工學碩士 學位論文

하이브리드방식에 의한 BLDC 모터 위치동기 제어계의 성능 향상

2002年 2月

釜慶大學校 大學院

메카트로닉스工學 協同課程

유 성 권

工學碩士 學位論文

하이브리드방식에 의한 BLDC 모터 위치동기 제어계의 성능 향상

指導敎授 鄭 碩 權

이 論文을 工學碩士 學位論文으로 提出함

2002년 2월

釜慶大學校 大學院

메카트로닉스工學 協同課程

兪 成 權

兪成權의 工學碩士 學位論文을 認准함

2001年 12月 26日

主　　審　工學博士　梁 注 鎬　㊞

委　　員　工學博士　金 相 奉　㊞

委　　員　工學博士　鄭 碩 權　㊞

목 차

Abstract ---------- 1

제1장 서 론 ---------- 3

1.1 연구배경 ---------- 3

1.2 연구목적 및 내용 ---------- 5

제2장 BLDC 모터의 모델링 및 비간섭 제어 ---------- 6

2.1 BLDC 모터의 회로방정식과 상태방정식 ------ 6

2.2 비간섭 제어 ---------- 8

제3장 위치동기 제어계의 설계 ---------- 10

3.1 기존의 위치동기 제어계 ---------- 10

3.1.1 마스터-슬레이브방식 ---------- 10

3.1.2 협조제어 방식 ---------- 10

3.2 하이브리드 위치동기 제어계 ---------- 11

3.3 제어기 설계 ---------- 12

3.3.1 전류 제어기 ---------- 12

3.3.2 단순 PI 속도제어기 ---------- 15

3.3.3 2자유도 PI 속도제어기 ---------- 19

3.3.4 위치동기 제어기 ---------- 21

제4장 수치 시뮬레이션 및 고찰 ---------- 25

4.1 마스터-슬레이브 방식과 협조제어 방식 ---------- 26

4.2 2자유도 PI 제어 ---------- 28

4.3 협조제어방식만에 의한 제어 ---------- 30
4.4 부하외란만을 적극 고려한 방식 ---------- 32
4.5 하이브리드 제어방식 ---------- 32
4.6 양 축에 외란이 인가될 경우의 비교 ---------- 34
4.7 PWM 인버터를 상정한 시뮬레이션 ------- 39

제5장 실험 및 고찰 ---------- 41
5.1 하드웨어의 구성 ---------- 41
5.2 소프트웨어의 구성 ---------- 45
5.3 실험 고찰 ---------- 48

제6장 결 론 ---------- 49

부 록 ---------- 50

감사의 글 ---------- 65

참고문헌 ---------- 66

Improvement of Performance in Position Synchronous Control System of BLDC Motors Using Hybrid Method

Seong-Kwon Yoo

Department of Mechatronics Engineering, Graduate School.
Pukyong National University

Abstract

Brushless DC(BLDC) motors have been widely used in a variety of applications in the automotive industry, industrial automation and consumer electric appliances. The BLDC motor is an synchronous motor with permanent magnets on the rotor and windings on the stator. Compared to induction motors, permanent magnet (PM) motors have higher efficiency due to the elimination of magnetizing current and copper losses in the rotor. It is also easier to achieve high-performance torque control with PM motors compared with induction machines.

This paper presents a precise position synchronous control of two axes rotating system using BLDC motors and a cooperative control based on decoupling technique and PI control law. The system is required performances both good speed following and minimum position synchronous errors simultaneously. To accomplish these goals, the three kinds of controllers are designed. At first, the current and speed controller are designed very simply to compensate the influences of disturbances and to follow up speed references quickly. Especially, the two degree of freedom(2DOF) PI controller is used considering both good tracking for speed reference input and quick rejection of disturbances in speed controller. Finally, a position synchronous controller is designed as a simple proportional controller to minimize position synchronous errors.

Also, In this paper describes a novel precise positon synchronous control using hybrid method The hybrid control is consisted of two different conventional position synchronous control types, a cooperative control and a little modified master-slave control. The cooperative control is adopted to the area within a small position synchronous error(PSE). The master-slave control is especially applied to the area which gives large PSE.

To verify the validity of this suggested method, some simulations were performed using MATLAB.

Through the simulations, following results were obtained.

(1) Comparing with the PI control, the 2DOF PI control showed better properties in the performances of reference tracking and reducing disturbance influence including modelling error.

(2) Through the simulation, the proposed system realizes not only more precise synchronous motion than conventional system ; master-slave control and cooperative control method, but also very stability system.

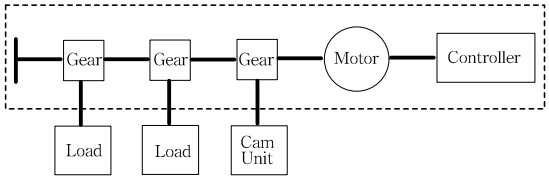
(3) The simulation results using hybrid method show maximum position synchronous error can be reduced approximately 50% comparing to the conventional control type.

Key words : Brushless DC(BLDC) Motor , Two-Degree-of-Freedom(2DOF) Position Synchronous Control(PSC), Position Synchronous Error(PSE), Proportional Integral(PI) Control, Decoupling Method, Cooperative Control, Master-Slave Control, Hybrid Method.

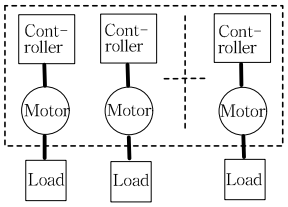
제 1 장
서 론

1.1 연구배경

액튜에이터로서 BLDC 모터는 전자석을 계자극으로 사용하는 경우에 비해서 구조가 간단하고, 대단히 견고하며, 적당한 조건하에서 사용하는 한 모터가 고장을 일으킬 가능성이 적고, 순간기동성이 매우 뛰어나므로 에너지소비가 적다. 특히 장시간 운전으로 인한 로터측(2차측)의 열 발생의 문제점이 없으면서도 DC 모터의 최대 결점인 브러쉬를 갖지 않으므로 서보제어에 이상적이다. 특히, 계자극에 사용되는 자성재료 기술의 발달로 소형 고출력화 또는 최대허용 토크와 정격토크 비율의 증대 등 각각의 특색을 살린 영구자석 동기 모터의 출현 등으로 가격 및 경제적 요소를 가미하여 용도에 적합한 전동기를 선별할 수 있게 됨으로써 향후, AC 서보 모터의 주력모터가 될 것으로 기대되고 있다.



(a) Single-axis driving system



(b) Multi-axis driving system

Fig 1.1 Driving types of position synchronous system between multi axes

최근, 다축간의 위치동기는 Fig 1.1(a)와 같은 집중구동방식에서 Fig 1.1(b)의 분산구동방식으로 이행하고 있다. 그 이유로는 분산구동방식이

기존의 집중구동방식이 갖는 장치의 거대화·복잡화를 피할 수 있고, 설치장소의 제약을 덜 받으며, 운전가동률과 에너지 이용률의 향상을 꾀할 수 있는 장점이 있기 때문이다. 특히, 시스템의 유연성 확보로 CIM(Computer Integrated Manufacturing)시스템 실현이 용이하고 장치의 소형화가 가능한 장점 등을 추가로 들 수 있다.

분산구동에는 축간의 위치동기오차를 슬레이브측 모터만의 속도를 가·감속함으로써 제거하는 마스터-슬레이브(M/S)방식[1~4]과 축 모두의 속도를 동시에 적절히 제어하는 협조제어방식[5,6]이 있다.

또한, M/S 방식에는 마스터 측의 속도를 검출하여 이를 슬레이브 측의 속도입력으로 직접 지령하는 방식과 속도지령에 대한 속도 및 위치동기제어기에 의해 위치동기를 취하는 방식이 있다[1,2]. 전자는 속도의 동기제어는 가능하나 정밀한 위치동기제어에는 부적합하다. 또한 후자는 위치동기만을 적극 고려한 방식으로 주어진 속도지령에 대한 추종성과 정밀한 위치동기를 동시에 요하는 시스템에는 그다지 적합하지 않다. 외란관측기를 사용한 위치동기 제어 방식도 제안되어 있으나, 전류의 피드백 루프가 구성되어 있지 않아 외란 인가시 실 시스템에서의 빠른 보상을 기대하기는 어렵다[3]. 그리고 협조제어방식은 M/S에 비해 특정 축의 속도를 희생시키지 않고 모터의 두 축이 부하를 적절히 분담함으로써 정밀한 위치동기를 실현하는 방법이다. 다만, 외란 인가시 다소 큰 위치동기오차가 발생하며, 이 오차가 결과적으로는 위치동기제어계 전체의 성능을 제약하는 요소로 작용한다. 따라서, 고정도 위치동기제어계를 구현하기 위해서는 전 운전구간에서의 일관된 협조제어방식 보다는 위치동기오차의 크기에 따라 제어전략을 별도로 구사하는 방식이 바람직하다.

한편, 고정도의 위치동기제어는 지령속도에 대한 빠른 추종성과 외란 인가시의 빠른 보상을 동시에 필요로 한다. 이 경우, 단순 PI 속도제어기만으로는 상반하는 두 요구조건을 동시에 실현하기가 용이하지 않다. 따라서, 본 논문에서는 속도제어기를 속도지령에 대한 명령추종성과 부하외란에 대한 외란 제거성을 각각 독립적으로 설계할 수 있는 2자유도 PI 제어칙을 이용하여 설계한다. 그리고, 위치동기오차를 제거하기 위한 위치동기제어기는 시스템의 복잡성을 피하고, 부하분담을 통해 빠른 동기를 취할 수 있는 P 제어칙에 근거한 협조제어방식을 이용한다. 뿐만 아

니라 전류제어기를 포함한 속도제어기는 BLDC 모터의 회로방정식에 기초한 비간섭 제어 수법을 통해 간편하게 설계되며, 특히 각 제어기의 게인설정은 시행착오적인 반복법을 지양하고 제어계의 캐리어주파수에 근거하여 설정되도록 설계된다.

1.2 연구목적 및 내용

본 논문에서는 외란이 인가될 경우에 최대 위치동기오차가 발생하는 것에 주목하여 특정영역에서는 외란이 상대적으로 큰 축의 속도응답을 다른 축의 속도지령으로 하는 M/S방식을 채용하고, 일반영역에서는 협조제어를 취하는 하이브리드형 위치동기제어 방안을 제안한다. 이로써 전 운전구간에서의 일관된 협조제어방식에 비해 위치동기오차 정도를 향상시킬 수 있음을 보인다. 또한, 본 논문에서는 실용상의 적용문제를 고려하여 BLDC 모터를 이용한 위치동기방식을 선택하였다. 뿐만 아니라 단순 PI제어를 지양하고 2자유도 PI를 적용함으로써 외란에 대한 빠른 속도보상과 동시에 명령추종시 오버슈터 발생을 최소화함으로써 위치동기에러가 최소화될 수 있도록 하였다. 이때에도 전류제어기와 속도제어기의 게인은 부하외란의 영향을 적극 고려하여 설계함으로써 위치동기오차가 최소화됨과 동시에 시행착오적 반복 설계를 피할 수 있도록 하였다.

2장에서는 제어칙 설계를 위해 모터에 대한 수치 모델링이 행해진다. 모터에 대한 회로방정식에 근거하여 모터의 전자기적 순시토크와 상태방정식을 유도하고, d-q축 전류를 대응되는 d-q축 전압으로 각각 독립적으로 제어하는 비간섭 제어 이론도 고찰한다.

3장에서는 제어기 설계부분으로 전류제어기, 속도제어기 및 위치동기제어기를 각각 설계하고, 속도 및 위치동기 제어기의 최적의 게인 설정을 통하여 BLDC 모터 위치동기제어계에 적용하는 방법을 기술한다.

4장에서는 위에서 설계된 제어기들을 각각의 제어방식에 접목하고, 다양한 수치시뮬레이션을 통해 상호 비교한다.

5장에서는 실험을 위한 하드웨어 및 소프트웨어 구성을 소개한다.

제 2 장
BLDC 모터의 모델링 및 비간섭 제어

2.1 BLDC 모터의 회로방정식과 상태방정식

3상 원통형 BLDC 모터의 등가모델은 d, q좌표계상에서 일반적으로 Fig 2.1과 같고, 그 회로방정식은 회전자와 동일한 각속도 ω_{re}로 회전하는 식(2.1)과 같이 나타내어진다. 이 좌표계에서는 계자의 자속 방향을 d축, $\pi/2$ 앞선 위상을 q축으로 설정한다.

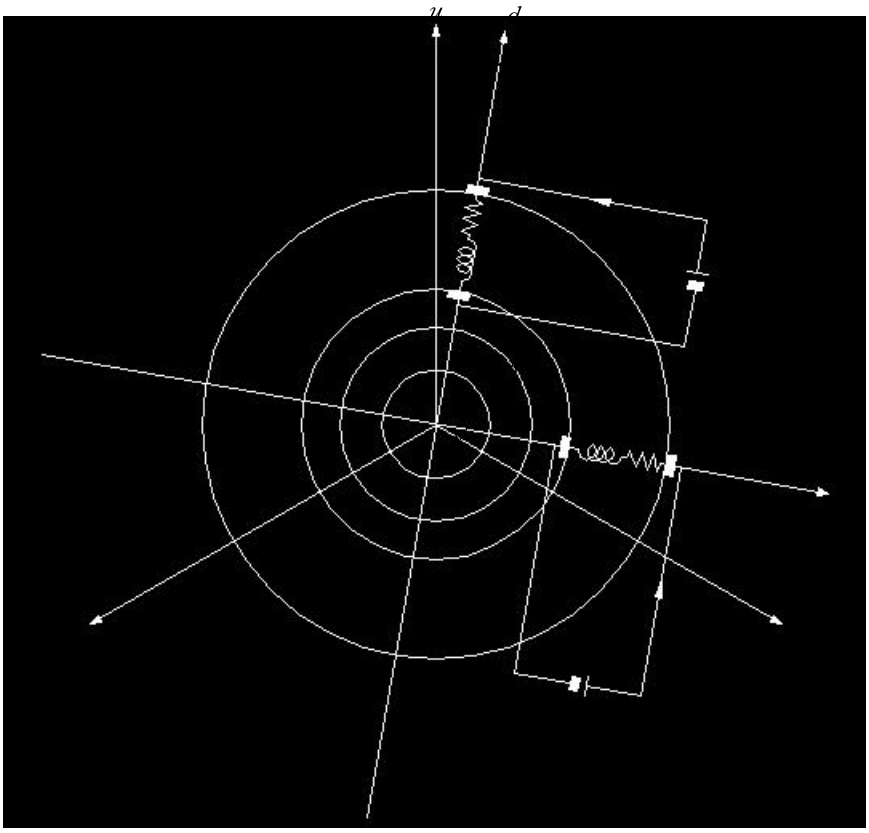


Fig 2.1 d-q equivalent model of 3-phase cylindrical BLDC motor

$$\begin{bmatrix} v_{da} \\ v_{qa} \end{bmatrix} = \begin{bmatrix} R_a + pL_a & -\omega_{re}L_a \\ \omega_{re}L_a & R_a + pL_a \end{bmatrix} \begin{bmatrix} i_{da} \\ i_{qa} \end{bmatrix} + \begin{bmatrix} 0 \\ e_{qa} \end{bmatrix} \tag{2.1}$$

단, $L_a = l_a + \frac{3}{2} M_a'$, $e_{qa} = \omega_{re}\Phi_{fa}$이다.

여기서, v_{da}, v_{qa}는 d, q축상의 전기자 전압, i_{da}, i_{qa} 는 d, q축상의 전기자 전류, l_a는 누설인덕턴스, R_a, L_a, M_a'는 전기자 권선저항, 자기인덕턴스, 권선간 상호인덕턴스를 각각 나타낸다. 또한, Φ_{fa}는 쇄교자속수이다. 여기서, Φ_{fa}'는 계자의 u, v, w상 전기자 권선 쇄교자속수의 최대치이므로, $\Phi_{fa} = \sqrt{3/2}\Phi_{fa}'$의 관계이다. p는 미분연산자, θ_{re}는 u상의 전기자권선을 기준으로 하여 시계방향으로 취한 계자의 전기각, e_{qa}는 q축 전기자권선에 유기되는 속도기전력을 각각 나타낸다.

