

모델기반 설계를 이용한 이륜 독립진자 로봇의 임베디드 제어시스템

구대관¹, 지준근^{1*}, 차귀수¹
¹순천향대학교 전기통신공학과

Embedded Control System of Segway Robot using Model Based Design

Dae-Kwan Ku¹, Jun-Keun Ji^{1*} and Guesoo Cha¹

¹Department of Electrical Communication Engineering, Soonchunhyang University

요 약 본 논문에서는 모델기반 설계를 이용한 이륜 독립진자 로봇의 설계방법에 대해 제시한다. 임베디드 시스템의 제어 프로그램 설계는 MATLAB/SIMULINK를 사용한 모델기반 설계에 의해 간편하고 손쉽게 구현되었으며, 로봇은 NXT 마인드스톰, 서보 직류전동기, 초음파센서, 자이로센서, 광센서로 구성되었다. 이 로봇은 불안정한 비선형 시스템이며 몸체 경사각 제어문제를 가지고 있는데, 제어기 설계는 상태궤환 LQR 제어를 이용하였다. 타겟이 되는 프로세서에 종속적이지 않은 모델기반 설계는 문서기반 설계보다 프로그램 개발, 오류 발견 및 수정, 소프트웨어 구조 파악의 측면에서 장점을 가지고 있음을 제어기 설계와 실험을 통해서 확인할 수 있었다.

Abstract In this paper, embedded control system of segway robot using model based design is presented. Design of control program in embedded system can be implemented simply and easily by model based design method using MATLAB/SIMULINK. Segway robot is consisted of a NXT Mindstorms controller, two DC servo motors, a ultrasonic sensor, a gyro sensor, and a light sensor. It is a unstable nonlinear system and has a control problem of body pitch angle. So controller of segway robot is designed using state feedback LQR control. It is confirmed through design and experiment of controller that the model based design method, that is not depend on target processor, has merits compared with the text based design in aspects such as a program development, an error detection/modify, and an insight of software structure.

Key Words : Segway, Model Based Design, NXT Mindstorms, Embedded Control System

1. 서론

오늘날 과학기술 및 공학의 발달로 인해 인간은 제어 시스템을 배제하고는 살 수 없게 되었다. 가정, 산업체, 항공, 해운 등 인간 의식주 생활과 경제활동에 있어 제어 시스템이 깊숙이 관여하고 있다. 제어시스템은 제어대상과 제어부로 구성되는데, 제어부는 마이크로프로세서(Microprocessor)나 DSP(Digital Signal Processor) 같은 CPU로 구성된 디지털 제어시스템으로 프로그램에 의해 제어대상의 목적에 맞게 제어를 수행한다.

기존의 제어시스템 설계는 C언어와 같은 텍스트(text)

기반의 프로그래밍을 통해 구현되었다. 이 방법은 제어시스템 개발 초기 단계에서 코드오류가 발생할 가능성이 있으나 오류의 발견이 쉽지 않고, 또한 프로그래밍 언어가 타겟(target)에 종속적이기 때문에 타겟의 변경이 생기면 프로그램을 새롭게 작성해야 한다는 단점이 있다. 하지만 최근 각광을 받고 있는 모델기반 설계(Model-Based Design : MBD) 방법으로 제어시스템을 먼저 설계한 후, 자동으로 C 코드(code) 생성을 하면 위와 같은 문제점을 겪지 않고 빠른 시간안에 손쉽게 제어시스템 설계가 가능하며 사용자들이 C 코딩(coding)에 대한 부담이 없이 편리하게 접근할 수 있어 임베디드(Embedded) 제어시스

*교신저자 : 지준근(jkji@sch.ac.kr)

접수일 10년 04월 20일

수정일 (1차 10년 06월 23일, 2차 10년 08월 09일)

게재확정일 10년 08월 10일

템의 구현에 큰 도움이 될 수 있다.

본 논문에서는 텍스트 기반 프로그래밍의 단점들을 갖지 않고 개발-유지-관리 측면에서 많은 장점을 갖는 모델 기반 설계에 기초한 자동 C 코드생성을 통해 세그웨이형 이륜 도립진자 로봇(2-wheeled inverted pendulum robots)의 임베디드 제어시스템을 설계 및 구현하였다.

