種：

1. 係數擷取盡量精確：

數位化實現時，參數的位數越多(小數點後 15 位數)時，則 z-domain 極零

點飄移較少，比較不會造成系統極零點飄移而導致不穩定。

2. 控制器降階：

我們也可將控制器降階，降低參數的敏感度，簡單的極零點對消降階後，由原先的 $8^{\text{階}}/9^{\text{階}}$ 降低為 $6^{\text{階}}/7^{\text{階}}$ ，可以發現參數擷取的位數可以降低至小數點後 9 位，仍可保持系統穩定，因此若能再將控制器降階，應該會有更好的效果。

3. 找尋其他數位化的方式。

而在本文實驗中對於 1. 以及 2. 兩種方法都以實現成功，在實驗結果上差異不大，因為極零點對消的降階方式不會對系統的響應產生太大的影響，在本文實驗採 1. 的方式實現。



第七章 結論與未來展望

7.1 結論

DOB 架構一直是各家學者研究的目標，因為它的精神以及優點使它不斷的被研究，包括能估測等效干擾及誤差、DOB 控制器設計簡單(low-pass filter)、能逼近系統成為 nominal plant(對抗 uncertainty)、簡化外迴路控制器設計等等，但也漸漸地發現它的缺點，包括對不連續干擾的效果不佳、無法排除高頻干擾、無法處理非極小相位系統、對非方陣 MIMO 的系統，因此許多 DOB 改良的架構不斷出現，在本文中提到實驗室所發展的 Coprime DOB 架構主要是針對非極小相位以及 MIMO 的系統所設計，在本文中藉由模擬以及 PC-based 實驗來驗證並實現此架構在線性馬達伺服控制系統之中。

在馬達控制系統中，因為結構設計的關係導致永磁馬達產生鈍齒效應(cogging)，影響機台精密定位的控制，是馬達輸入干擾的重要因素。除了輸入干擾(input disturbance)之外，更加考慮外界干擾導致輸出端 sensor 測量上的誤差等輸出干擾(output disturbance)，並且針對輸出干擾將 Coprime DOB 設計成 Notch filter 形式來特別抵抗輸出干擾，實驗結果可知 Coprime DOB 對抗特定干擾的能力確實較傳統 DOB 優異，換言之，我們可以針對不同的需求而設計特定的 CDOB 控制器，但也不會犧牲太多傳統 DOB 對抗干擾的性能，比傳統 DOB 提升了不少使用彈性及指定強度。

在實現上，實驗室對於即時多工運動控制卡的軟、硬體開發已有豐富的成果，目前能夠達到四軸 0.5ms 的伺服更新速率，因實驗室在軟體的規劃設計上已經有一定水準，因此若想要再提升運動控制卡的效能，則必須採用運算速度更快的 DSP，因此本文在前半部將實驗室開發的即時多工系統移植到 TI 的 TMS320C6722 中，配合使用 TI 公司所發展的即時作業核心 DSP/BIOS 以及作業系統發展工具 CCS 幫助軟體規劃及移植，並修改了 Task 配置、優先權規劃、資料佇列、信號機機制、Task 規劃以達到更好的移植效果，讓即時多工系統能運作的更流暢，並達到準確的定位。

7.2 未來展望

綜觀 DOB 控制架構的性能以及 DSP 軟體移植，已有不錯的成果，但仍有下列幾點改善需要改善與未來可發展的部分：

1. 在本文中雖然將即時多工控制系統進行移植，但實際在 DSP 運動控制卡上進行測試、除錯、實驗，將是未來可以進行的方向。
2. 雖然機台以雷射位移計作為機台的 Encoder，解析度最大最小分別為 $0.02 \mu m$ 、 $0.0791 \mu m$ ，但本文中使用 $0.0791 \mu m$ 的精度稍嫌太大($0.02 \mu m$ 有問題)，而且機台本身也會自動飄移數個 count，因此若能提高量測的精度，就能更加提高控制的準確度。
3. 由 Encoder 量測出機台位置資訊，但為了獲得機台速度資訊，我們必須將位置資訊微分，在本文中是採用 alpha-beta filter 來實現低頻微分的動作，但 alpha-beta filter 頻寬設計上會有響應與雜訊的取捨，因此，更準確的速度估測方法將是另一個可以進行的目標。
4. 在 DCF DOB 架構中，若設計成 Notch 形式，則 Q 的階數將會很高，導致數位化實現上有一定的難度，如參數精確度要求很高。是否有其他設計的方式能夠降低 Q 的階數、或是有某種數位化實現方法能夠降低參數變動靈敏度，將是未來的一個發展目標。
5. 在本文的研究目標之一，干擾觀測器補償鈍齒力，如果能以鈍齒力模型之主頻來進行 Notch filter 之設計補償，並比較直接以模型補償鈍齒力之效果，或許 DCF DOB 的應用會更廣泛。