한편, 모터에서 발생되는 전자기적 순시토크 T_e는 3상에서 2상의 d, q 좌표계로의 좌표변환에 의해서 전기자권선 쇄교자속과 이에 직교하는 전기자전류의 곱으로 식(2.2)와 같이 주어진다.

$$\begin{aligned} T_e &= P\Phi_{fa}'\left\{-i_{ua}\sin\theta_{re} - i_{va}\sin(\theta_{re} - \frac{2}{3}\pi) - i_{wa}\sin(\theta_{re} + \frac{2}{3}\pi)\right\} \\ &= P\Phi_{fa}\{-i_{\alpha a}\sin\theta_{re} + i_{\beta a}\cos\theta_{re}\} \\ &= P\Phi_{fa} i_{qa} \end{aligned} \tag{2.2}$$

여기서, P는 극대수 나타낸다. BLDC 모터에서는 자속이 회전자인 영구자석에 의해 발생되므로, 전기자전류 i_{qa}만을 제어함으로써 순시토크를 제어할 수 있음을 보여준다. 위 식에서 i_{da}는 i_{da}에 의한 모터의 동손이 없고, 최대효율을 얻을 수 있다는 장점 때문에 실제적으로 $i_{da} = 0$이 되도록 제어하는 것이 일반적이다. 식(2.1)의 회로방정식을 상태방정식 형태로 표현하면 식(2.3)과 같이 유도된다. 이는 d, q축의 전압 v_{da}, v_{qa}를 제어입력으로 하여, 상태변수인 전기자전류 i_{da}, i_{qa}를 제어할 수 있음을 나타낸다. 여기서, e_{qa}는 제어가능한 파라미터가 아니다.

$$p\begin{bmatrix} i_{da} \\ i_{qa} \end{bmatrix} = \begin{bmatrix} -\frac{R_a}{L_a} & \omega_{re} \\ -\omega_{re} & -\frac{R_a}{L_a} \end{bmatrix} \begin{bmatrix} i_{da} \\ i_{qa} \end{bmatrix} + \frac{1}{L_a}\begin{bmatrix} v_{da} \\ v_{qa} \end{bmatrix} - \frac{1}{L_a}\begin{bmatrix} 0 \\ e_{qa} \end{bmatrix} \tag{2.3}$$

2.2 비간섭 제어

식(2.3)의 상태방정식은 d, q축 사이에 간섭항인 속도 기전력이 존재한다. 따라서 이 간섭항들을 소거하기 위해 식(2.4)와 같이 간섭항에 해당하는 값을 입력측 전압 지령치에 역 부호로 입력시킴으로써 간섭항을 제거할 수 있다. 식(2.1)로부터 비간섭화된 전압 v'_{da} , v'_{qa} 는 아래와 같이 얻어진다.

$$\begin{aligned} v'_{da} &= v_{da} + \omega_{re} L_a i_{qa} \\ v'_{qa} &= v_{qa} - \omega_{re}(\Phi_{fa} + L_a i_{da}) \end{aligned} \tag{2.4}$$

여기서, ω_{re}, i_{da}, i_{qa}는 실제로 검출가능한 양이고, L_a, Φ_{fa}는 모터 파라미터로서 상수치이므로 식(2.4)의 우변 제2항은 연산이 가능하다. 따라서, 비간섭화된 모터의 상태방정식은 식(2.5)와 같이 간단히 나타낼 수 있다.

$$p\begin{bmatrix} i_{da} \\ i_{qa} \end{bmatrix} = \begin{bmatrix} -\frac{R_a}{L_a} & 0 \\ 0 & -\frac{R_a}{L_a} \end{bmatrix} \begin{bmatrix} i_{da} \\ i_{qa} \end{bmatrix} + \frac{1}{L_a}\begin{bmatrix} v'_{da} \\ v'_{qa} \end{bmatrix} \tag{2.5}$$

이와같이 d축 전압으로 d축 전류를 q축 전압으로 q축 전류를 각각 독립적으로 제어할 수 있도록 상호 간섭항을 제거하는 제어가 비간섭제어이다. Fig 2.2는 식(2.3)에서 얻어지는 블럭선도이다. Fig 2.2에는 토크식 식(2.2)와 기계계에서의 회전운동방정식 식(2.6)이 동시에 표현되어 있다.

$$pJ_M(\omega_{re}/P) = pJ_M\omega_{rm} = T_e - T_L \tag{2.6}$$

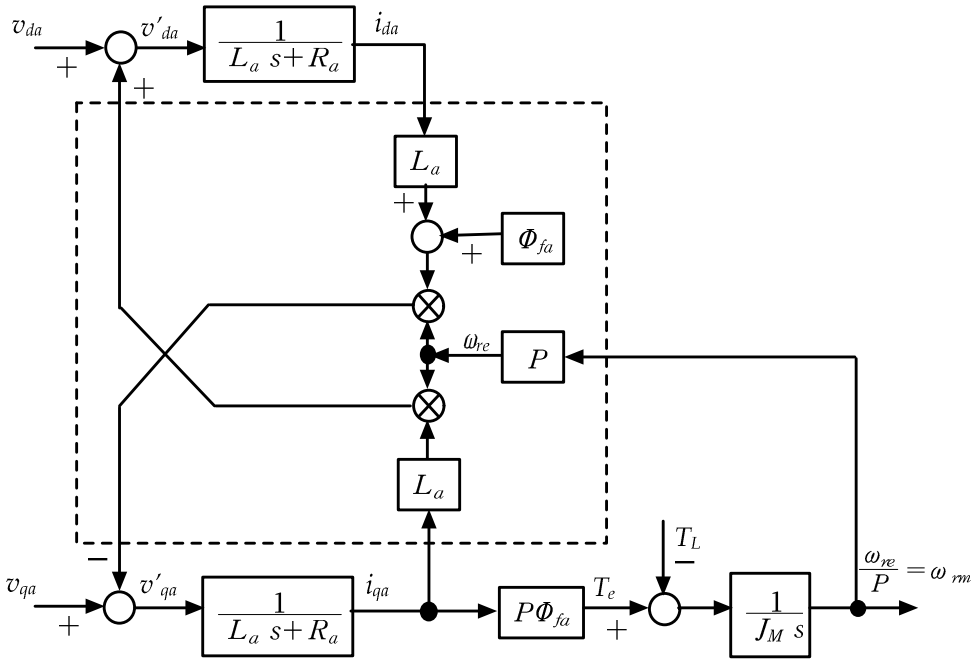


Fig 2.2 Block diagram of BLDC motor in d-q rotating reference frame

그림에서 소문자 s는 복소변수, J_M은 모터의 기계적 회전 관성모멘트를 각각 나타낸다. 마찰계수 D_M은 서보모터의 경우 일반적으로 매우 작은 양이므로 이하의 논의에서 수식전개의 간편성을 위해 생략하였다. Fig 2.3은 식(2.5)를 이용한 제어 블록선도이다. 여기에서 상호 간섭항들이 모두 제거되어 제어계가 훨씬 간단해지고 직관적으로 이해하기 쉽게 되어 있음을 알 수 있다.

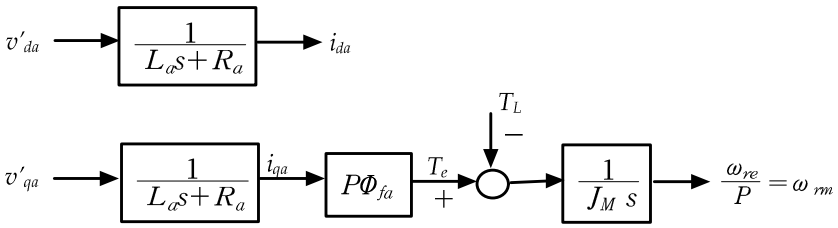


Fig 2.3 Decoupled block diagram of BLDC motor

제 3 장
위치동기 제어계 설계

3.1 기존의 위치동기 제어계

3.1.1 마스터-슬레이브 방식

Fig 3.1은 전형적인 Master-Slave(M/S) 제어방식에 의한 위치동기 시스템을 나타낸다. 이 방식은 두 모터의 응답을 확인하여 응답이 빠른쪽을 슬레이브 모터로 하여 과도상태부터 두 모터의 위치오차를 검출하고 이에 대한 보상은 슬레이브측 모터만의 속도를 가·감속시킴으로써 정밀한 위치동기를 취하는 방식이다.

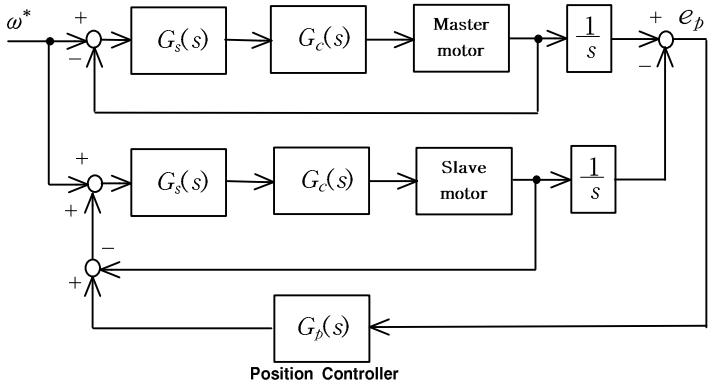


Fig 3.1 Block diagram of the master-slave control method

이 제어방식은 마스터 모터에 들어오는 외란에 대한 대응은 용이하지만, 슬레이브 모터에 들어오는 외란에 대한 대응은 그다지 용이하지 않다. 이러한 문제점을 보완하기 위하여 제안된 가변구조 M/S 제어방식이 있지만, 속도지령에 대한 빠른 추종성과 정밀한 위치동기를 동시에 요구하는 시스템에는 그다지 적합하지 않다.

3.1.2 협조제어 방식

Fig 3.2는 협조제어 방식에 의한 위치동기 시스템을 나타낸다. 이 방식

은 앞서 언급된 M/S 제어방식에 비하여 특정 축의 속도를 희생시키지 않고 모터의 두 축이 부하를 적절히 분담함으로써 정밀한 위치동기를 실현하는 방법이다. 다만, 외란인가시 다소 큰 위치동기오차가 발생하며, 이 오차가 결과적으로는 위치동기제어계 전체의 성능을 제약하는 요소로 작용한다.

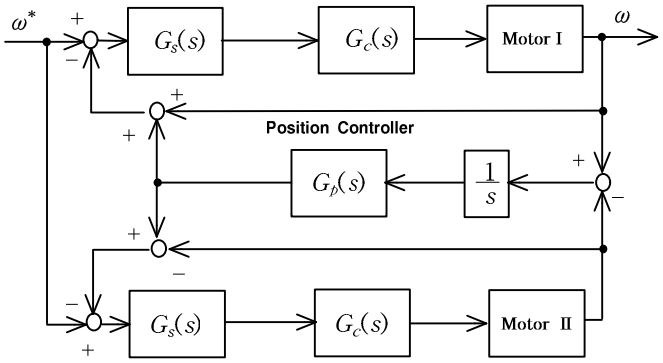


Fig 3.2 Block diagram of the cooperative control method

따라서, 고정도 위치동기 제어계를 구현하기 위해서는 전 운전구간에서의 일관된 제어방식보다는 위치동기오차의 크기에 따라 제어 전략을 별도로 구사하는 방식이 바람직하다.

본 논문에서는 외란인가시 최대 위치동기오차가 발생하는 것에 주목하여 특정영역에서는 외란이 상대적으로 큰 축의 속도응답을 다른 축의 속도지령으로 하는 M/S 방식을 채용하고, 기동시를 포함한 일반영역에서는 협조제어를 취하는 하이브리드형 위치동기제어 방안을 제안한다.

3.2 하이브리드 위치동기 제어계

Fig 3.3은 본 논문에서 제안하는 하이브리드형 위치동기제어(Position Synchronous Control : PSC)계의 구성도로서 협조제어 및 M/S 혼합형을 보인다. 그림에서 System #1, #2는 속도제어기, 모터부를 포함한 전류제어기, 기계계로 구성된다. G_p는 위치제어기, e_p는 속도차의 적분에 의한 위치오차를 각각 나타낸다. 또한, 스위치는 threshold로써, 설정된

범위(값) 이내에서는 input 1을 통과시키고, 설정된 범위 밖에서는 input 3을 통과시킨다. input 2는 설정된 값과 비교되는 값이다. 그리고 load comparison은 양 축의 부하를 비교하여 부하가 상대적으로 큰 축의 속도정보를 다른 축의 속도지령으로 인가하는 역할을 하게된다.

이 방식은 양축의 위치동기화 오차가 설정치를 초과할 경우, 외란이 상대적으로 큰 축의 속도를 다른 축의 속도지령으로 인가하는 M/S제어(점선), 그렇치 않은 경우는 기동시를 포함하여 전 운전영역에서 협조제어(실선)를 취하는 방식이다.

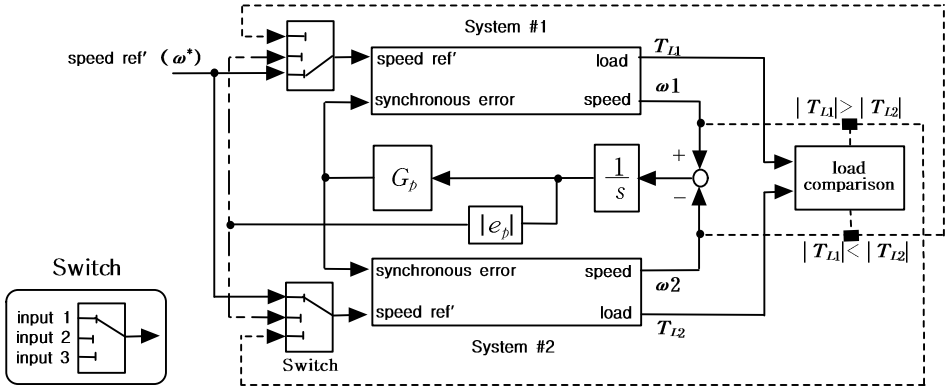


Fig 3.3 PSC system of the hybrid type

이하에서는 각 제어기의 설계과정을 보인다.

3.3 제어기 설계

3.3.1 전류 제어기

Fig 3.4는 하이브리드형 위치동기 시스템의 한 축만을 나타낸다.

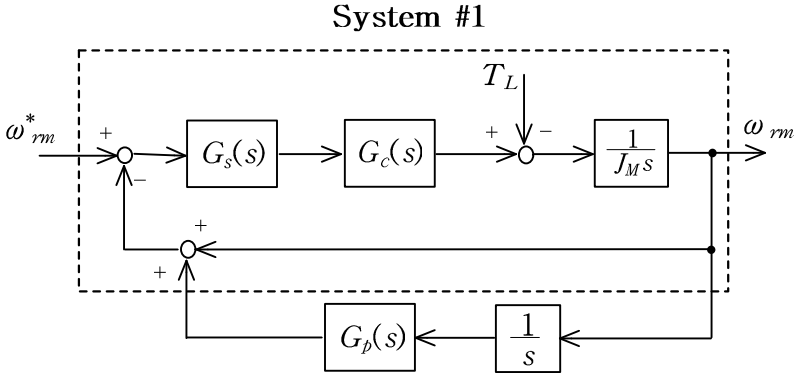


Fig 3.4 Block diagram of one axis in PSC

앞장에서 언급된 비간섭제어 블록도인 Fig 2.3에 전류제어기를 부가하면 Fig 3.5와 같이 된다. 여기서, 전류제어기($G_c(s)$)는 정상편차를 제거하고, 안정성과 속응성을 만족하도록 PI제어기로 설계하면 i_{da}, i_{qa}전류제어기의 출력 v'_{da}, v'_{qa}는 식(3.1)과 같다.

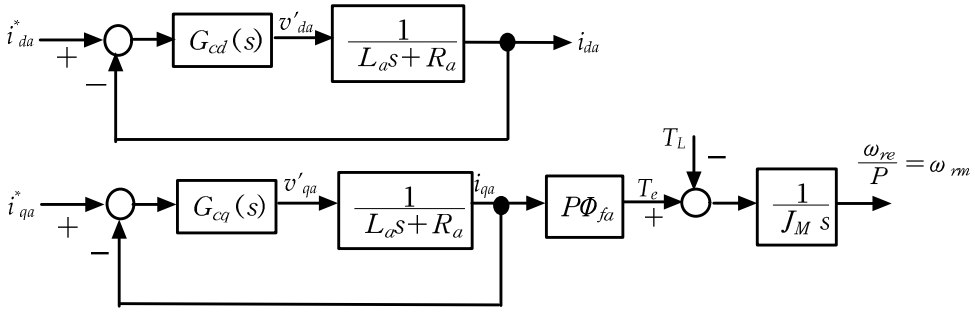


Fig 3.5 Decoupled block diagram of BLDC motor with current feedback controller

$$v'_{da} = K_{id}(1 + \frac{1}{T_{id} s})(i^*_{da} - i_{da})$$
$$v'_{qa} = K_{iq}(1 + \frac{1}{T_{iq} s})(i^*_{qa} - i_{qa}) \quad (3.1)$$

여기서, i_{da}^{*}, i_{qa}^{*}는 제어기의 전류지령치, K_{id}, K_{iq}는 비례게인이며, T_{id}, T_{iq}는 적분시간을 각각 나타낸다. 그리고 전류제어기의 비례게인과 적분시간은 일반적으로 d축과 q축이 같은 값으로 설정된다 ($K_{id} = K_{iq} = K_i$, $T_{id} = T_{iq} = T_i$). 그러므로, PI제어를 행할 경우, 다음의 식과 같이 동일하게 적용할 수 있다.

$$v'_{da} = v'_{qa} = K_i(1 + \frac{1}{T_i s})(i_a^{*} - i_a) \tag{3.1-a}$$

설계된 전류제어계의 개루프 전달함수는 식(3.2)와 같다.

$$G_{cd}^{o} = G_{cq}^{o} = G_c^{o} = K_i(1 + \frac{1}{T_i s}) \cdot \frac{1}{L_a s + R_a} \tag{3.2}$$

여기서, R_a, L_a는 전기자 권선저항, 자기인덕턴스를 각각 나타낸다. 적분시간 T_i는 연산의 편의성을 고려하여 모터계의 시정수로 표현하면 다음과 같다.

$$T_i = L_a / R_a \tag{3.3}$$

이때, 전류제어계의 개루프 전달함수는 식(3.4)와 같이 간단한 1차 적분요소로 된다.

$$G_c^{o} = \frac{1}{(L_a / K_i)s} \tag{3.4}$$

여기서, 전류제어계의 교차각 주파수(이하, 응답 주파수) ω_c는 $\omega_c = K_i / L_a$이므로, 비례게인 K_i는 식(3.5)와 같이 주어진다.

$$K_i = \omega_c L_a \tag{3.5}$$

식(3.5)에서 K_i를 크게하면 ω_c가 증가하여 대역폭이 넓어지게 되므로 속응성을 높일 수 있다. 그러나, 실제의 전류제어계에서는 캐리어 주파수 f_c에 의존하므로 $\omega_c \leq 2\pi f_c / 3$ 정도가 한계이다.

전류제어계의 폐루프 전달함수 $G_{cd}^{c}(s)$, $G_{cq}^{c}(s)$는 식(3.6)로 된다.

$$G_c^c(s) = \frac{1}{(L_a/K_i)s+1} \tag{3.6}$$

이 식의 보드선도는 단순 1차 지연요소로서, 응답주파수에서 게인은 K_i 값과 관계없이 0[dB]이고 정상편차는 0이다. 또한, 위상각은 -90[°]가 한계인 매우 안정한 계이다.

3.3.2 단순 PI 속도 제어기

전류제어계에 PI 속도제어기($G_s(s)$)를 부가하면 Fig 3.6과 같은 속도제어계가 얻어진다. PI제어칙에 의한 속도제어계의 개루프 전달함수는 식(3.7)과 같다. 단, K_{sp}, K_{si} 는 속도제어기의 비례게인 및 적분게인이고, K_T는 토크정수로서 $K_T = P\Phi_{fa}$ 이다. 기계계에 있어 점성마찰 계수는 서보계의 경우 그 값이 매우 작으므로 0으로 간주하였다.

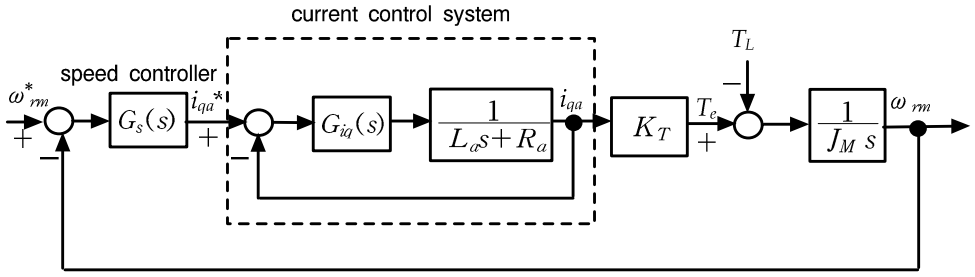


Fig 3.6 Block diagram of speed control system

$$G_{sc}^{o}(s) = (K_{sp} + \frac{K_{si}}{s}) \cdot \frac{1}{\frac{L_a}{K_i}s + 1} \cdot \frac{K_T}{J_M s} \tag{3.7}$$

따라서, 식(3.7)의 주파수 특성은 속도제어기, 모터모델을 포함한 전류제어계 및 기계계 각각의 전달함수로 Fig 3.7과 같이 나타내어진다.