NXT 마인드스톰을 이용한 기존의 이륜 도립진자 로봇 연구는 LQR 제어를 사용하고 있으며, 본 논문에서는 세그웨이 로봇의 LQR 제어를 다시 설계 및 구현하고 라인주행 기능 및 장애물 회피 기능을 추가하여 실험을 수행하였다.

2. 전체 시스템의 구성

이륜 도립진자는 도립진자원리를 이용한다. 도립진자 시스템은 가장 일반적인 진자시스템이다. 이 시스템은 기본적으로 불안정한 비선형 제어 시스템이고 제어 시스템의 성능을 평가하기 위한 제어 대상으로 많이 다루어져 왔으며, 2축 보행 로봇이나 미사일의 비행 자세를 제어하는 문제에 많이 응용되어 왔다[1,2].

2.1 NXT 마인드스톰 제어기

NXT는 32비트 ARM7 마이크로프로세서(256 Kbytes FLASH, 64 Kbytes RAM)를 장착한 소형 컨트롤러이다. 32비트 CPU는 기본 디지털 입출력, 인터럽트, 디스플레이, USB 2.0 및 블루투스 통신 등과 같은 제어 로직을 실행한다. 또한 NXT에 함께 내장된 8비트 AVR 마이크로프로세서(4 Kbytes FLASH, 512 Byte RAM)는 A/D 변환, PWM 신호발생 및 버튼 조작 등의 기능을 실행한다. 각각의 프로세서에는 플래쉬(Flash) 메모리와 RAM이 있으며, 프로세서 간에는 I2C 방식으로 통신한다[3,4].

2.2 입력포트와 센서

NXT는 4개의 센서 입력포트(1,2,3,4)를 갖고 있다. 이들 포트에는 터치 센서, 광 센서, 사운드 센서 및 초음파 센서를 연결할 수 있다. 이 센서들을 이용해 측정 데이터를 궤환시켜 프로세서가 피드백 제어 프로그램을 실행시킬 수 있도록 만들 수 있다[3,4].

2.3 출력포트

NXT는 상단부에 3개의 디지털 출력포트 (A, B, C)를 갖고 있으며, 각각 독립적으로 모터 또는 램프의 구동이 가능하다. "ON" 상태에서는 모터의 "정방향" 또는 "역방

향" 구동이 가능하며 "OFF" 상태에서는 모터를 강제로 브레이크 걸거나, 혹은 모터가 자연 감속해서 정지하도록 할 수 있다. 이 포트는 기본적으로 PWM을 통해 모터를 구동한다[3,4].

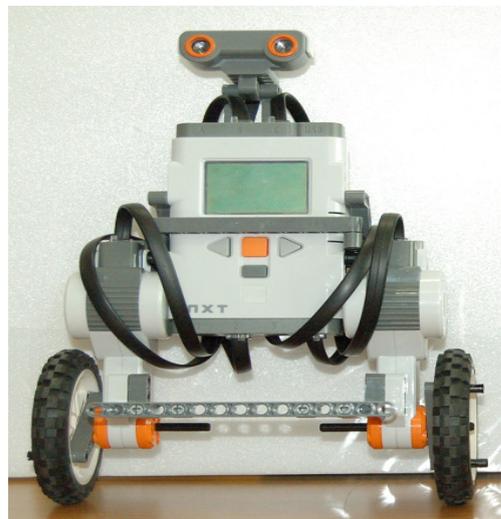
2.4 직류 전동기

마인드스톰의 서보전동기는 전력효율이 매우 좋으며, 크기가 작은 부하에 대해서는 전동기 자체의 플라이휠 효과가 전동기 전력효율을 좋게 한다.[3,4].

3. 이륜 도립진자 로봇의 설계

3.1 이륜 도립진자 로봇의 외형

그림 1은 세그웨이 로봇의 전체 모습을 보여준다. 로봇의 각 부분을 기능으로 나누면 크게 제어부, 센서부, 구동부로 나눌 수 있다. 제어부는 NXT 마인드스톰의 ARM7 & AVR 프로세서를 장착한 임베디드 시스템으로 다운로드한 제어 프로그램을 실행하는 역할을 수행한다. 센서부는 제어가 제어를 하기위해 필요한 물리량들을 측정한다. 전동기의 회전수를 측정하기 위한 엔코더, 로봇 몸체의 경사각을 측정하기 위한 자이로스코프, 전방 장애물을 탐지하기 위한 초음파 센서, 라인 트레이싱을 위한 광센서를 사용한다.