參考文獻

- [1] 雷仕全，「DSP-based 運動控制卡之即時多工控制及伺服馬達鈍齒效應之補償」，碩士論文，國立交通大學機械研究所，民國九十五年七月。
- [2] 洪敬堯，「DSP-Based 之即時多工運動控制器研製」，碩士論文，國立交通大學機械工程研究所，民國九十四年十月。
- [3] S. M. Hwang, J. B. Eom, G. B. Hwang, W. B. Jeong, Y. H. Jung, “Cogging Torque and Acoistic Noise Reduction in Permanent Magnet Motors by Teeth Pairing”, *IEEE, Trans. Magn*, vol.36, no.5, Sep 2000.
- [4] C. Breton, J. Bartolome, J. A. Benito, G. Tassinario, C. W. Lu, B. J. halmers, “Influence of Machine Symmetry on Reduction of Cogging Torque in Permanent-Magnet Brushless Motors”, *IEEE, trans on magn*, vol.36, No.5 SEP 2000.
- [5] Z. Q. Zhu, D. Howe, “Influence of Design Paeameters on Cogging Torque in Permanent Magnet Machines”, *IEEE, trans on energy conversion*, vol.15, no.4, Dec 2000.
- [6] C. M. Liaw, R. Y. Shue, H. C. Chen, S. C. Chen, “Development of a linear brushless DC motor drive with robust position control”, *IEE Proc. Electr. Power Appl.*, vol.148, no.2, Mar 2001.
- [7] B. Yao, M. Al-Majed, M. Tomizuka, “High-Performance Robust Motion control of Machine Tools : An Adaptive Robust Control Approach and Comparative Experiments”, *IEEE, Trans. Mechatronics*, vol.2, no.2, Jun 1997.
- [8] T. Umeno, Y. Hori, ”Robust Speed Control of DC Servomoters Using Modern Two Degrees-of-Freedom Controller Design”, *IEEE, Trans. Ind. electronics*, vol.38, no.5, Oct 1991.
- [9] T. Umeno, T. Kaneko, Y. Hori, ”Robust Servosystem Design with Two Degrees of Freedom and its Application to Novel Motion Control of Robot Manipulators”, *IEEE, Trans. Ind. electronics*, vol.40, no.5, Oct 1993.
- [10] 潘怡仁，「雙互質分解干擾觀測器」，博士論文計畫書，國立交通大學機械研究所，民國九十六年八月。
- [11] Texas Instrument Inc., “TMS320 User’s Guide”, 2004
- [12] J. W. Jeon, Y. Y. Ha., ”A Generalized Approach for the Acceleration and Deceleration of Industrial Robots and CNC Machine Tools”, *IEEE, Trans. Ind. electronics*, vol.47, no.1, Feb 2000.
- [13] C. Studer, A. Keyhani, T. Sebastian, S. K. Murthy., ” Study of Cogging Torque in Permanent Magnet Machines” in *Proc. Rec. IEEE IAS Annu. Meeting New Orleans, LA*, October 1997, pp. 42-49.
- [14] D. C. Hanselman, *Brushless Permanent-Magnet Motor Design*, New York: Mc-Graw Hill, 1994.

- [15] K. H. Kim, D. J. Sim, J. S. Won, "Analysis of Skew Effects on Cogging Torque and BEMF for BLDCM" *Rec. IEEE Ind. Applicat. Soc. Annu. Meeting*, pp. 191-197, 1991.
- [16] 周亨隆,「線性馬達運動系統在巨觀及微觀區之下之摩擦力分析及補償」,碩士論文,國立交通大學機械研究所,民國九十三年七月。
- [17] 施禕迪,「直線馬達運動平台之高精度控制含摩擦力補償」,博士論文,國立交通大學機械研究所,民國九十三年四月。
- [18] 洪格中,「線性馬達平台誤差補償之順滑模態觀測器設計」,碩士論文,國立交通大學機械研究所,民國九十二年七月。
- [19] L. C. Phillips, H. T. Nagle, *Digital Control System Analysis and Design*, Prentice-Hall, Inc., Upper Saddle River, NJ, 1989