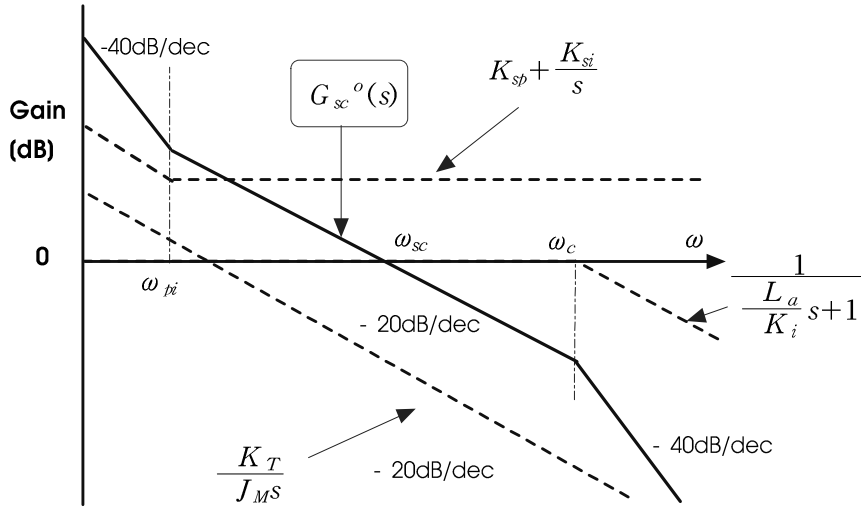


Fig 3.7 Bode plot of PI speed control system

Fig 3.5에서 전체 속도제어계의 응답주파수 ω_{sc} 부근에서는 $G_{sc}^{o}(s)$를 단순 1차 적분요소로 근사화가 가능하게 ω_{sc}와 ω_c사이를 $1/m_1$배 이하로 설계하면 $G_c^c(s) \fallingdotseq 1$로 되어서 전류제어계가 속도응답에 미치는 영향을 무시할 수 있고, 속도제어기의 응답주파수 ω_{pi}는 $\omega_{pi} = K_{si}/K_{sp}$이므로 ω_{pi}와 ω_{sc}사이를 $1/m_2$배 이하로 설계하면, 속도제어계는 식(3.8)과 같이 단순 1차 적분요소로 근사화 된다.

$$| G_{sc}^{o}(j\omega_{sc}) | = \left| K_{sp} \cdot \frac{K_T}{J_M(j\omega_{sc})} \right| \fallingdotseq 1 \tag{3.8}$$

위 식에서 비례게인 값은 식(3.9)와 같이 구해진다.

$$K_{sp} = \frac{J_M \omega_{sc}}{K_T} \tag{3.9}$$

결국, 적절한 m_1, m_2의 선정에 의해서 속도제어기의 게인들이 구해짐을 알 수 있다.

Fig 3.4에서 속도지령 ω^*_{rm}에 대한 전동기 속도응답 ω_{rm}의 전달함수 $G_1(s)$는 식(3.10)과 같다. 단, 간략화를 위해 전류 및 위치제어루프를 무시하며, 기계계의 마찰계수는 0으로 간주한다.

$$G_1(s) = \frac{G_s(s)}{J_M s + G_s(s)} \tag{3.10}$$

또한, 부하토크 T_L에 대한 전동기 속도응답 ω_{rm}의 전달함수 $G_2(s)$는 식(3.11)과 같다.

$$G_2(s) = -\frac{1}{J_M s + G_s(s)} \tag{3.11}$$

PI 속도제어기의 비례게인 K_{sp}및 적분게인 K_{si}는 각각 다음과 같이 설정되어진다[9].

$$\begin{aligned} K_{sp} &= \omega_{sc} \cdot J_M \\ K_{si} &= \omega_{pi} \cdot K_{sp} \end{aligned} \tag{3.12}$$

여기서, ω_{pi}는 속도제어기만의 응답주파수, ω_{sc}는 속도제어계 전체의 응답주파수이다. 따라서, 식(3.12)를 이용하여 K_{sp}, K_{si}를 소거하면 다음 식이 얻어진다.

$$G_1(s)' = \frac{J_M(\omega_{sc}s + \omega_{sc}\omega_{pi})}{J_M(s^2 + \omega_{sc}s + \omega_{sc}\omega_{pi})} \tag{3.13}$$

식(3.13)에서 ω_{pi}의 값을 파라미터로 지령치에 대한 스텝응답을 구하면 Fig 3.8과 같다.

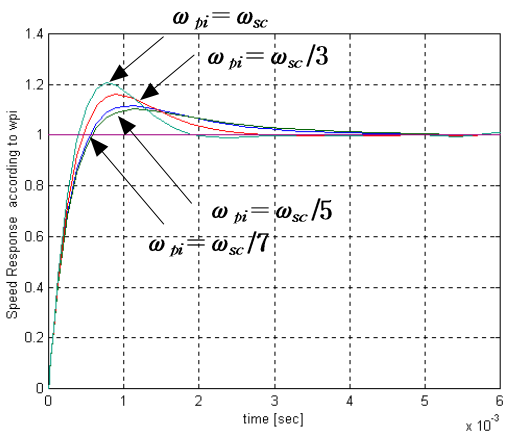


Fig 3.8 Speed responses according to ω_{pi}

Fig 3.8에서 오버슈터를 작게 하면서 동시에 속도의 회복시간을 단축하기 위해서는 ω_{pi}를 ω_{sc}의 5분의1배로 설정하면 좋다는 것을 알 수 있다. 이와 같은 관계는 $\omega_{sc} = \omega_c/5$의 관계에서도 마찬가지였다.

그러나, 실제의 시스템에서는 부하외란 인가시 최대 위치동기오차가 발생하므로, 속도지령에 대한 응답의 특성만을 고려하는 것은 문제가 된다. 따라서, 부하외란에 대한 속도응답을 고려한다면 식(3.14)와 같이 된다.

$$G_2(s)' = -\frac{s}{J_M(s^2 + \omega_{sc}s + \omega_{sc}\omega_{pi})} \tag{3.14}$$

식(3.14)에서 부하토크 T_L에 대한 전동기 속도응답 ω_{rm}의 스텝응답을

구하면 Fig 3.9와 같다. 그림에서 부하토크에 대한 응답의 회복시간을 단축하기 위한 속도제어기의 응답주파수는 식(3.15)과 같이 설정된다.

$$\omega_{pi} = \omega_{sc} / 3 \tag{3.15}$$

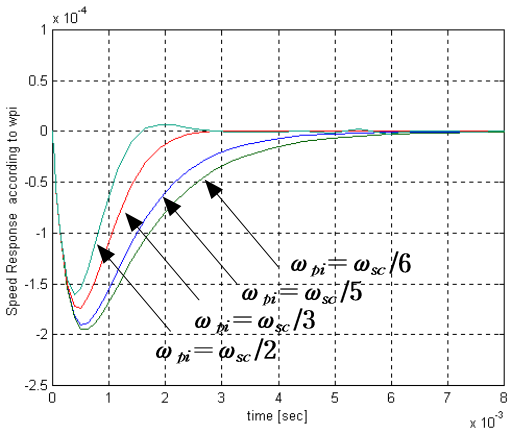


Fig 3.9 Speed responses according to ω_{pi}

이상과 같이하여 앞에서 논의된 적절한 m_1, m_2의 선정이 이루어지게 된다.

3.3.3 2자유도 PI 속도제어기

PI 속도제어계는 속도지령 ω^*_{rm}의 지령에 대한 모터속도 ω_{rm}의 응답의 오버슈트(overshoot)를 작게 하면서도, 부하토크 T_L에 대한 응답의 회복시간을 짧게하는 속도제어를 실현하는 것이 곤란하다. 이 문제를 해결하기 위한 제어방식으로서 Fig 3.10에 보이는 2자유도 PI 속도제어계를 고찰한다. 이 제어계에서의 속도지령 ω^*_{rm} 및 부하토크 T_L에 대한 모터속도 ω_{rm}의 전달함수를 구하면 각각 식(3.16), 식(3.17)과 같다.

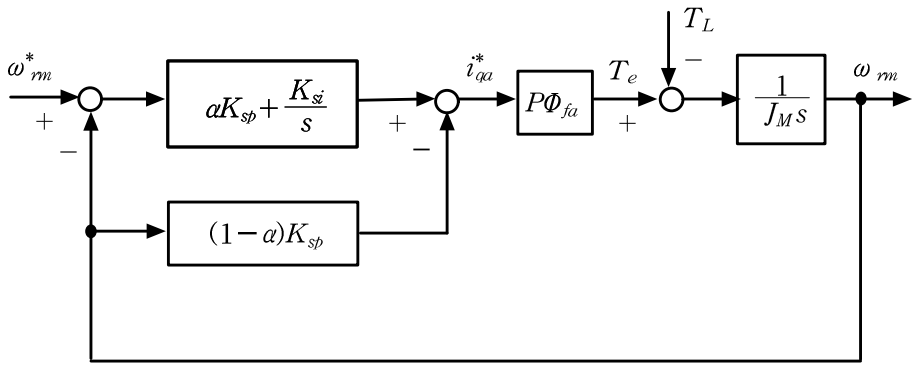


Fig 3.10 Block diagram of 2DOF PI speed control system

$$\frac{\omega_{rm}(s)}{\omega^*_{rm}(s)} = \frac{K_T(\alpha K_{sp}s + K_{si})}{J_M s^2 + K_T K_{sp} s + K_T K_{si}} \tag{3.16}$$

$$\frac{\omega_{rm}(s)}{T_L(s)} = -\frac{s}{J_M s^2 + K_T K_{sp} s + K_T K_{si}} \tag{3.17}$$

식(3.16)에서 속도지령에 대한 속도응답은 게인값이 결정되어도 파라미터 α값의 조절에 의해 변화됨을 알 수 있다.

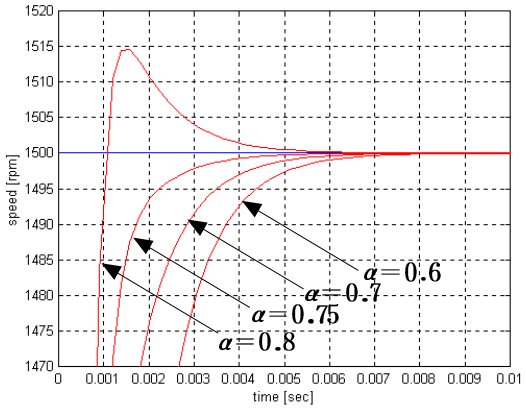


Fig 3.11 Speed responses according to α

Fig 3.11은 실험에 이용된 BLDC 모터의 실제 사양에 대하여 2자유도 PI 속도제어기를 이용하여 1500[rpm]의 스텝지령시 α의 변화에 따른 속도응답을 나타낸다. 이 그림에서 지령치에 오버슈터가 작으면서 빠르게 추종하는 파라미터 값으로 0.75인 α값을 선정할 수 있다.

Fig 3.12에서는 이상과 같이 설계된 2자유도 PI 속도제어계의 주파수 특성을 나타내었다. 그림에서 속도제어계는 오버슈터가 전혀 발생하지 않는 안정한 계임을 알 수 있다.

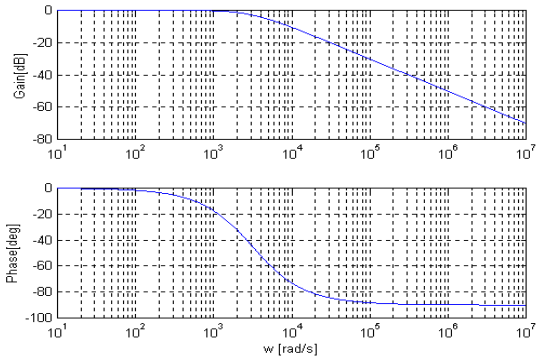


Fig 3.12 Frequency responses of 2DOF PI system

3.3.4 위치동기 제어기

위치제어계에서는 위치에 대한 스텝응답이 오버슈터를 발생하는 것은 바람직하지 않다. 따라서, 앞 절에서 설계한 전류제어기와 속도제어기만으로는 각 모터의 동특성의 차이 및 외란의 영향등으로 인해 발생하는 위치동기오차를 제거하는 것은 용이하지 않기 때문에 Fig 3.3과 같이 위치동기 제어기($G_p(s)$)를 설계한다. 우선, 위치동기제어계에서 기동시인 t_s부터 임의의 시각 t동안 발생되는 위치동기오차의 크기 e_p는 속도오차 e_ω를 이용하면 다음 식으로 표현된다.

$$e_p = \int_{t_s}^{t} |e_{\omega 1} - e_{\omega 2}| dt \tag{3.18}$$

여기서, $e_{\omega j}$에서 j는 1축과 2축을 나타낸다. 정확한 위치동기를 위해서는 기동시부터 전 운전영역에 걸쳐서 미지의 외란 인가시에도 $e_{\omega 1}=e_{\omega 2}$인 것이 요구되고, 이것은 $\omega_1=\omega_2$의 조건과 같다. 결국, 위치동기오차는 두 모터의 속도오차의 적분으로 나타남을 알 수 있다.

$$\omega_j(s)=\frac{K_T(\alpha K_{sp}s+K_{si})}{J_M s^2+K_T K_{sp}s+K_T K_{si}}\,\omega^*(s) - \frac{s}{J_M s^2+K_T K_{sp}s+K_T K_{si}} * T_j(s) \tag{3.19}$$

위치동기 제어기를 갖지 않는 경우의 각 모터의 속도는 식(3.19)와 같이 유도되고, 이 식에서 양축에 D_1/s, D_2/s의 스텝성 외란이 인가되었을 경우의 위치동기오차는 위 식에 최종치 정리를 적용하게 되면 다음 식으로 유도된다.

$$\begin{aligned} e_p(s) &= \lim_{s\to 0} s\frac{1}{s}[\omega_1(s)-\omega_2(s)] \\ &= -\frac{D_1}{K_T K_{s1}}+\frac{D_2}{K_T K_{s2}} \end{aligned} \tag{3.20}$$

식(3.20)에서 외란 D_1, D_2에 의해서 위치동기 오차가 발생하며 그 크기는 1축과 2축에 들어오는 외란의 부호에 따라 위치동기 오차가 감소하거나 증가하는 것을 알 수 있다.

이하에는 위치동기 오차를 최소로 하기 위한 위치동기 제어기를 설계한다. 우선, Fig 3.3에서 위치동기 제어기를 단순 P 제어기로 상정하고, 그 게인값을 1로 하고, 어느 한 축에만 D/s의 크기를 갖는 외란을 인가했을 경우에 최종치 정리를 적용하여 위치동기 오차를 구해보면 식(3.21)과 같이 정상편차가 0임을 알 수 있고, 위치동기 제어기는 전체 제어계의 차수 저감화 차원에서 단순 P 제어기만으로 설계가 가능함을 알 수 있다.

$$e_{ss}(s)=\lim_{s\to 0} s\,e_p(s)\frac{D}{s}=0 \tag{3.21}$$

본 연구에서 위치동기제어기는 과도상태부터 두 모터의 속도오차를 연산하여 위치동기오차로 변환시킨 뒤 이 정보를 다시 속도제어기의 입력정보로 피드백하는 역할을 하게 된다. 또한 위치동기제어계에서는 전류제어계가 부하외란의 영향을 빠르게 보상하므로 위치동기제어기는 비례제어기만으로도 정밀한 위치동기제어를 실현할 수 있게된다. 본 논문에서는 외란인가시 최대 위치동기오차가 발생하는 것에 착안하여, 양 축의 위치동기오차가 0.01[°]의 범위를 벗어날 경우, 외란이 인가되는 축의 속도정보를 다른 축의 속도지령으로 인가하는 M/S방식을 채용한다. 이로써 위치동기오차의 최대 진폭의 크기를 축소할 수 있다. 한편, 이 방식은 기동시를 포함하여 위치동기오차가 0.01[°] 이내의 범위에서는 협조제어를 행하게 되므로 하이브리드 제어방식으로 칭한다.

위치제어기 $G_p(s)$의 설계는 앞서 언급한 것처럼 비례제어기만으로도 충분하다. 따라서, 비례게인 K_{pp}를 설계하면 다음과 같다[9].

$$K_{pp} = \omega_{pc} \tag{3.22}$$

단, ω_{pc}는 위치제어루프의 응답주파수이다.

Fig 3.4에서 위치제어기가 부가되면, 식(3.13)은 식(3.22)를 이용하여 다음 식과 같이 구해진다.

$$G_1(s)'' = \frac{J_m(\omega_{sc}s^2 + \omega_{sc}\omega_{pi}s)}{J_m(s^3 + \omega_{sc}s^2 + \omega_{sc}(\omega_{pi} + \omega_{pc})s + \omega_{sc}\omega_{pi}\omega_{pc})} \tag{3.23}$$

여기서, ω_{pi}는 기지이므로 ω_{pc}의 값을 파라미터로 속도지령 ω^*_{rm}에 대한 전동기 속도 ω_{rm}의 스텝응답을 구하면 Fig 3.13과 같다. 그림에서 속도응답에 오버슈터를 발생하지 않으면서 빠르게 추종하기 위해서는 ω_{pc}가 ω_{sc}의 5분의 1배로 설정되어야함을 알 수 있다. 하지만, 실제의 제어계에서는 제어계 내부의 시간지연등의 이유 때문에 ω_{pc}와 ω_{sc}가 가까울수록 속도응답이 진동적으로 된다. 그러므로 초기치를 5배 정도로 설정

하고 제어계의 응답을 확인하면서 ω_{pc}의 값을 조절할 필요가 있다.

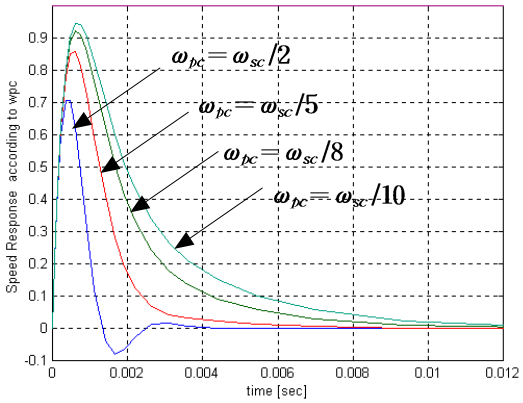


Fig 3.13 Speed responses according to ω_{pc}

제 4 장
수치 시뮬레이션 및 고찰

앞장에서 언급한 모터의 회로방정식 및 상태방정식에 근거하여 설계된 전류제어기, 2자유도 PI 속도제어기와 위치동기제어기를 이용하여 구성한 위치동기제어계의 타당성을 검토하기 위해 2축의 BLDC 모터를 상정하여 시뮬레이션을 행한다.