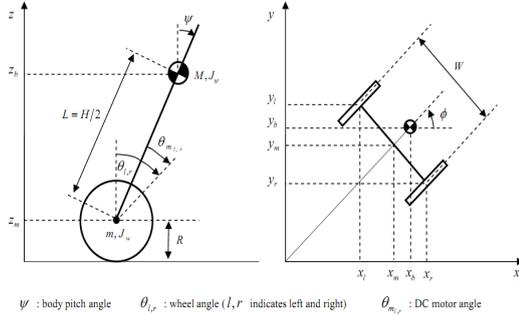


[그림 1] 이륜도립진자 로봇

3.2 이륜 도립진자 로봇의 모델링

제어 대상의 운동방정식을 유도하고 상태방정식을 세

우기 위해서 이륜 독립진자의 자유체도(Free body diagram)를 나타내면 그림 2와 같다.



[그림 2] 이륜 독립진자 자유체도

그림 2를 참조해 운동방정식을 세울 수 있는데, 본 논문에서는 라그랑지안 방법(Lagrangian Method)를 이용해 운동방정식을 유도할 것이다. 라그랑지안 역학(Lagrangian mechanics)에서 라그랑지안(Lagrangian)은 운동에너지와 위치에너지의 차로 정의하는데, 자유체도를 분석해 운동에너지와 위치 에너지를 구할 수 있다[1,2,5].

라그랑지안 방법은 다음 식 (1)을 이용한다.

$$L = T - V \tag{1}$$

$$\frac{d}{dt} \frac{\partial T}{\partial \dot{q}_j} - \frac{\partial T}{\partial q_j} = Q_j, \quad j = 1, 2, \dots, s$$

3.2.1 운동 에너지 T와 위치 에너지 V

이륜 독립진자 운동에너지는 2가지로 나눌 수 있다. 바퀴 2개와 몸체가 평면에서 직선 운동 할 때 병진 운동 에너지를 갖고, 이륜 독립진자가 중력에 의해 회전하거나 조향할 때 회전 운동 에너지를 갖는다. 병진 운동 에너지 T_1 과 회전 운동 에너지 T_2 는 다음과 같다.

$$T_1 = \frac{1}{2} m (\dot{x}_l^2 + \dot{y}_l^2 + \dot{z}_l^2) \tag{2}$$

$$+ \frac{1}{2} m (\dot{x}_r^2 + \dot{y}_r^2 + \dot{z}_r^2) + \frac{1}{2} M (\dot{x}_b^2 + \dot{y}_b^2 + \dot{z}_b^2)$$

$$T_2 = \frac{1}{2} J_\omega \dot{\theta}_l^2 + \frac{1}{2} J_\omega \dot{\theta}_r^2 + \frac{1}{2} J_\psi \dot{\psi}^2 + \frac{1}{2} J_\phi \dot{\phi}^2 \tag{3}$$

$$+ \frac{1}{2} n^2 J_m (\dot{\theta}_l - \dot{\psi})^2 + \frac{1}{2} n^2 J_m (\dot{\theta}_r - \dot{\psi})^2$$

바퀴와 몸체는 높이를 갖고 있기 때문에 위치에너지가 존재 하는데, 전체 위치 에너지는 다음식 (4)와 같다.

$$V = mgz_l + mgz_r + Mgz_b \tag{4}$$

3.2.2 라그랑지안 방정식

운동에너지와 위치 에너지를 구했으므로 라그랑지안 L을 구할 수 있으며, L은 다음과 같다.

$$L = T_1 + T_2 - V \tag{5}$$

이륜 독립진자의 동태를 완전히 표현하는 일반화 좌표를 다음과 같이 정한다.

$$\theta, \psi, \phi \tag{6}$$

일반화 좌표를 이용한 일반화 힘을 구하면 비선형 미분방정식인데, 이륜 독립진자의 궁극적인 제어목표는 몸체의 경사각 ψ 를 항상 0° 로 수렴시키는 것이다. 이 특징을 이용해 미분방정식의 선형화가 가능하다[5].