Table 4.1은 각 제어기의 게인값을 나타낸다. 이것은 앞에서 설계된 각 파라미터 정보에 의해서 각각의 값들이 연산된다. 여기에서는 실제 시스템의 캐리어주파수 f_c를 10[kHz]로 설정하고 시뮬레이션을 수행한다. 또한, 앞서 검토한 바와 같이 ω_{sc}를 기준으로하여 ω_c와 ω_{pi}를 각각 5배로 하였을 경우이며, 위치동기제어기의 비례게인 K_{pp}는 ω_{sc}의 10분의 1 이하의 범위에서 가변시키면서 시뮬레이션을 수행하였다.

Table 4.1 Values of controller parameter

Parameter		Value
Current controller	K_i	366
Speed controller	K_{sp}	0.38
	K_{si}	303
Position synchronous controller	K_{pp}	1~400

Table 4.2는 위치동기제어 시뮬레이션에 사용된 1축과 2축이 같은 사양을 지니는 BLDC 모터 2대의 파라미터 값들을 나타낸다.

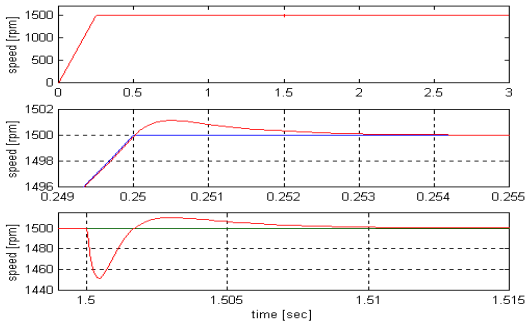
본 논문에서 두 모터의 사양을 동일하게 한 것은 설계상 및 시스템 구축상의 편의를 위한 것이나, 논문에서 제안된 제어계의 설계법은 사양이 각기 다른 모터라 하더라도 그 일반성은 잃지 않는다. 시뮬레이션의 수행은 Matlab simulink 툴을 이용하였다.

Table 4.2 The values of motor parameters

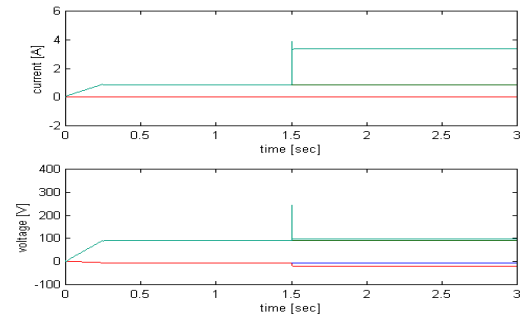
Parameter	Symbol	Unit	Value
정격 출력	P_R	W	300
정격 회전수	N_{rm}	rpm	3000
전기자 저항	R_a	Ω	2.68
전기자 인덕턴스	L_a	H	0.02
쇄교자속수	Φ_{fa}	$V \cdot s/rad$	0.28
극대수	P		2
전기자 관성	J_M	$N \cdot m \cdot s^2/rad$	5.4×10^{-5}
마찰계수	D_M	$N \cdot m \cdot s/rad$	3.3×10^{-6}
정격 토크	T_{eR}	$N \cdot m$	0.95
정격 전기자전류	i_{qaR}	A	1.68

4.1 마스터-슬레이브 방식과 협조제어 방식

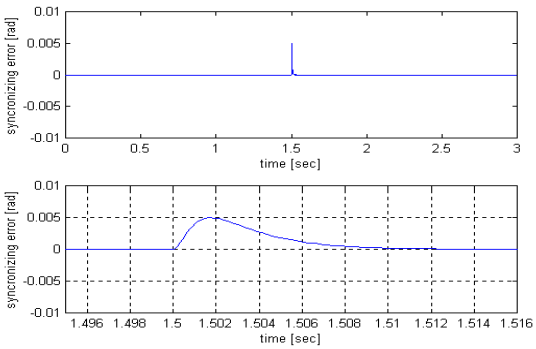
Fig 4.1은 속도제어기는 단순 PI 제어기, 위치동기 제어기의 비례게인 값은 400으로 설정하고, 슬레이브측에 1.5초에 정격의 150%의 스텝상 부하외란을 인가하였을 경우의 마스터-슬레이브 방식에 의한 시뮬레이션 결과들이다. (a)의 속도응답은 초기에 약간의 오버슈터가 발생하고 외란이 인가되었을 경우는 초기보다는 큰 오버슈터와 다소 긴 정착시간을 가지는 것을 알 수 있다. (b)는 전류응답과 전압응답을 나타낸다. (c)는 위치동기오차를 보여준다. 여기서, 위치동기오차가 부하외란 인가시 $5\times10^{-3}[rad]$의 오차가 발생하며, 일정시간이 지난 뒤 0에 수렴하고 있다.



(a) Speed response



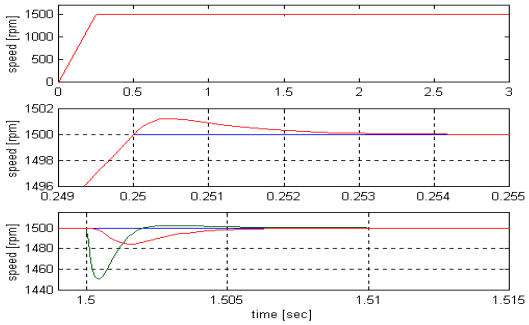
(b) Current and voltage response



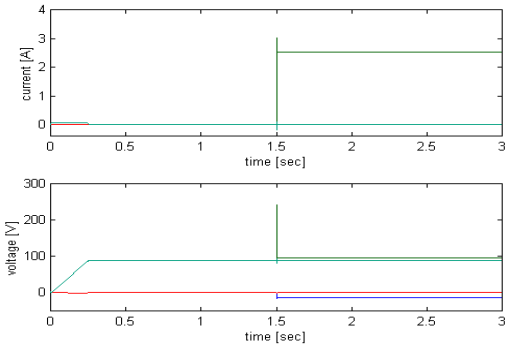
(c) Position synchronous error

Fig 4.1 Simulation using master-slave method

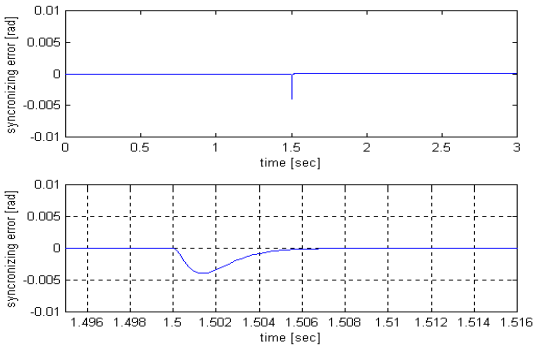
Fig 4.2는 마스터-슬레이브 방법과 조건이 동일한 경우의 협조제어방식에 의한 시뮬레이션 결과를 보여준다. (a)의 속도응답에서 초기에는 그다지 차이가 없지만, 외란이 인가된 후의 속도응답에서는 협조제어방식이 마스터-슬레이브방식보다 작은 오버슈터와 짧은 정착시간을 가지는 것을 알 수 있다. (b)는 전류응답과 전압응답이며, (c)는 부하외란 인가시 $3.6\times10^{-3}[rad]$의 미소한 위치동기오차가 발생하지만, 마스터-슬레이브 방식에 비하여 그 값이 작고, 빠르게 0으로 수렴하고 있음을 알 수 있다.



(a) Speed response



(b) Current and voltage response



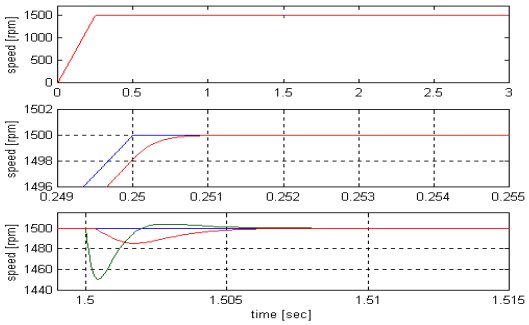
(c) Position synchronous error

Fig 4.2 Simulation using cooperative control method(PI)

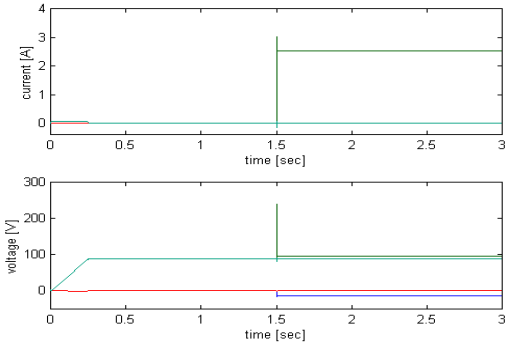
4.2 2자유도 PI 제어

Fig 4.3은 속도제어기가 2자유도 PI 제어기로 설계된 협조제어방식에 의한 시뮬레이션 결과를 나타낸다. (a)는 속도응답에 있어서 다소 지연시간을 가지지만 오버슈터가 전혀 발생하지 않고 단순 PI 제어방식에 비해 정상상태 도달시간이 매우 짧은 양호한 추종성을 보여준다. (b)는 전류응답과 전압응답이며, (c)는 정격의 150%에 상응하는 부하외란 인가시 $3.6\times10^{-3}[rad]$의 위치동기오차가 발생하는 것을 알 수 있고, 이 값은 단순 PI 제어기와 같은 수치이다.

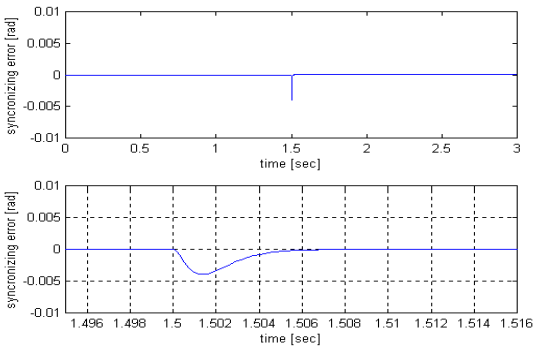
이상의 결과로부터 협조제어방식은 기존의 위치동기제어기를 갖는 M/S방식에 비해 약 28% 정도의 위치동기제어 개선효과를 있음을 알 수 있었다. 그리고, 2자유도 PI 속도제어기는 단순 PI 속도제어기에 비하여 지령속도에 대한 양호한 추종성이 증명되었다.



(a) Speed response



(b) Current and voltage response



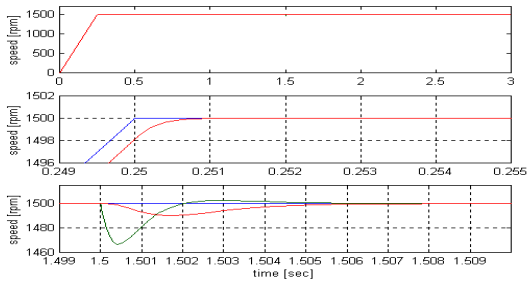
(c) Position synchronous error

Fig 4.3 Simulation using cooperative control method(2DOF PI)

이하에서는 2자유도 PI 속도제어기를 채용하고, 위치동기제어 방식을 기본적으로 협조제어 방식만을 채택하였을 경우, 부하 외란만을 적극 고려한 경우와 제안된 하이브리드 방식과의 비교결과를 나타낸다.

4.3 협조제어 방식만에 의한 제어

Fig 4.4는 부하가변시의 협조제어방식만에 의한 시뮬레이션 결과를 보여준다. 여기서, ω_{sc}를 기준하여 ω_c와 ω_{pi}를 각각 5배로 하였을 경우이며, 2자유도PI 속도제어기의 파라미터 α는 0.75일 경우이다. 그림에서 위는 정격의 100%, 아래는 정격의 200%의 스텝 부하외란 인가시의 속도응답을 나타낸다.



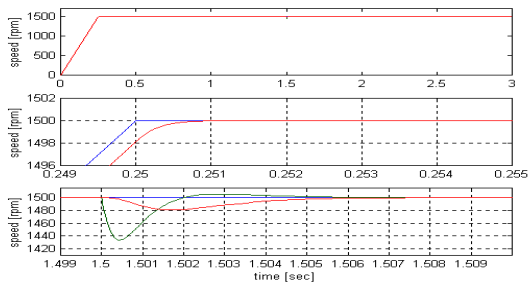


Fig 4.4 Speed responses with variable load in $\omega_{sc}/5$

Fig 4.5는 이때의 위치동기오차를 보여준다. Fig 4.4의 속도응답에 있어 2자유도PI 속도제어기의 영향에 의한 오버슈터는 발생하지 않고, 부하외란 인가시 정격의 100%에서 2.2%, 200%에서 4.53%의 언더슈터를 발생한다. 또한, Fig 4.5에서 위치동기오차가 정격의 100% 부하외란 인가시는 0.143[°], 200%에서는 0.298[°]가 발생되는 것을 알 수 있다.

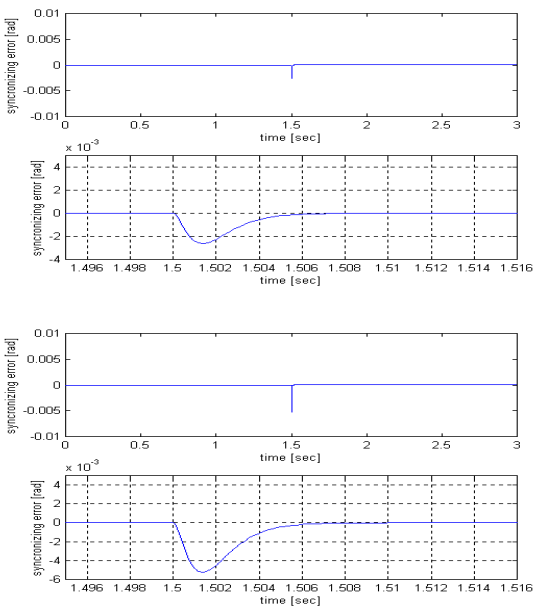


Fig 4.5 PSE with variable load in $\omega_{sc}/5$

4.4 부하외란만을 적극 고려한 방식

앞서 논의한 바와 같이 부하외란 인가시 가장 큰 위치동기오차가 발생하므로 부하외란에 대한 속도응답의 관계로부터 속도제어기의 응답주파수 ω_{pi}는 $\omega_{pi}=\omega_{sc}/3$가 도출되었다. 이하에서는 이 값에 의한 시뮬레이션 결과를 보여준다. 또한, ω_{pi}의 수정으로 속도제어기의 적분 게인값이 식(3.12)에 의해 변하게 된다. 따라서, 2자유도PI 속도제어기의 파라미터 α도 속도응답의 오버슈터를 작게 하면서 속응성을 좋게 하는 값으로 수정되고, 이때 얻어진 값은 0.3이었다. Fig 4.6에서 위치동기오차는 정격의 100% 부하외란 인가시는 0.129[°], 200% 인가시는 0.2636[°]임을 알 수 있다.

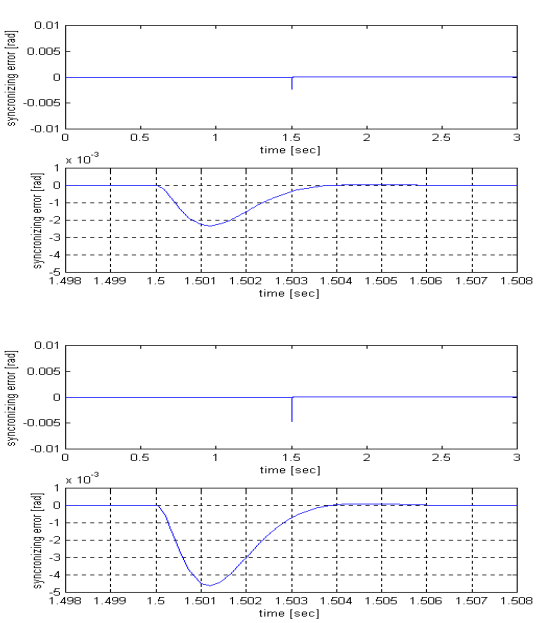


Fig 4.6 PSE with variable load in $\omega_{sc}/3$

4.5 하이브리드 방식

Fig 4.7과 Fig 4.8은 하이브리드 제어방식에 의한 속도응답과 위치동기오차의 시뮬레이션 결과이다.

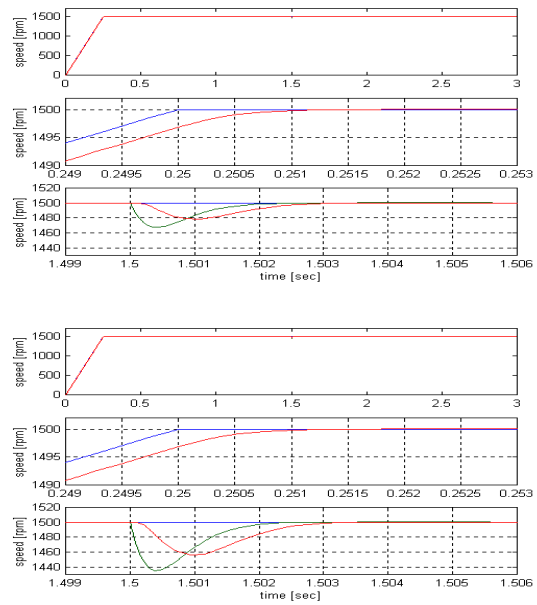


Fig 4.7 Speed responses in hybrid method

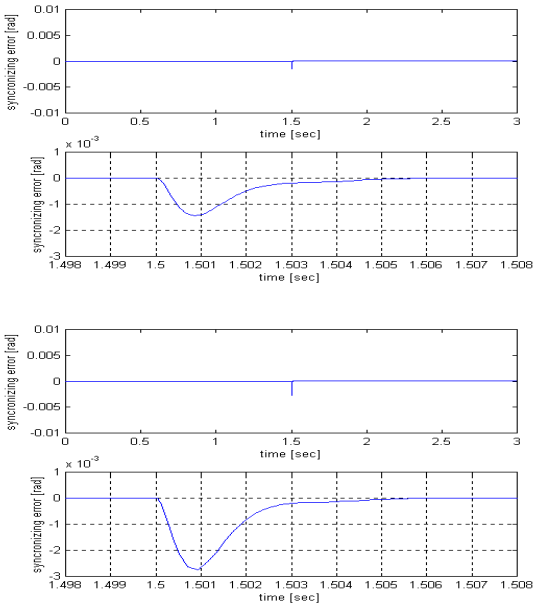


Fig 4.8 PSE in hybrid method

Fig 4.7의 속도응답은 위치동기오차가 설정치를 벗어날 경우, 외란이 인가되지 않은 축이 외란이 인가된 축의 속도를 추종하고 반면에, 설정치 이내에서는 다시 협조제어 방식으로 복귀하는 것을 알 수 있다. Fig 4.8에서는 위치동기오차가 정격의 100% 부하외란 인가시는 0.08[°], 200% 인가시는 0.155[°]임을 알 수 있다. 이것은 하이브리드 방식이 협조제어만을 취한 방식 또는, 부하외란의 영향만을 고려한 방식에 비해 위치동기오차를 최소화할 수 있는 방식임을 보여준다.