그러므로 $\lim_{\psi \rightarrow 0} \sin \psi \rightarrow \psi, \cos \psi \rightarrow 1, \dot{\psi}^2 \rightarrow 0$ 이 되고 다음과 같은 식이 된다.

$$F_\theta = [(2m + M)R^2 + 2J_w + 2n^2 J_m] \ddot{\theta} + (MLR - 2n^2 J_m) \ddot{\psi} \tag{7}$$

$$F_\psi = (MLR - 2n^2 J_m) \ddot{\theta} + (ML^2 + J_\psi + 2n^2 J_m) \ddot{\psi} - MgL\psi$$

$$F_\phi = [\frac{1}{2} m W^2 + J_\phi + \frac{W^2}{2R^2} (J_w + n^2 J_m)] \ddot{\phi}$$

전동기와 토크를 이용해 일반화 힘을 다음 식들과 같이 다르게 표현할 수 있다.

$$F_\theta = F_l + F_r \tag{8}$$

$$F_l + F_r = \frac{nK_t}{R_a} (v_l + v_r) + 2\dot{\psi} (\frac{nK_t K_b}{R_a} + f_m) \tag{9}$$

$$- 2 (\frac{nK_t K_b}{R_a} + f_m + f_w) \frac{1}{2} (\dot{\theta}_l + \dot{\theta}_r)$$

$$F_\psi = -\alpha (v_l + v_r) + 2\beta \dot{\theta} - 2\beta \dot{\psi} \tag{10}$$

$$F_\phi = \frac{W}{2R} (F_r - F_l) \tag{11}$$

$$= \frac{W}{2R} \alpha (v_r - v_l) - \frac{W^2}{2R^2} (\beta + f_m) \dot{\phi}$$

3.2.3 상태 방정식

두 가지 방법으로 구한 일반화 힘을 정리해서 다음 변수에 따라 상태방정식을 만들 수 있다.

$$x_1 = \begin{bmatrix} \theta \\ \psi \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix}, x_2 = \begin{bmatrix} \phi \\ \dot{\phi} \end{bmatrix}, u = \begin{bmatrix} v_l \\ v_r \end{bmatrix} \tag{12}$$

$$\begin{aligned} \dot{x}_1 &= A_1x_1 + B_1u \\ \dot{x}_2 &= A_2x_2 + B_2u \end{aligned} \quad (13)$$

x_1 과 x_2 에 대한 상태방정식은 다음 식들과 같고,

$$\begin{bmatrix} \dot{\theta} \\ \dot{\psi} \\ \ddot{\theta} \\ \ddot{\psi} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ A_1(3,1) & A_1(3,2) & A_1(3,3) & A_1(3,4) \\ A_1(4,1) & A_1(4,2) & A_1(4,3) & A_1(4,4) \end{bmatrix} \begin{bmatrix} \theta \\ \psi \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} \quad (14)$$

$$+ \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ B_1(3,1) & B_1(3,2) \\ B_1(4,1) & B_1(4,2) \end{bmatrix} \begin{bmatrix} v_l \\ v_r \end{bmatrix}$$

$$\begin{bmatrix} \dot{\phi} \\ \ddot{\phi} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ A_2(2,1) & A_2(2,2) \end{bmatrix} \begin{bmatrix} \phi \\ \dot{\phi} \end{bmatrix} \quad (15)$$

$$+ \begin{bmatrix} 0 & 0 \\ B_2(2,1) & B_2(2,2) \end{bmatrix} \begin{bmatrix} v_l \\ v_r \end{bmatrix}$$

식 (7)과 (8),(9),(10),(11)의 항등 관계를 이용하면 식 (14), (15)의 미지수를 구할 수 있다. 미지수를 계산하고, 위 두 개의 상태방정식을 합쳐서 나타내면 다음 식 (16)과 같다.

$$\begin{bmatrix} \dot{\theta} \\ \dot{\psi} \\ \ddot{\theta} \\ \ddot{\psi} \\ \ddot{\phi} \\ \dot{\phi} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & -409.7184 & -162.1273 & 162.1273 & 0 & 0 \\ 0 & 269.6273 & 78.1496 & -78.1496 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1.0000 & 0 \\ 0 & 0 & 0 & 0 & 0 & -95.5684 \end{bmatrix} \begin{bmatrix} \theta \\ \psi \\ \dot{\theta} \\ \dot{\psi} \\ \phi \\ \dot{\phi} \end{bmatrix}$$

$$+ \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 157.5798 & 157.5798 \\ -75.9576 & -75.9576 \\ 0 & 0 \\ -53.0787 & 53.0787 \end{bmatrix} \begin{bmatrix} v_l \\ v_r \end{bmatrix} \quad (16)$$

우변 첫째 항은 로봇의 자세 제어 운동을 표현하고 두 번째 항은 조향운동을 표현한다[1].

4. 제어 시스템 설계

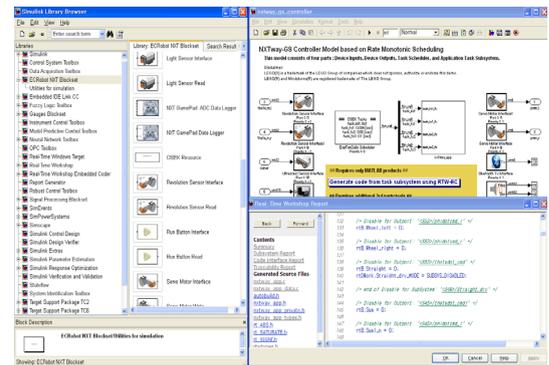
이론 도입진자 로봇을 제어하기 위해 제어프로그램 설계가 필요하다. 전통적인 임베디드 시스템 제어 프로그램은 C언어를 이용한 문지기반설계기법을 통해 개발되었지만 본 논문에서는 MATLAB/SIMULINK를 이용한 모델기반설계기법으로 제어프로그램을 개발한다.

4.1 개발 환경

널리 알려져 있는 모델기반설계를 구현 할 수 있는 소

프트웨어는 MATLAB/SIMULINK, SCADE, LABVIEW, DSPACE 등이 있다. 본 논문에서는 이 중에서 Mathworks사의 제품인 MATLAB/SIMULINK를 사용한다. MATLAB/SIMULINK를 이용한 MBD기법으로 NXT 마인드스톱 제어 프로그램을 설계하기 위해서는 nxtOSEK와 Embedded Coder Robot NXT 소프트웨어가 필요하다.

nxtOSEK는 레고 마인드스톱 NXT를 위한 오픈소스 플랫폼으로 leJOS NXJ의 장치 드라이버, TOPPERS/ATK(TOPPERS/OSEK)와 TOPPERS/JSP 실시간 운영체제로 구성되어 있다. Embedded Coder Robot NXT는 nxtOSEK 운영체제 하에서 실행될 제어 프로그램을 SIMULINK를 이용한 MBD기법으로 설계 할 수 있게 해주는 개발 환경이다.

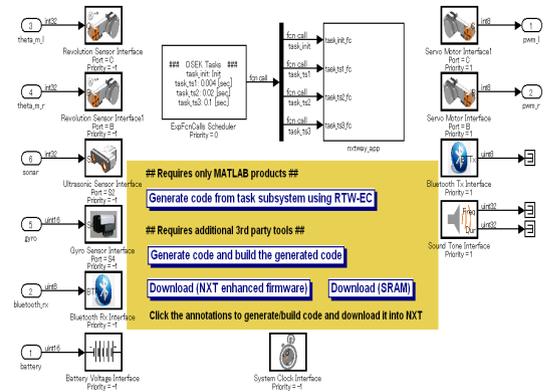


[그림 3] MATLAB/SIMULINK 개발환경

그림 3은 MATLAB/SIMULINK 개발환경과 C코드 생성결과를 보여준다.

4.2 제어 프로그램 설계

4.2.1 전체 구성



[그림 4] 제어 프로그램의 전체 구성

그림 4는 무선조종주행 이륜 도립진자 로봇 제어 프로그램의 전체적인 구성을 보여준다. 좌측 블록들은 Device Input들로서 주로 데이터를 얻어서 프로세서로 보내야 하는 계측장치들이다. 우측 블록들은 Device output들로서 전동기, 블루투스, 스피커 같은 출력장치들이다. 이 로봇은 PC에 연결한 조이 스틱을 이용해 무선으로 주행 제어를 할 수 있으며 라인 주행 운전을 할 수 있다. 두 경우 모두 자율로 몸체 경사각 제어를 하며 주행을 할 수 있다.

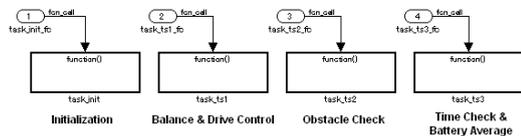
중앙의 Exported Function-Calls Scheduler는 제어주기를 정하고 그 제어주기 마다 호출해 수행해야 하는 서브시스템이 연결된 블록이다. 호출해야 하는 서브시스템은 nxtway_app 블록 안에 프로그램이 되어있다. 중앙 하단부 버튼들은 SIMULINK로 만든 제어 프로그램을 C코드로 변환해 NXT 마인드스톱으로 다운로드 하기위한 것들이다.