Fig 4.9는 이상과 같이 논의된 세 가지 방식에 대한 비교분석 자료이다. 그림에서 (1)은 하이브리드 제어방식, (2)는 부하외란의 영향만을 적극 고려한 방식, (3)은 협조제어 방식이다. 그림은 본 논문에서 제안한 하이브리드 방식이 협조제어 방식에 비해 약 50%의 저감효과가 있음을 나타낸다.

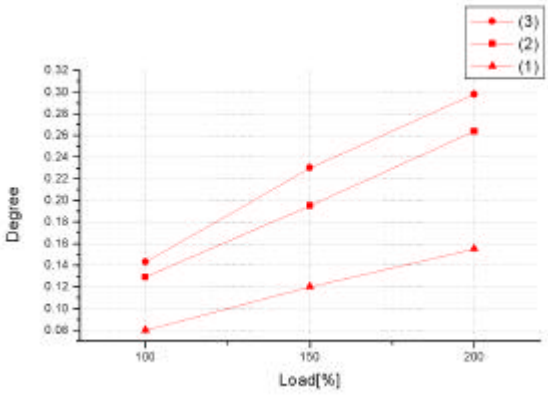


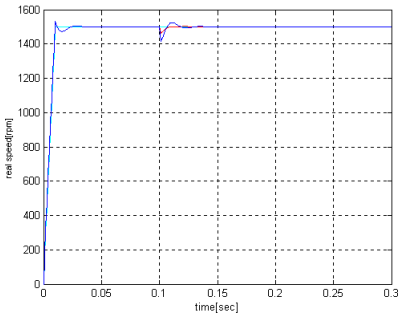
Fig 4.9 Comparison of three method

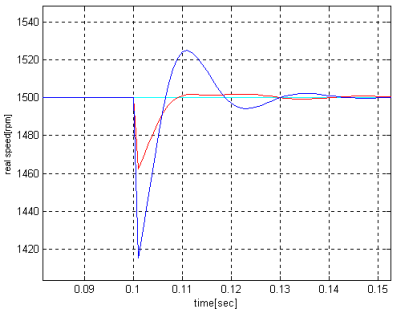
4.6 양축에 외란이 인가될 경우

지금까지는 특정의 한 축에 외란이 인가되었을 경우를 고려한 시뮬레이션을 수행하였다. 이하에는 양 축에 동상(같은부호), 역상(다른부호)의 외란이 각각 들어올 경우를 상정하여 협조제어 방식과 제안된 하이브리드 방식과의 비교 시뮬레이션 결과를 보인다. 시뮬레이션의 수행은 Matlab M-file을 이용하였다.

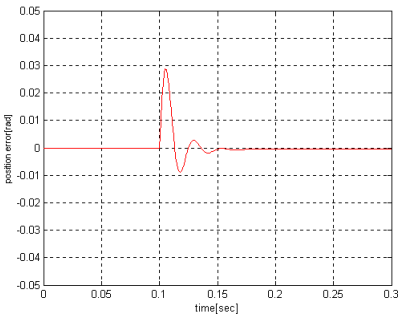
Fig 4.10은 협조제어 방식의 양 축에 동상외란 인가시를 고려한 수치

시뮬레이션 결과이다.

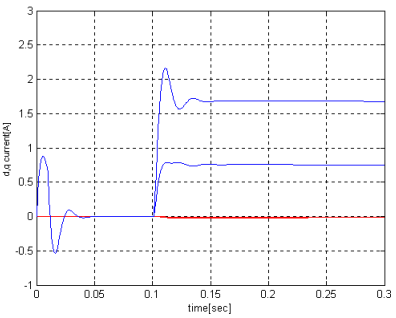




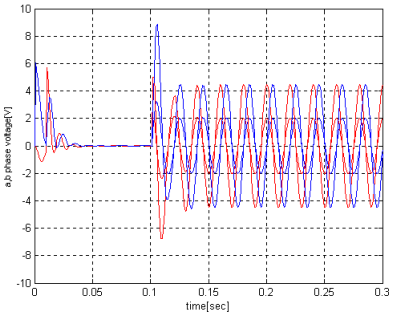
(a) Speed response



(b) PSE



(c) Current response(d, q)

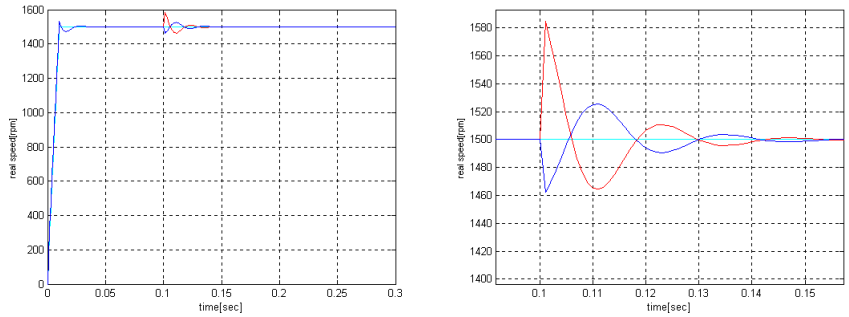


(d) Voltage response(α, β)

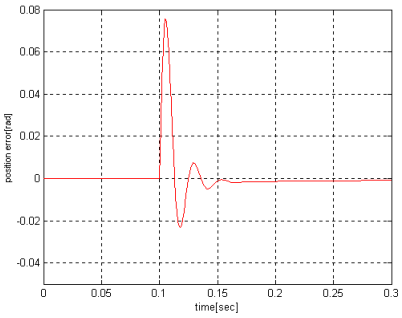
Fig 4.10 Simulation using cooperative control method(In-phase)

그림은 1축에는 정격의 50%인 0.425[N · m], 2축에는 정격의 100%인 0.95[N · m]의 부하외란이 0.1초에 인가되었을 경우의 결과이다. 여기서,

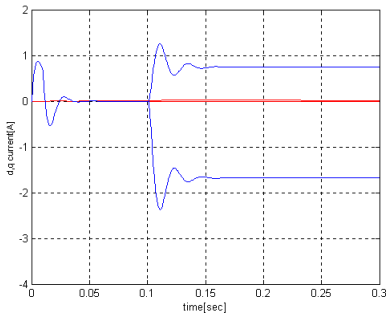
주목하고 있는 위치동기오차는 $2.8\times10^{-2}[rad]$정도 발생되고 있는 것을 알 수 있다.



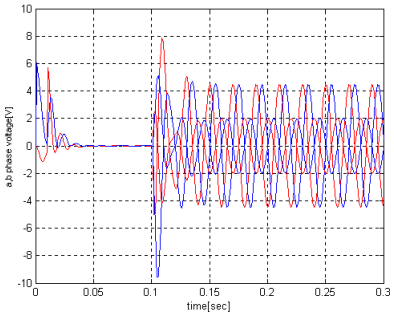
(a) Speed response



(b) PSE



(c) Current response(d, q)

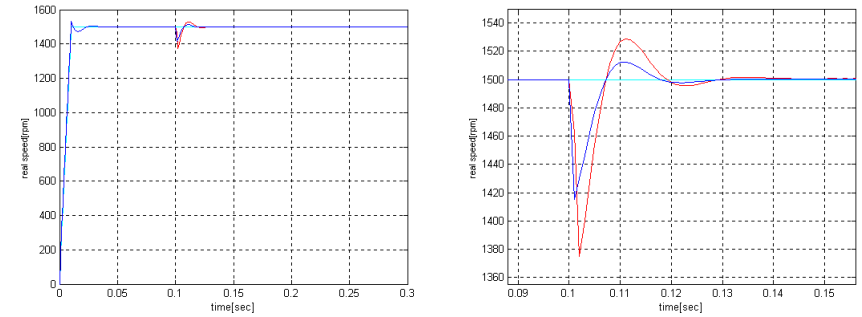


(d) Voltage response(α, β)

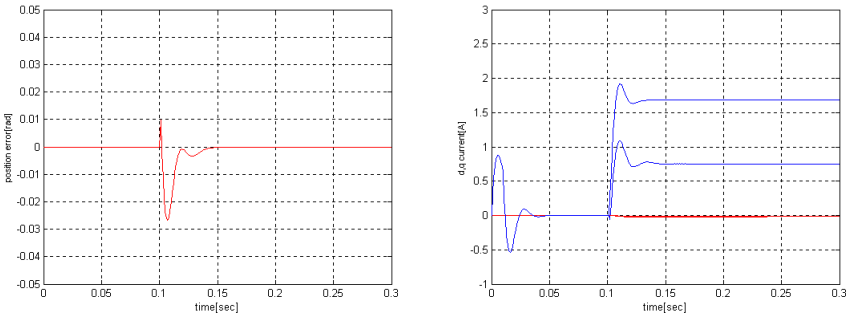
Fig 4.11 Simulation using cooperative control method(Anti-phase)

Fig 4.11은 협조제어 방식의 양 축에 역상외란 인가시를 고려한 수치 시

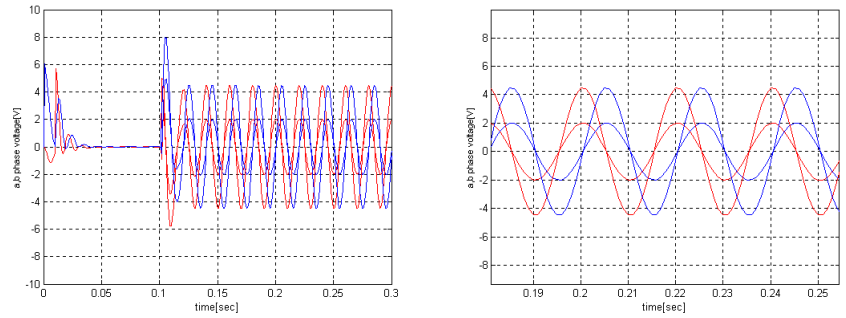
뮬레이션 결과이다. 여기서, 1축에는 -0.95[N · m], 2축에는 0.425[N · m]의 부하외란이 0.1초에 인가되었을 경우의 결과이다. 그림에서 위치동기 오차는 7.6×10^{-2} [*rad*] 이었다.



(a) Speed response



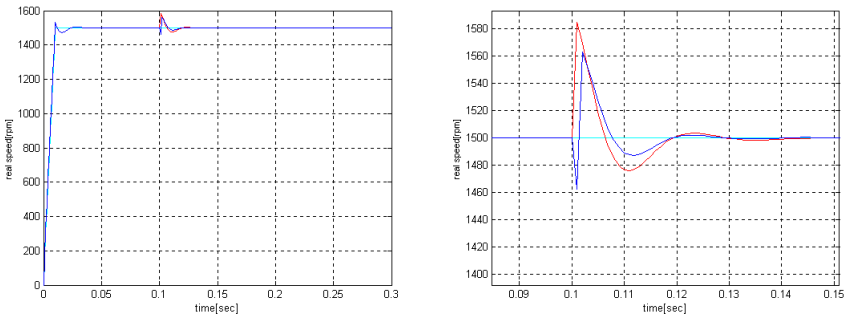
(b) PSE (c) Current response(d, q)



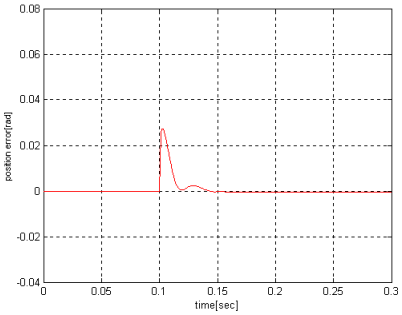
(d) Voltage response(α, β)

Fig 4.12 Simulation using hybrid method(In-phase)

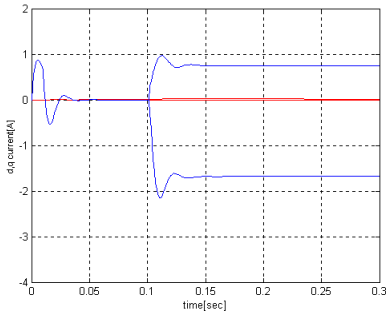
Fig 4.12는 하이브리드 방식의 양 축에 동상외란 인가시를 고려한 수치 시뮬레이션 결과를 나타낸다. 여기서, 위치동기오차는 $2.6\times10^{-2}[rad]$가 발생하며, 협조제어 방식에 비하여 빠르게 0으로 수렴하는 것을 알 수 있고, 전류와 전압 파형도 상대적으로 작다는 것을 알 수 있다.



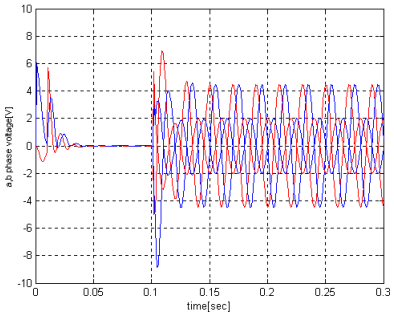
(a) Speed response



(b) PSE



(c) Current response(d, q)



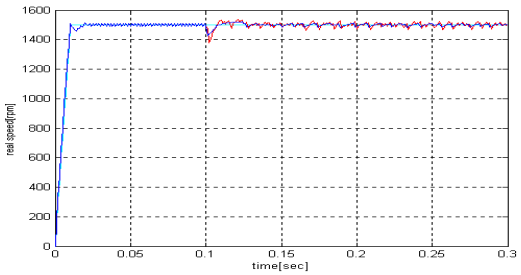
(d) Voltage response(α, β)

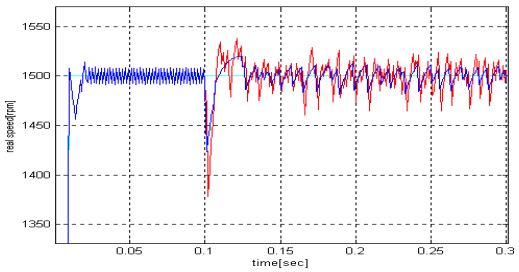
Fig 4.13 Simulation using hybrid method(Anti-phase)

Fig 4.13은 하이브리드 방식의 양 축에 역상외란 인가시를 고려한 수치 시뮬레이션 결과이다. 여기서, 위치동기오차는 $2.7\times10^{-2}[rad]$가 발생하는 것을 알 수 있고, 이 값은 협조제어 방식에 비하여 36%정도의 수치 값이다. 이상의 결과로부터 협조제어방식은 양 축에 동상외란과 역상외란 인가시 위치동기오차의 차이가 심하지만, 제안된 하이브리드 방식은 동상과 역상의 외란인가시에도 크게 차이가 나지 않는 매우 안정한 제어 방식임을 확인할 수 있다.

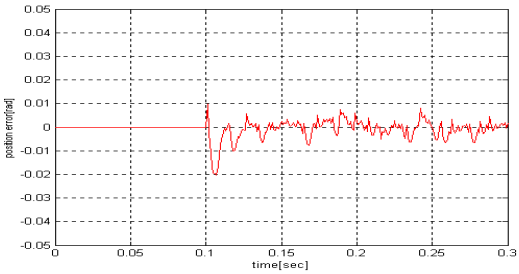
4.7 PWM 인버터를 상정한 시뮬레이션

실제 실험을 고려하여 우선, 동상외란이 시스템에 인가되었을 경우에 대하여 PWM 인버터를 상정한 시뮬레이션을 수행하였다. 여기서, 전류정보는 1[㎲]마다, 전압정보는 100[㎲]마다, 속도정보는 1[㎳]마다 갱신되며, PWM 신호 발생에 사용된 캐리어주파수는 10[kHz]이다. Fig 4.14는 목표치를 0.01초에 1500[rpm]으로 상승하는 램프지령과 0.1초에 1축에는 정격의 50%인 0.425[N · m], 2축에는 정격의 100%인 0.95[N · m]의 부하외란을 인가하였을 경우, 하이브리드 제어방식에 대하여 Matlab M-file로 시뮬레이션 한 결과이다. 2자유도 PI 속도제어기를 사용한 다른 결과들과 비교할 때 속도응답에 오버슈터가 발생하는 것은 설정된 상승시간(Rise time)의 차이와 PWM의 특성에 기인한다. 만약, 상승시간을 동일하게 설정한다면 Fig 4.14의 속도응답은 오버슈터가 거의 발생하지 않게 된다. (a)는 전체 속도응답(上)과 전체응답의 확대된 그림(下)이며, (b)는 위치동기 오차, (c)는 α,β상의 전류응답(上) 및 그때의 주파수 및 위상특성(下)을 보여준다.

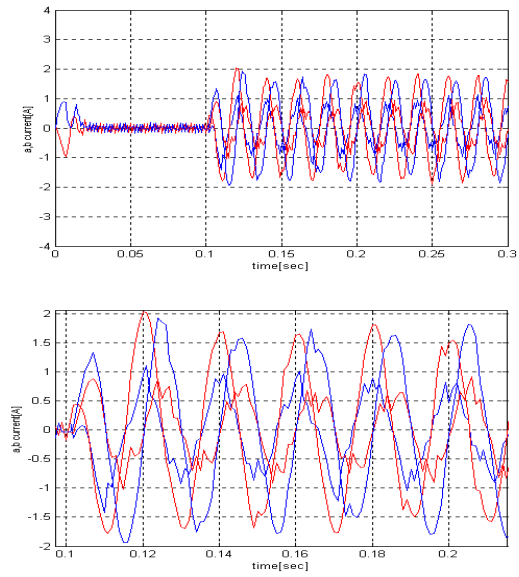




(a) Speed response



(b) PSE



(c) Current response(α , β)

Fig 4.14 Simulation results with PWM

제 5 장
실험 및 고찰

5.1 하드웨어의 구성

실기실험을 위해 사용한 1축과 2축이 동일한 BLDC 모터의 사양을 Table 5.1에 나타내었다.

Table 5.1 Specification of BLDC motor used in experiment

Motor	HA-FF33(Mistubishi)
Rated output	300[W]
Torque	0.95[N · m]
Maximum torque	2.9 [N · m]
Rated speed	3000 [rpm]
Maximum speed	4000 [rpm]
Rated current	1.9 [A]
Maximum current	5.7 [A]
phase-to-phase resist.	5.35[Ω](20℃)
phase-to-phase inductance	24.4 [mH]
torque constant	0.57 [N · m/A] (5.8 kgf · cm/A]
moment of inertia	0.537 [kg · cm^2] (2.0 [kgf · cm · sec^2])
mechanical time const.	1.2 [ms]
electrical time const.	4.6 [ms]
isolated class(field wind.)	B class isolated
isolated limit(field winding)	1500[V]. during 1 min
isolated resi.(field winding)	more than 100[MΩ]
connection of field winding	Y - connection
encoder	8192 Pulse/revolution

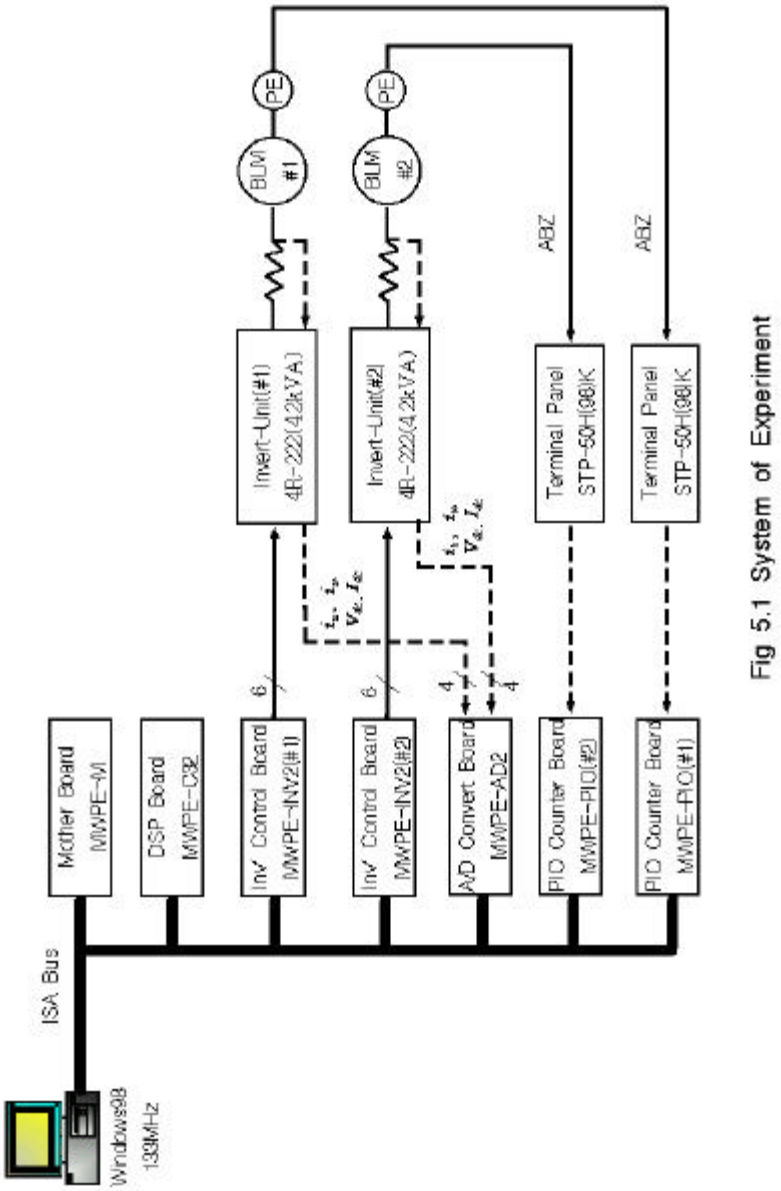


Fig 5.1 System of Experiment

모터구동에 사용된 인버터 및 인버터 제어보드는 DSP를 베이스로 한 Fig 5.1의 시스템을 사용하였다. 이 시스템은 제어를 위해 필요한 각종 보드를 모듈화하여 사용자가 제어시스템을 쉽게 구축하도록 설계되어 있

는 것이 특징이다. 하드웨어의 중요사항은 다음과 같다.

① 마더보드

- 파형표시에 필요한 데이터를 DSP보드에서 얻어 DualPort 메모리를 통해 PC에 전송하며, PC에서는 파형표시 프로그램상에서 실시간으로 데이터의 파형을 관측할 수 있다. DSP보드를 포함한 전체의 시스템을 관리하며, 제어파라미터의 인도, 데이터의 업로드 등을 실행한다.

② DSP보드

- Texas Instrument(T.I.)사의 DSP TMS320C32-50MHz를 이용하고 있다. 부동소수점 연산방식이며, 연산속도는 1명령 실행당 40[ns]이다.

③ 인버터제어보드

- 인버터제어에 필요한 게이트 신호를 생성한다. 기본적으로 전압형 인버터용으로써, 공간벡터변조, 삼각파변조, 게이트 신호 직접출력의 변조방법 중 하나를 선택할 수 있도록 되어 있으나 본 실험에서는 삼각파 변조방식을 이용하였다. PWM에는 f=20[MHz]의 카운터가 쓰이며 삼각파 변조의 경우 반송파(carrier)주파수 f_s는 20[kHz]까지 설정 가능하나 본 실험에서는 10[kHz]로 하였다. 한편, 스위칭 소자의 양단 암(arm)간의 단락방지를 위해 데드타임(dead time)은 4.0[μs]로 설정했다. 이 제어보드는 PWM의 반송파에 동기하여 인터럽트루틴을 수행할 수 있도록 되어있다. 제어보드에서 스위칭 소자에로의 게이트신호출력에는 optical fiber를 사용하였으며, BLDC 모터 2대의 위치동기제어를 위하여 하드웨어적으로 BD0과 BD1으로 구분하여 사용하였다.

③ AD보드

- AD7862가 4개 장착되어 8ch까지 입력이 가능하며, 분해능은 12bit이다. 4ch씩 동기하여 샘플링과 변환을 한다. 각 채널은 Idc, vdc, iw, iu로 변환되고, 이 값들을 모터의 제어 알고리즘에 이용하였다. AD의 변환시간은 3.2[μs]가 소요된다. 따라서, 프로그램 작성시에는 이 시간이 안전하게 확보될 수 있도록 필요한 대기시간을 강제로 설정하였다. 또한, 이 보드의 출력전압의 오프셋(offset)치는 20[mV]이하이다.

④ PIO 카운터보드

- 주로 엔코더에서 나오는 신호를 카운트하는 기능과 디지털 입출력 기능을 한다. 펄스를 얻기위한 인가전압의 형태로서 open-collector, line

driver, TTL레벨입력에 각각 대응하도록 되어있다. 본 실험에서는 open-collector 입력방식을 사용하였다. 엔코더로부터는 ABZ, 또는 ABZUVW상의 펄스정보를 4체배하여 읽어오며 카운트 적산치는 U상 펄스의 상승 에지(edge)에 동기하여 리셋(reset)된다. 본 실험에 사용된 엔코더는 기계적으로 1회전시 8192 pulse를 발생하므로 카운터보드는 1회전당 32768 pulse를 입력받는다. BLDC 모터 2대의 위치동기 제어를 위하여 하드웨어적으로 BD0과 BD1으로 구분하여 사용하였다.

PC를 통해 속도지령을 주면 DSP보드에서 프로그램에 따라 속도오차 및 전류오차를 연산하여 이를 제거하기 위한 3상 전압 지령치를 구하고 인버터제어보드를 통해 인버터모듈에 6개의 optical fiber로 게이트신호를 인가한다. 인버터에서 출력된 3상 전압에 의해 모터는 회전하고, 이때 검출된 3상 전류 등은 AD보드를 통해 DSP보드로 샘플링되어 제어연산에 사용된다. 엔코더에서 출력되는 펄스는 PIO카운터보드를 통해 전기각 및 기계 각속도로 연산되고, DSP보드에서 좌표변환 및 속도제어기에 각각 사용된다.

Fig 5.2는 인버터 유닛의 기능블록도로서 주회로부, PT(Pulse Transformer) 전압검출부, 센서 모니터부, 게이트 드라이브부, 보호회로부 및 제어전원부로 구성되어 있다. 주회로부에서는 R, S, T 단자에서 입력된 3상교류전원을 V_{dc}로 정류하여 IPM(Intelligent Power Module)에 인가하고, 게이트 드라이브부를 통해 입력된 게이트신호에 따라 IPM에서 PWM 3상전압이 발생, U, V, W 단자를 통해 모터에 전압이 인가되게 된다. PT 전압검출부는 3상전압신호를 검출하고, 센서모니터부는 주회로부 센서와 PT 전압검출부에서 검출한 값들을 동축 커넥터로 출력한다. 보호회로부는 V_{dc}전압이나 I_{dc}전류가 설정치를 초과할 때 IPM의 게이트 드라이브를 강제로 정지시킨다.

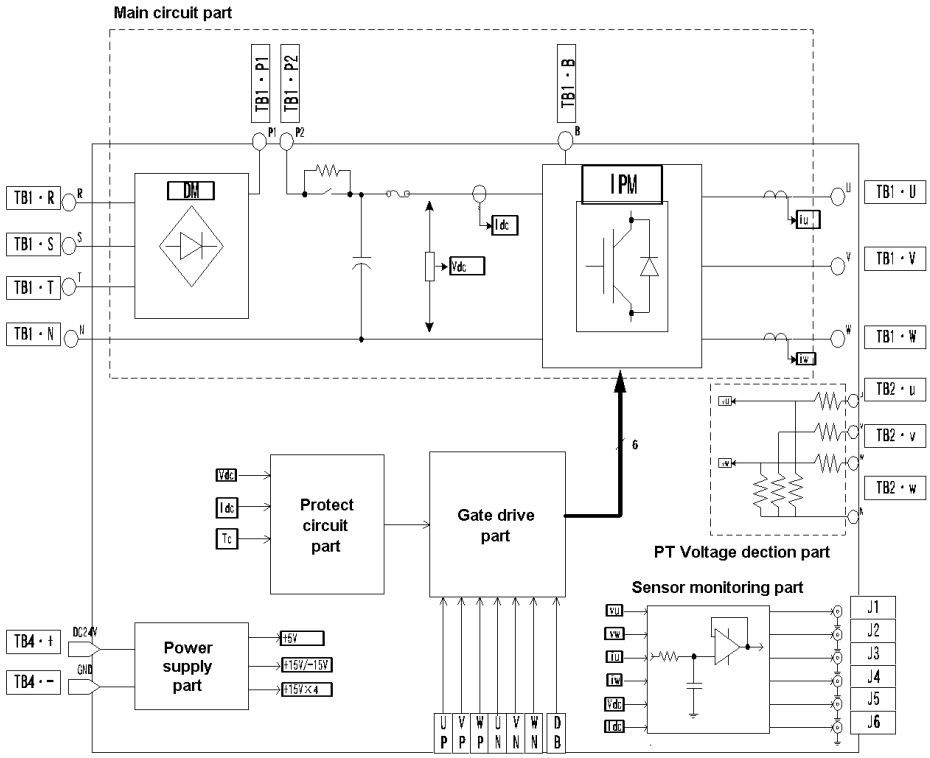


Fig 5.2 Function block diagram of inverter

5.2 소프트웨어의 구성

BLDC 모터 위치동기를 위한 제어블록선도는 Fig 5.3에, 제어 프로그램의 순서도는 Fig 5.4에 각각 보인다. 프로그램은 크게 3부분으로서 main 루틴, carrier 동기 인터럽트 루틴, timer 인터럽트 루틴으로 구성되어 있다. 이하에서는 각 루틴별 주요사항에 대해서 간략히 기술한다.

① main 루틴

- 프로그램에 사용되는 정수치 설정 및 변수정의, 속도지령치, 샘플링 주기, AD range 등을 정의하고, 변수, 보드(BD0, BD1)의 초기화, 타이머 인터럽트(timer0, timer1), 캐리어 인터럽트(INT1, INT2) 동작의 정의를 행한다. 인터럽트 시간외에는 무한루프를 통해 인터럽트루틴에서 연산된 변수 값들을 파형표시 프로그램에서 검출되도록 한다.

② 캐리어 동기 인터럽트 루틴

- 인버터제어보드의 PWM carrier에 동기하여 인터럽트 동작을 하며, carrier 주파수를 10[kHz]로 지정하였으므로 100[μs]마다 인터럽트동작을 하게 된다. 카운터 입력치로부터 계산한 전기각을 이용하여, AD보드로 입력된 모터의 3상전류를 좌표변환 한 다음 전류제어를 행하고 구해진 전압지령을 다시 좌표변환을 통해 3상 PWM 신호로 출력한다.

③ 타이머 인터럽트 루틴

- 설정된 타이머 주기에 의해 인터럽트되는 루틴이다. PIO카운터 입력치로부터 속도응답을 계산하고, 속도제어를 통해 q축 전류지령치를 연산하고, 양 축의 위치동기 오차를 연산한다. 속도정보는 전류정보에 비해 속응성이 그다지 요구되지 않기 때문에 타이머 주기를 1[ms]로 설정하여 1[ms] 주기로 인터럽트 되도록 하였다.

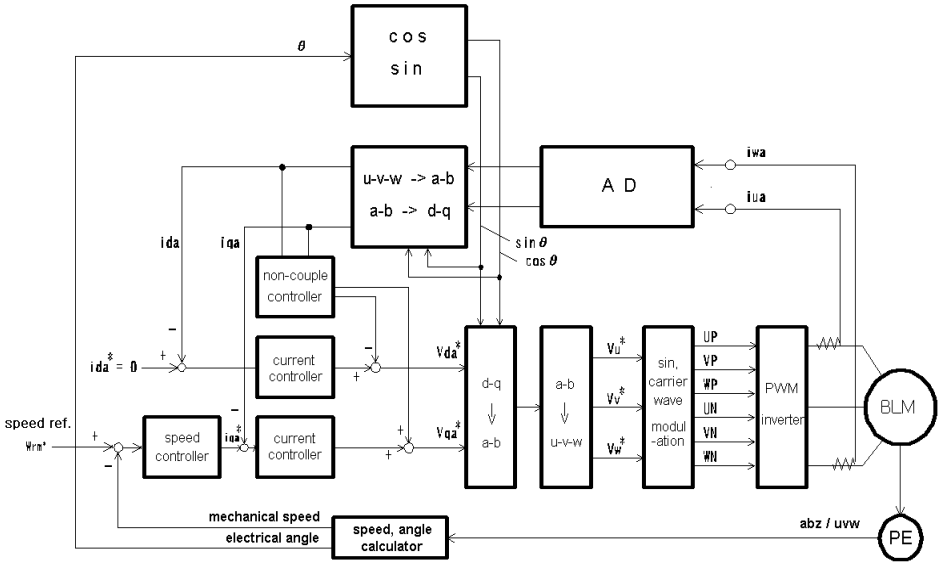


Fig 5.3 Block diagram of BLDC motor control

제어 알고리즘은 C언어를 사용하였으며, 컴파일러로서는 T.I. 사의 "TMS320 Floating-Point DSP Optimizing C Compiler Ver 4.70"을 사용하였다.

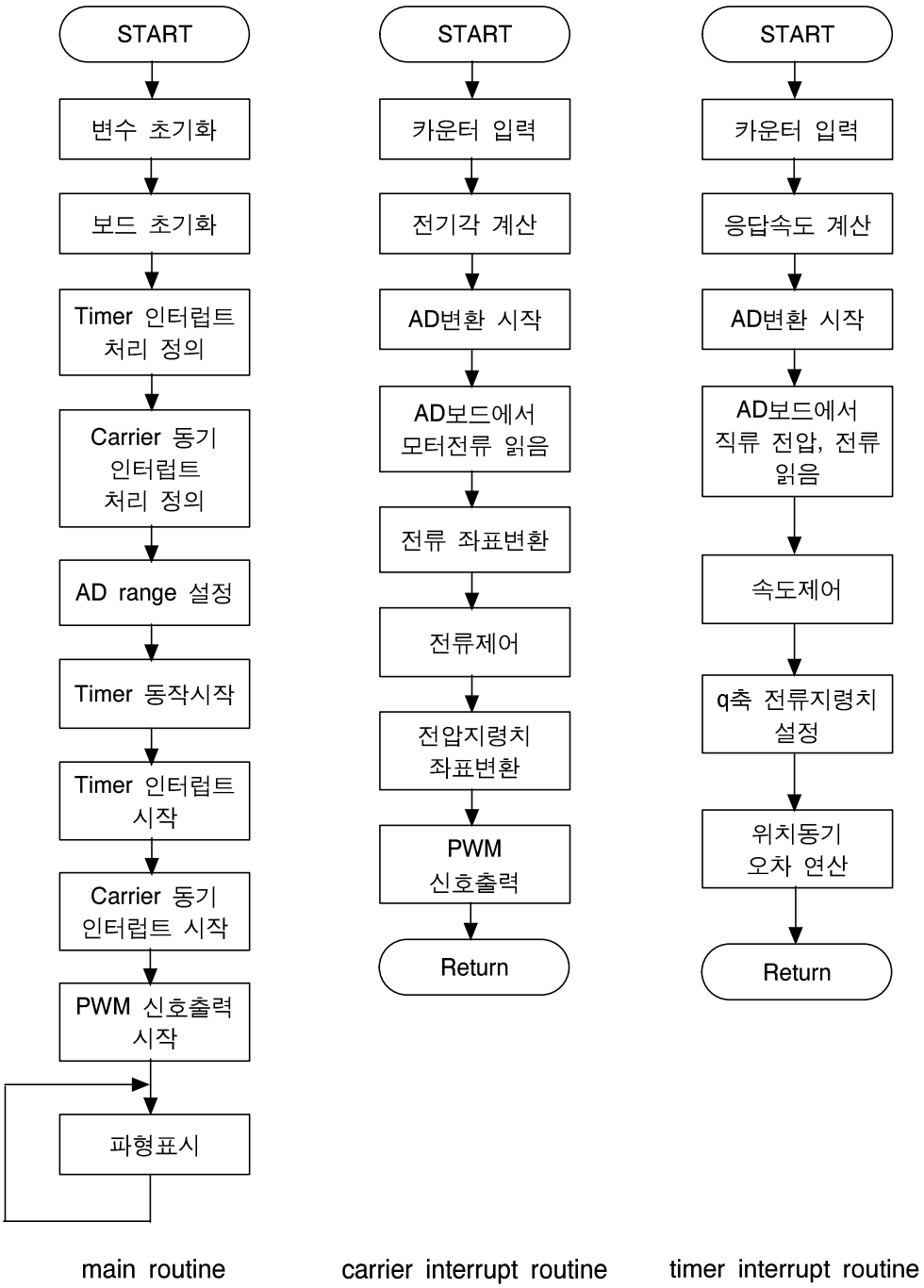


Fig 5.4 Flow chart of control program

5.3 실험 고찰

Fig 5.5 Experiment Equipment

Fig 5.5는 2축 위치동기 제어계의 실제 실험에 이용한 DSP, 위치동기용 BLDC 모터, 인버터 유닛 장치 등을 각각 보여주는 그림이다.

실험에 다소 지연이 발생한 원인으로써, BLDC 모터의 기동에 있어 초기 자극위치 정보는 필수적이다. 이것은 엔코더의 정보(인크리멘탈형/앱솔루트형)에 의해서 A상, B상, Z상 정보가 얻어지게 되고, 이러한 펄스 정보를 가지고 프로그램으로 구현하여 모터의 초기 자극위치를 추정할 수 있게된다. 하지만, 현재 사용중인 엔코더는 엔코더 기술의 발달로 인하여 앰프를 부착하여야만 그러한 정보를 얻게 되는 맹점이 존재한다. 이것은 엔코더와 앰프와의 동기화를 통하여 인크리멘탈형과 앱솔루트형을 구분하게 되고 앰프측 신호를 파악함으로써, 각 펄스의 파형을 얻게 되는 것이다.

이상과 같은 문제점으로 인하여, 실험에 다소 지연이 발생함으로써 실험 데이터를 얻는 것이 불가능하였다.

제 6 장
결 론

본 논문에서는 위치동기 제어계의 성능향상을 도모하고자 하이브리드형 위치동기 제어법을 제안하였다. 연구의 실용적인 측면을 고려하여 대상 모터를 BLDC로 선정하였으며, 각 제어방법들과의 비교 시뮬레이션을 통하여 얻은 결과는 다음과 같다.

(1) 2자유도 PI 속도제어기를 통해 오버슈터가 작고, 부하외란의 영향을 고려한 게인 설정을 통해 위치동기오차를 줄이는 것이 가능하였다.

(2) 제안된 하이브리드 제어방식은 기존의 협조제어 방식에 비해 위치동기오차를 약 50% 저감할 수 있었다.

(3) 하이브리드 방식은 역상의 외란인가시에도 위치동기오차가 크게 변화지 않는 안정한 시스템임을 확인하였다.

향후, 엔코더와 관련한 문제점을 해결하기 위하여 전문 기업체에 의뢰하여, 현 BLDC 모터측에 부착된 엔코더를 제거하고, 인크리멘탈형 1000[pulse/revolution]인 보급형의 엔코더를 부착하여 제안된 하이브리드 제어법의 우수성에 관한 검토가 이루어져야 할 것으로 사료된다.

부 록

A. Matlab M-file Simulation Program

```
**************************************************************************************************
% 2DOF PI Control (decoupling)M-File
% Non-PWM method
**************************************************************************************************
function[y] = twopim( )
% Variation Initalizer ============================

fs=10000.0;                          %sampling frequency
vdc_ref=160.0;
freq=50.0;
iq_refmax=1.68;
Tm_max=0.95;
%motor parameters
Ra=2.675;
La=0.0183;
Phi=0.283635;
pole=2.0;
Jm=0.0000537;
Dm=0.003024;
a=0.3;
Pr=300.0;
nrpm=3000.0;
%controllers gain
Kcp=75.6;
Kci=366.0;                           %La*Wc
Ksp=0.3787;                          %Jm*Wsc/Kt;
Ksi=302.925;                         %Ksp*Wpi;
kpp=50.0;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
c_1=0.0;
c_2=0.0;
sum_1=0.0;
sum_2=0.0;
e_wre=0.0;
e_theta_1=0.0;
c_theta=e_theta_1;
% 1-axis System Initalizer ==============================
wref=0.0;
wrm_1=wref;
wref_0_1=pole*wref*2.0*pi/60.0;
wre_0_1=wref_0_1;
vq_0=wre_0_1*Phi/vdc_ref;
iq_1=iq_refmax*wref/nrpm;
```

```
vd_0=-wre_0_1*La*iq_1;
id_1=0.0;
zero=0.0;
Tl_1=0.0;
seta_1=0.0;
s_eid_1=0.0;
s_eiq_1=0.0;
s_ew_1=0.0;

% 2-axis System Initalizer ============================
wref=0.0;
wrm_2=wref;
wref_0_2=pole*wref*2.0*pi/60.0;
wre_0_2=wref_0_2;
vq_0=wre_0_2*Phi/vdc_ref;
iq_2=iq_refmax*wref/nrpm;
vd_0=-wre_0_2*La*iq_2;
id_2=0.0;
zero=0.0;
Tl_2=0.0;
seta_2=0.0;
s_eid_2=0.0;
s_eiq_2=0.0;
s_ew_2=0.0;

%*****************************************************

%sampling period and initalized values
T_t=0.3;                        %total time
ts=0.001;                       %1ms
ti=0.0001;                      %100us
dt=0.000001;                    %1us
k_1=T_t/ts;                     %speed roof repeat number(300)-1ms[per]
k_2=ts/ti;                      %current roof repeat number(10)-100us[per]
k_3=ti/dt;                      %motor model repeat mumber(100)-1us[per]
zz=0;
%total cycle repeat number at time shearing
k_0=k_1+1;
y0=zeros(k_0,1);                %time vector
y1=y0 ;                         %speed reference
y2=y1;                          %1-axis speed response
y3=y2;                          %2-axis speed response
y4=y3;                          %positon error
y5=y4;                          %d-1axis current
y6=y5;                          %q-1axis current
y7=y6;                          %a phase voltage
y8=y7;                          %b phase voltage
y9=y8;                          %a phase current
y10=y9;                         %b phase current
```

```
%replace for matrix first atoms variable value at 0 sec
y0=0:ts:T_t;
y1(1,1)=wref;
y2(1,1)=wrm_1;
y3(1,1)=wrm_2;
y4(1,1)=e_theta_1;
y5(1,1)=id_1;
y6(1,1)=iq_1;
y7(1.1)=Ra*id_1;
y8(1,1)=Ra*iq_1;
y9(1,1)=La*id_1;
y10(1,1)=La*iq_1;

%*****************************************************

% ##### Speed Roof Start ##### %*****************
for k_v=1:1:k_1;

  if k_v >=0&k_v<10 ;                          %speed reference of ramp
      wref=150000*0.001+wref;
   end
   if k_v >=10&k_v<300 ;
      wref=1500.0;
   end
   y1(k_v+1,1)=wref;
   wref_0_1=pole*wref*2.0*pi/60.0;
   wref_0_2=pole*wref*2.0*pi/60.0;

% load torque input
  if k_v >=0.1/ts+1;
     Tl_1=0.425;
     Tl_2=-0.95;
  end

% comparison of disturbuance
    c_1=abs(Tl_1);
    c_2=abs(Tl_2);

          if c_1   > c_2 & c_theta >= 0.000174 ; % 0.01 degree
           wref_0_2=wre_0_1;
          end

           if c_2   > c_1 & c_theta >= 0.000174 ;
            wref_0_1=wre_0_2;
           end

% 1-axis q-axis current response calculation
e_w_1=wref_0_1-sum_1;
```

```
s_ew_1=s_ew_1+(e_w_1*ts) ;
iq_ref1_1=a*Ksp*e_w_1+Ksi*s_ew_1;
iq_ref2_1=(1-a)*Ksp*wre_0_1;
iq_ref_1=iq_ref1_1-iq_ref2_1;

% 2-axis q-axis current response calculation
e_w_2=wref_0_2+sum_2;
s_ew_2=s_ew_2+(e_w_2*ts) ;
iq_ref1_2=a*Ksp*e_w_2+Ksi*s_ew_2;
iq_ref2_2=(1-a)*Ksp*wre_0_2;
iq_ref_2=iq_ref1_2-iq_ref2_2;

%************************************************

%##### Current roof start #####%*****************
for k_i=1:1:k_2;
    seta_1=seta_1+wre_0_1*ti;
    if seta_1 >=(2.0*pi)
        seta_1=seta_1-(2.0*pi);
    end
    if seta_1<=(-2.0*pi)
        seta_1=seta_1+(2.0*pi);
    end

%1-axis voltage calculation
e_id_1=zero-id_1 ;
s_eid_1=s_eid_1+(e_id_1*ti);
vdd_1=(Kcp*e_id_1+Kci*s_eid_1-wre_0_1*La*iq_1)/vdc_ref;
e_iq_1=iq_ref_1-iq_1 ;
s_eiq_1=s_eiq_1+(e_iq_1*ti);
vqq_1=(Kcp*e_iq_1+Kci*s_eiq_1+wre_0_1*(Phi+La*id_1))/vdc_ref ;
va_1=cos(seta_1)*vdd_1-sin(seta_1)*vqq_1;
vb_1=sin(seta_1)*vdd_1+cos(seta_1)*vqq_1;
%*******************************************************
  seta_2=seta_2+wre_0_2*ti;
  if seta_2 >=(2.0*pi)
      seta_2=seta_2-(2.0*pi);
  end
  if seta_2<=(-2.0*pi)
      seta_2=seta_2+(2.0*pi);
  end

% 2-axis voltage calculation
e_id_2=zero-id_2 ;
s_eid_2=s_eid_2+(e_id_2*ti);
vdd_2=(Kcp*e_id_2+Kci*s_eid_2-wre_0_2*La*iq_2)/vdc_ref;
e_iq_2=iq_ref_2-iq_2 ;
s_eiq_2=s_eiq_2+(e_iq_2*ti);
vqq_2=(Kcp*e_iq_2+Kci*s_eiq_2+wre_0_2*(Phi+La*id_2))/vdc_ref ;
```

```
va_2=cos(seta_2)*vdd_2-sin(seta_2)*vqq_2;
vb_2=sin(seta_2)*vdd_2+cos(seta_2)*vqq_2;

%*****************************************************

% ##### Motor model Simulation part Start ##### %***
for k_m=1:1:k_3;
% 1-axis current
did_1=(-Ra/La*id_1) +1/La*vdd_1 ;
diq_1=(-Ra/La*iq_1) +1/La*vqq_1 ;
id_1=id_1+did_1*dt;
iq_1=iq_1+diq_1*dt;

% 2-axis current
did_2=(-Ra/La*id_2) +1/La*vdd_2 ;
diq_2=(-Ra/La*iq_2) +1/La*vqq_2 ;
id_2=id_2+did_2*dt;
iq_2=iq_2+diq_2*dt;

end                                    %motor simulation roof end
%*************************************************************
ia_1=cos(seta_1)*id_1-sin(seta_1)*iq_1;
ib_1=sin(seta_1)*id_1+cos(seta_1)*iq_1;

ia_2=cos(seta_2)*id_2-sin(seta_2)*iq_2;
ib_2=sin(seta_2)*id_2+cos(seta_2)*iq_2;

% 1-axis torque
Te_1=pole*Phi*iq_1;
Tm_1=Te_1-Tl_1;
% 2-axis torque
Te_2=pole*Phi*iq_2;
Tm_2=Te_2-Tl_2;

end                                    %current roof end
%*************************************************************
%1,2-axis speed calculation
wre_0_1=Tm_1*ts/Jm+wref_0_1 ;
wrm_1=wre_0_1*60.0/(2.0*pi)/pole;
y2(k_v+1,1)=wrm_1;

wre_0_2=Tm_2*ts/Jm+wref_0_2 ;
wrm_2=wre_0_2*60.0/(2.0*pi)/pole;
y3(k_v+1,1)=wrm_2;

%*****************************************************
%*****************************************************
% position synchronous controller
```

```
er_1=wre_0_1-wre_0_2;
e_theta_1=e_theta_1+er_1*ts;
e_wre=e_theta_1*kpp;
y4(k_v+1,1)=e_theta_1;
c_theta=abs(e_theta_1)

sum_1=e_wre+wre_0_1;
sum_2=e_wre-wre_0_2;

y5(k_v+1,1)=id_1;
y6(k_v+1,1)=iq_1;
y7(k_v+1,1)=va_1;
y8(k_v+1,1)=vb_1;
y9(k_v+1,1)=ia_1;
y10(k_v+1,1)=ib_1;

zz=zz+1
end                                    %speed roof end

%=========================================================
%=========================================================
figure(1)
set (gcf, 'NumberTitle','off')
set (gcf, 'Name','Speed')
plot(y0,y1,'c-',y0,y2,'r-',y0,y3,'b-')
xlabel('time[sec]')
ylabel('real speed[rpm]','fontsize',8)
axis([0,0.3,0,1600])
grid

figure(2)
set (gcf, 'NumberTitle','off')
set (gcf, 'Name','Position')
plot(y0,y4,'r-')
xlabel('time[sec]')
ylabel('position error[rad]','fontsize',8)
axis([0,0.3,-0.05,0.05])
grid

figure(3)
set (gcf, 'NumberTitle','off')
set (gcf, 'Name','Current')
plot(y0,y5,'r-',y0,y6,'b-')
xlabel('time[sec]')
ylabel('d,q current[A]','fontsize',8)
axis([0,0.3,-1.0,1.2])
grid

figure(4)
```

```
set (gcf, 'NumberTitle','off')
set (gcf, 'Name','a,b phase Voltage')
plot(y0,y7,'r-',y0,y8,'b-')
xlabel('time[sec]')
ylabel('a,b phase voltage[V]','fontsize',8)
axis([0,0.3,-4.0,8.0])
grid

figure(5)
set (gcf, 'NumberTitle','off')
set (gcf, 'Name','a, b phase Current')
plot(y0,y9,'r-',y0,y10,'b-')
xlabel('time[sec]')
ylabel('a,b phase  current[V]','fontsize',8)
axis([0,0.3,-1.0,1.0])
grid
```

B. Experiment Program

```
************************************************************************************************************************
   BLDC Motor 2-axis Position Synchronous Control System
   HW : PE-Expert 2.0
   SW : TI社 C-compiler 4.70
                                                                    By Seong-Kwon, Yoo 2001.10.24
************************************************************************************************************************

#include   <mwmath.h>
#include   <mwio.h>

/********  변수치 설정  *********/

#define    FS                  10000.0          /* 캐리어 주파수 [Hz] */
#define    de_tm0              1000.0           /* timer0 인터럽트 주기 [us] */
#define    vdc_ref             161.0            /* Inverter의 직류 최대전압 [V] */
#define    iq_ref_max          1.68             /* iq 최대 지령전류 [A] */
#define    de_pulse            32768.0          /* 8192p/rev의 4체배 */
#define    pole                4.0              /* 극대수 */
#define    ke                  0.2836           /* 역기전력 계수  [rad s /m] */
#define    La                  0.0183           /* d,q 축 인덕턴스 [H] */
#define    kcp                 75.6             /* 전류제어기 P 게인  */
#define    kci                 366.0            /* 전류제어기 I 게인  */
#define    ksp                 0.3787           /* 속도제어기 P 게인  */
#define    ksv                 302.9245         /* 속도제어기 I 게인  */
#define    kpp                 100.0            /* 위치제어기 P 게인 */
#define    alpha               0.3              /* PI 2자유도 게인 */
#define    Alfa                0.0              /* 엔코터 Z상 영점보정 */
```

```
#define    dn_max                  0.9*de_pulse
#define    iu_offset      -0.11
#define    iw_offset      -0.11

#define    CH0_range               31.25              /* u_0상 검출전류 AD range */
#define    CH1_range               31.25              /* w_0상 검출전류 AD range */
#define    CH2_range      450.0                       /* Vdc_0 AD range */
#define    CH3_range               31.25              /* Idc_0 AD range */
#define    CH4_range               31.25              /* u_1상 검출전류 AD range */
#define    CH5_range               31.25              /* w_1상 검출전류 AD range */
#define    CH6_range      450.0                       /* Vdc_1 AD range */
#define    CH7_range               31.25              /* Idc_1 AD range */

#define    wref_range     157.0                       /* 속도지령 [rad/s](1500rpm)*/

/*********   각종 변수 지정   *********/

double     twopi, zero, one;                          /* 프로그램 정수 */
double     ts, ftm0, tm0;                             /* 캐리어 & 타이머 */
double     mu_ref0, mv_ref0, mw_ref0 ;                /* 3상 전압지령치 */
double     mu_ref1, mv_ref1, mw_ref1 ;
double     idc_0, vdc_0, iw_0, iu_0;                  /* AD 변수 */
double     idc_1, vdc_1, iw_1, iu_1;
double     iu_offset, iw_offset ;                     /* 상전류 offset 보정치  */
double     w_ref_0, w_ref_1;                          /* 지령속도*/
double     va_0, vb_0, vd_0, vq_0;                    /* ab축, dq축 전압지령 */
double     ia_0, ib_0, id_0, iq_0, iq_ref_0, iq_ref1_0, iq_ref2_0;
double     ia_1, ib_1, id_1, iq_1, iq_ref_1, iq_ref1_1, iq_ref2_1;
           /*ab축, dq축 전류검출. q축 전류지령, 2자유도 전류지령*/
double     eid_0, eiq_0, s_eid_0, s_eiq_0  ;          /* d,q축 전류오차 및 오차적분 */
double     eid_1, eiq_1, s_eid_1, s_eiq_1  ;
double     wrrpm_0, wrrpm_1;                          /* 속도제어 변수 */
double     w_0, we_0, wrpm_0, ew_0, ewrpm_0, s_ew_0;
double     w_1, we_1, wrpm_1, ew_1, ewrpm_1, s_ew_1;
           /* 검출속도(기계, 전기. [rpm]), 속도오차, 오차적분 */
double     a, k;                                      /* 속도증가 연산 */
double     cntn_0, cnto_0, dn_0, cntn_1, cnto_1, dn_1;
           /* 엔코더 현재, 과거 카운트값, 증분(timer int 루틴) */
double     wrt_0, wrt_1, wet_0, wet_1, n_wet_0, n_wet_1;
double     e_rad, e_theta, e_wre, sum_0, sum_1;        /*위치동기 제어기 */

/*****************************************************************************************************/
/* BD0, BD1의 Carrier 동기 인터럽트 루틴 ( 인터럽트 주기 100us : INT1,INT2)
              - 전기각 검출 및 전류제어를 통한 3상전압 지령치 계산 -              */
/*****************************************************************************************************/
void c_int10( void )
```

```
{
  set_board(BD0);                                          /* BD0 초기화 */
  di_int2();                                               /* INT2 금지 */

/************     1축 초기각 설정 및 전기각 검출       ****************/
    cnto_0 = cntn_0;
    cntn_0 = (double)pio_in_cnt();                         /* abz  초기각도 계산 */
    wrt_0 = twopi * cntn_0 / de_pulse + Alfa ;             /* 기계각 연산 */
        if(wrt_0 > twopi) {wrt_0 -= twopi;}                /*  twopi reset. 정전시 */
        else if(wrt_0 < -twopi) {wrt_0 += twopi;}          /*  twopi reset. 역전시 */

    wet_0 = wrt_0*pole ;                                   /* 전기각 연산 */
    n_wet_0 = (signed int)(wet_0 / twopi) * twopi;         /* -2pi ~ 2pi */
        if(wet_0 < 0.0) n_wet_0 += twopi;
    wet_0 = wet_0 - n_wet_0;

/***  u, w 상 전류검출(AD변환), d-q 축 인가전압 계산 및 u-v-w 축 전압인가  ***/
    ad2_start(ADG0);                                       /* AD변환 시작 */
    wait(60);                                              /* 3.4 us 강제지연 */
    ad2_in_grp(ADG0, &idc_0, &vdc_0, &iw_0, &iu_0);/* ADG0 값 읽어들임 */

    ad2_start(ADG1);                                       /* AD변환 시작 */
    wait(60);                                              /* 3.4 us 강제지연 */
    ad2_in_grp(ADG1, &idc_1, &vdc_1, &iw_1, &iu_1);

    iu_0=iu_0 - iu_offset;                                 /* offset 보정 */
    iw_0=iw_0 - iw_offset;

    /* u-v-w 축 -> a-b 축 -> d-q 축 좌표변환 */
    uw2ab(iu_0, iw_0, &ia_0, &ib_0);                       /* uvw -> ab 변환 */
    ab2dq(ia_0, ib_0, wet_0, &id_0, &iq_0);                /* ab -> dq 변환 */

    /* 1축 전류제어기 */
    eid_0 = - id_0;                                        /* d축 지령전류 : 0 */
    s_eid_0 += (eid_0 * ts);                               /* 오차적분 */
    /* d축 지령전압 계산 (비간섭 제어, 정규화) */
    vd_0 = (kcp * eid_0 + kci * s_eid_0 - we_0 * La * iq_0) / vdc_ref;

    eiq_0 = iq_ref_0 - iq_0 ;                              /* q축 지령전류 : iq_ref */
    s_eiq_0 += (eiq_0 * ts);                               /* 오차적분 */
    /* q축전압 (비간섭 제어, 정규화) */
    vq_0 = (kcp * eiq_0 + kci * s_eiq_0 + we_0 * ( ke + La * id_0 ))/vdc_ref;

    /* d-q 축 -> a-b 축 -> u-v-w 축 좌표변환 */
    dq2ab(vd_0, vq_0, wet_0, &va_0, &vb_0);                /* dq -> ab 변환 */
    ab2uvw(va_0, vb_0, &vu_ref0, &vv_ref0, &vw_ref0);/* ab -> uvw 변환 */

    vdc_inv0 = 2.0 / vdc_0;                                /* 변조율 계산 */
```

```
    mu_ref0 = vu_ref0 * vdc_inv0;
    mv_ref0 = vv_ref0 * vdc_inv_0;
    mw_ref0 = vw_ref0 * vdc_inv_0;

    inv2_set_uvw( mu_ref0, mv_ref0, mw_ref0 );          /* inverter에 3상전압 인가 */

  ei_int2();                                            /* INT2 허가 */

}

void c_int08( void )
{
  set_board(BD1);                                       /* BD1의 초기화 */
  di_int1();                                            /* INT1의 금지 */

/***********     2축 초기각 설정 및 전기각 검출      ****************/
    cnto_1 = cntn_1;
    cntn_1 = (double)pio_in_cnt();                      /* abz  초기각도 계산 */
    wrt_1 = twopi * cntn_1 / de_pulse + Alfa ;          /* 기계각 연산 */
        if(wrt_1 > twopi) {wrt_1 -= twopi;}             /*  twopi reset. 정전시 */
        else if(wrt_1 < -twopi) {wrt_1 += twopi;}

    wet_1 = wrt_1*pole ;                                /* 전기각 연산 */
    n_wet_1 = (signed int)(wet_1 / twopi) * twopi;
        if(wet_1 < 0.0) n_wet_1 += twopi;
    wet_1 = wet_1 - n_wet_1;

/***  d-q 축 인가전압 계산 및 u-v-w 축 전압인가  ***/
    iu_1=iu_1 - iu_offset;                              /* offset 보정 */
    iw_1=iw_1 - iw_offset;

    /* u-v-w 축 -> a-b 축 -> d-q 축 좌표변환 */
    uw2ab(iu_1, iw_1, &ia_1, &ib_1);                    /* uvw -> ab 변환 */
    ab2dq(ia_1, ib_1, wet_1, &id_1, &iq_1);             /* ab -> dq 변환 */

    /* 2축 전류제어기  */
    eid_1 =  zero - id_1;                               /* d축 지령전류 : 0 */
    s_eid_1 += (eid_1 * ts);                            /* 오차적분 */
    vd_1 = (kcp * eid_1 + kci * s_eid_1 - we_1 * La * iq_1) / vdc_ref;

    eiq_1 = iq_ref_1 - iq_1 ;                           /* q축 지령전류 : iq_ref */
    s_eiq_1 += (eiq_1 * ts);                            /* 오차적분 */
    vq_1 = (kcp * eiq_1 + kci * s_eiq_1 + we_1 * ( ke + La * id_1 ))/vdc_ref;

    /* d-q 축 -> a-b 축 -> u-v-w 축 좌표변환 */
    dq2ab(vd_1, vq_1, wet_1, &va_1, &vb_1);             /* dq -> ab 변환 */
    ab2uvw(va_1, vb_1, &vu_ref1, &vv_ref1, &vw_ref1);   /* ab -> uvw 변환 */
```

```
    vdc_inv1 = 2.0 / vdc_1;
    mu_ref1 = vu_ref1 * vdc_inv1;
    mv_ref1 = vv_ref1 * vdc_inv_1;
    mw_ref1 = vw_ref1 * vdc_inv_1;

    inv2_set_uvw( mu_ref1, mv_ref1, mw_ref1 );

  ei_int1();                                          /* INT1의 허가 */

}

/*************************************************************************************************/
/* BD0, BD1의 Timer 인터럽트 루틴 (인터럽트 주기 1000us. timer0,timer1)
                - 속도계산 및 속도제어기에 의한 전류지령치 계산 -              */
/*************************************************************************************************/
void c_int09(void)
{
  set_board(BD0);                                     /* BD0의 초기화 */
  di_timer0();                                        /* timer0의 금지 */

    /* 1축 속도계산 */
    dn_0 = -(cntn_0 - cnto_0);                        /* 카운트 값 변동분 */
    if( dn_0 >= dn_max ) dn_0 -= dn_max;      /* counter의 보정  */
    else if( dn_0 <= -dn_max ) dn_0 += dn_max;

    w_0 = twopi * dn_0 / de_pulse * FS;               /* 기계각속도 [rad/s] */
    we_0 = w_0*pole;                                  /* 전기각속도 [rad/s] */
    wrpm_0 = 60.0/twopi*w_0;                          /* 기계각속도 [rpm] */

    w_ref_0 = -a*wref_range;
    a=a+k;                                            /* 속도증가 연산 */
    if(a>1.0) { a=1.0; }                              /* 속도증가 한계 지정 */
    /* 지령속도 한계 : -5*31.426[rad/s]=-1500[rpm] */
    if(w_ref_0<-wref_range) w_ref_0 = -wref_range;
    /* 지령속도 한계 : 5*31.426[rad/s]=1500[rpm] */
    if(w_ref_0>wref_range) w_ref_0 = wref_range;

    wrrpm_0 = 60.0*w_ref_0/twopi;                     /* 지령속도 [rpm] */

    /* 1축 속도제어기 */
    ew_0 = w_ref_0 - sum_0;                           /* 속도오차 */
    ewrpm_0 = 60.0*ew_0/twopi;                        /* 속도오차 [rpm] */
    s_ew_0 += ew_0 * tm0;                             /* 오차적분 */

    iq_ref1_0 = alpha * ksp * ew_0 + ksi * s_ew_0;    /* q축 전류 계산  */
    iq_ref2_0 = (1-alpha)*ksp*w_0;
    iq_ref_0 = iq_ref1_0 - iq_ref2_0;                 /* 2 자유도 제어 */
```

```
  set_board(BD1);                                          /* BD1의 초기화 */

    /* 2축 속도계산 */
    dn_1 = -(cntn_1 - cnto_1);                             /* 카운트 값 변동분 */
    if( dn_1 >= dn_max ) dn_1 -= dn_max;      /* counter의 보정  */
    else if( dn_1 <= -dn_max ) dn_1 += dn_max;

    w_1 = twopi * dn_1 / de_pulse * FS;                    /* 기계각속도 [rad/s] */
    we_1 = w_1*pole;                                       /* 전기각속도 [rad/s] */
    wrpm_1 = 60.0/twopi*w_1;                               /* 기계각속도 [rpm] */

    w_ref_1 = -a*wref_range;
    a=a+k;                                                 /* 속도증가 연산 */
    if(a>1.0) { a=1.0; }                                   /* 속도증가 한계 지정 */

    if(w_ref_1<-wref_range) w_ref_1 = -wref_range;
    if(w_ref_1>wref_range) w_ref_1 = wref_range;

    wrrpm_1 = 60.0*w_ref_1/twopi;                          /* 지령속도 [rpm] */

    /* 2축 속도제어기 */
    ew_1 = w_ref_1 + sum_1;
    ewrpm_1 = 60.0*ew_1/twopi;
    s_ew_1 += ew_1 * tm0;

    iq_ref1_1 = alpha * ksp * ew_1 + ksi * s_ew_1;
    iq_ref2_1 = (1-alpha)*ksp*w_1;
    iq_ref_1 = iq_ref1_1 - iq_ref2_1;

    /* 위치동기 제어기 */
    e_rad = we_0 - we_1 ;
    e_theta += e_rad * tm0 ;
    e_wre = e_theta*kpp ;
    sum_0 = e_wre + we_0 ;
    sum_1 = e_wre - we_1 ;

  ei_timer0();                                             /* timer0의 허가 */

}

/*****************************************************************************************************/
/*                                        Main 루틴
        - 초기치 설정 및 초기화, 사용함수의 초기화, AD변환 range 설정 -          */
/*****************************************************************************************************/

void main( void )
{
/********************  초기치 설정  **********************/
   twopi = 6.28318 ;
```

```
zero   = 0.0 ;
one    = 1.0 ;

/* Carrier, Timer 주기 */
ts = one / fs;                          /* 1 sampling 시간 : 100 [us] */
tm0 = de_tm0 / 1000.0 / 1000.0;         /* timer 주기 : 1000[us] => 0.001[s] */
ftm0 = one / tm0;                       /* timer interrupt 주파수 : 1000[Hz] */

/* 3상전압 지령치 */
mu_ref0 = zero;
mv_ref0 = zero;
mw_ref0 = zero;
mu_ref1 = zero;
mv_ref1 = zero;
mw_ref1 = zero;

/* 출력 선전압 */
iu_0 = zero;
iw_0 = zero;
iu_1 = zero;
iw_1 = zero;

/* 지령속도 */
w_ref_0 = zero;
w_ref_1 = zero;

/* 속도변속 (증가) */
a=0.0;
k=tm0;

/* encoder 카운트 */
cntn_0 = 0.0;
cnto_0 = 0.0;
dn_0    = 0.0;
cntn_1 = 0.0;
cnto_1 = 0.0;
dn_1    = 0.0;

/* 전기각, 기계각 */
wet_0   = zero;
wet_1   = zero;

/* dq축, ab축 전압지령 */
vd_0 = zero;
vq_0 = zero;
va_0 = zero;
vb_0 = zero;
vd_1 = zero;
vq_1 = zero;
```

```
    va_1 = zero;
    vb_1 = zero;

    /* dq축, ab축 검출전류, q축 지령전류 */
    ia_0 = zero;
    ib_0 = zero;
    id_0 = zero;
    iq_0 = zero;
    iq_ref_0 = zero;
    ia_1 = zero;
    ib_1 = zero;
    id_1 = zero;
    iq_1 = zero;
    iq_ref_1 = zero;

    /* 전류오차 오차적분 */
    eid_0 = zero;
    eiq_0 = zero;
    s_eid_0 = zero;
    s_eiq_0 = zero;
    eid_1 = zero;
    eiq_1 = zero;
    s_eid_1 = zero;
    s_eiq_1 = zero;

    /* 지령속도, 속도오차, 오차적분 */
    w_0 = zero;
    ew_0 = zero;
    s_ew_0 = zero;
    w_1 = zero;
    ew_1 = zero;
    s_ew_1 = zero;

/*************          Main 연산          ***************/

     set_board(BD0);                          /* Board 0(1축)의 초기화  */

      /* ＡＤ보드 range 설정 */
     ad2_set_range(ADG0, CH0_range, CH1_range, CH2_range, CH3_range);
     ad2_set_range(ADG1, CH4_range, CH5_range, CH6_range, CH7_range);

     /*  인터럽트 처리의 정의 */
     init_timer0(c_int09, de_tm0);            /* timer0 처리루틴 & 주기 */
     inv2_init(0,FS);                         /* INT1 변조방식(삼각파변조) & 주기 */

     init_int1(c_int10);                      /* INT1 지정 */
     ei_int1();                               /* INT1 허가 */
     inv2_set_syncint(5.0);                   /* 캐리어 주기 5us 이전에 인터럽트 */
```

```
    /* Counter */
    pio_clr_cnt();                          /* 카운터 보드 클리어  */

    /* Timer */
    start_timer0();                         /* Timer0 동작 개시  */
    ei_timer0();                            /* Timer0 인터럽트 허가  */

    /*  인버터 출력개시  */
    inv2_start_int();                       /* 캐리어 동기 인터럽트 개시 */
    inv2_start_pwm();                       /* 캐리어 파형 출력 개시 */

/*************************************************************************/
    set_board(BD1);                         /* Board 1(2축)의 초기화  */

    /*  인터럽트 처리의 정의 */
    inv2_init(0,FS);                        /* INT2 변조방식(삼각파변조) & 주기 */

    init_int2(c_int08);                     /* INT2 루틴 지정 */
    ei_int2();                              /* INT2 허가 */
    inv2_set_syncint(5.0);                  /* 캐리어 주기 5us 이전에 인터럽트 */

    /* Counter */
    pio_clr_cnt();                          /* 카운터 보드 클리어  */

    /*  인버터 출력개시  */
    inv2_start_int();                       /* 캐리어 동기 인터럽트 개시 */
    inv2_start_pwm();                       /* 캐리어 파형 출력 개시 */

    for(;;)
     {
      watch_data();                         /* 각 변수 파형 관측 */
      }

}
```

감사의 글

누군가 “삶은 스스로 개척하는 것” 이라고 말을 하였습니다. 2년 동안 AMCL에서 연구하면서 저는 나 자신의 지식의 틀을 넓히는 계기를 가졌다고 생각합니다.

먼저, 본 논문이 나오기까지 질적·양적으로 많이 애써 주신 정석권 지도교수님께 깊은 감사를 드립니다. 많이 부족한 저에게 결코 쉽지 않은 분야인 모터제어의 기초에서 응용까지 착실히 지도해 주신 덕분으로 오늘의 이 결과가 있었다고 생각됩니다. 또한, 연구분야 이외에도 항상 관심과 사랑으로 지켜봐 주셨으며, 애정 어린 충고에서 많은 가르침을 받았다고 생각됩니다. 그리고 자동제어분야의 권위자이신 양주호 교수님께도 감사를 드립니다. 학부과정에서부터 졸업하는 순간까지 항상 옆에서 지켜봐 주시고 질책과 사랑으로 이끌어주신 것에 대해 고개 숙여 감사를 드립니다. 아울러 심사과정에서 적절한 충고로서 더 나은 논문을 이끌 수 있도록 도와주신 김상봉 교수님께도 감사를 드립니다. 그리고 멀리 여수에서 후배들의 연구에 항상 관심을 가지고 지켜봐 주시는 변정환 교수님께도 이 자리를 빌어 감사를 드립니다.

짧은 연구실 생활이었지만 그 안에서 소중한 사람들과 같이 있음으로써 행복했던 시절이었던 것 같습니다. 이국 땅의 생활에 이제 나보다 더 적응을 잘하고, 고비 때마다 지식의 틀을 잡도록 도와준 이진국군, 같은 파트너로써 실험실의 온갖 궂은일들을 자신의 일처럼 하며 실험실을 이끌어 왔던 김영진군, 생각하면 늘 아쉬움이 가득한 이동길군, 선배들과 전공이 달라서 많이 애먹었던 냉동공조학과의 귀염둥이 유정원군과 안혜경양, 지금 졸업하고 산업현장에서 일하고 있는 황동일군과 일본에서 박사과정을 수료중인 권혁진군, 남태평양과 군대 그리고 일본에서 항상 우리들을 주시하고 있는 이병혁군과 최봉석군, 정영미양, 그리고 AMCL에 몸담았었던 모든 이들에게 깊은 고마움을 전하면서 항상 건강하고 행복하기를 바랍니다. 또한, 여러분들 각자가 바로 AMCL의 주춧돌이라는 것을 늘 생각하시기를 바랍니다.

끝으로 언제나 나를 믿어주시고 사랑으로 보살펴 주신 어머님과 형에게 깊은 고마움을 전합니다. 아참, 누님들에게도 감사해요.....^^;

참 고 문 헌

1. 변정환, 정석권, 양주호, “다축 구동 시스템의 정밀 위치동기제어(Ⅰ)”, 정밀공학회지, Vol. 13, No. 7, pp. 115～121, 1996
2. 양주호, 변정환, 김영복, 정석권, “다축구동 시스템의 정밀 위치동기제어(Ⅱ)-가변구조 및 H_∞제어법에 의한 설계”, 정밀공학회지, Vol. 14, No. 3, pp. 98～106, 1996
3. 변정환, 여동준, “외란 관측기를 이용한 2축 구동 시스템의 동기제어”, 대한기계학회 논문집 A권, Vol. 25, No. 2, pp. 242～249, 2001
4. 변정환, 여동준, “H_∞제어기법을 이용한 2축 구동 시스템의 위치동기제어”, 정밀공학회지, Vol. 18, No. 2, pp. 192～198, 2001
5. 정석권, 김영진, “2축 연속회전시스템의 고정도 위치동기제어”, 전기학회 춘계학술대회 논문집, pp. 311～313, 2001
6. 유성권, 권혁진, 정석권, “BLDC 모터와 2자유도PI 제어기를 이용한 2축 시스템의 정밀 위치동기제어”, 동력기계공학회 춘계학술대회 논문집, pp. 51～56, 2000
7. 小山正人, 長野鐵明, “規範モデルを用いた電動機の2自由度位置制御”, 日本電氣學會論文集D, Vol. 114, No. 2, pp. 137～143, 1994
8. 後藤聰 外 3人, “複數サーボ系における縱軸逆ダイナミクスを用いた主軸位置同期制御法”, 日本計測自動制御學會論文集, Vol. 30, No. 6, pp. 669～676, 1994
9. 杉本英彦, 小山正人, 玉井伸三, “ACサボシステムの理論と設計の實際”, 綜合電子出版社, pp. 76～80, 1990
10. Y. Dote and S. Kinoshita, "Brushless Servomotors", Clarendon Press, 1990
11. F. Blaschke, "The Principle of Field Orientation as Applied to the New TRANSVEKTOR Closed-Loop Control System for Rotating-Field Machines", Siemens Review, Vol. 34, pp. 217～220, 1972