그림 5는 Exported Function-Calls Scheduler 블록에서 호출하는 서브시스템 프로그램이다. Initialization 시스템은 제어 프로그램 실행 시 (시작 기준으로 0초) 단 한번만 수행되고, Balance & Drive Control 시스템은 4ms 마다 한번, Obstacle Check 시스템은 20ms 마다 한번, Time Check & Battery Average 시스템은 100ms 마다 한번 수행한다. 중요도가 높은 서브시스템일수록 수행 주기가 짧아진다.

Shared data는 서브시스템 사이에서 공유해야 하는 변수들을 선언하고 초기화 해준 것이다.

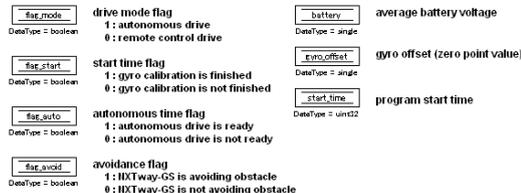
Application Task Subsystems

Each function-call subsystems are driven by function-call signals of the task scheduler and generated as OSEK task functions in the generated code using Export functions feature of RTW-EC.



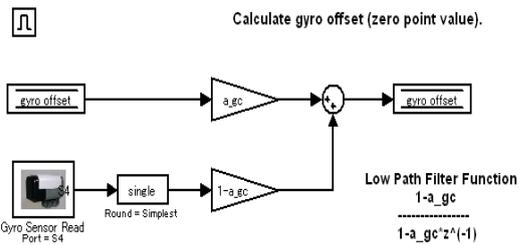
Shared Data

Data Store Memory is used as a shared data between tasks.

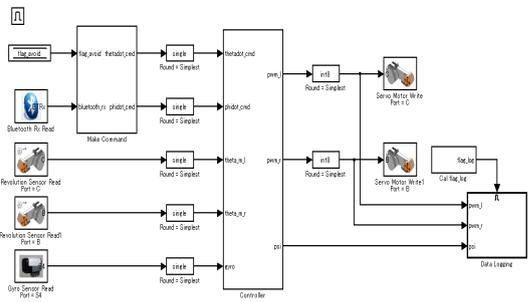


[그림 5] 태스크 서브 시스템

그림 6은 자이로스코프의 오프셋을 계산하는 블록이다. 자이로스코프는 반드시 오프셋을 가지고 있으며 특정 시간 동안 오프셋을 계산한다. DC 오프셋을 측정하기 위해 디지털 저역통과 필터를 이용한다. 이 계산을 몇 초 동안 수행한다[1,5].

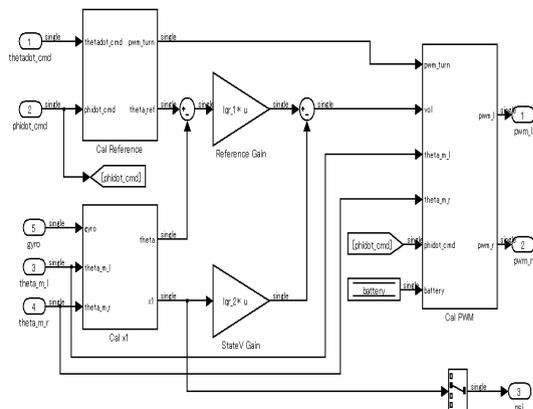


[그림 6] 자이로스코프 보정



[그림 7] 자세 제어 및 주행 제어

그림 7은 자세 제어 및 주행 제어 블록도이다. Make command 박스에서 나온 지령을 Controller 박스에서 제어한 결과로 PWM 제어 신호를 생성해 좌우 서보 전동기에 인가한다. Make command 박스안의 프로그램을 수정하면 라인 트레이싱 로봇을 만들 수 있다.



[그림 8] 상태궤환 LQR 제어기

그림 8은 제어 프로그램의 가장 핵심적인 제어기 부분이다. Cal Reference 블록은 Make Command 블록에서 생성되는 θ_{cmd} 지령 값을 계산하고, Cal x1 블록은 자이로

센서와 서보전동기 엔코더로부터 계측 값을 받아 귀환시켜야 하는 상태변수를 계산한다. 사용한 제어기는 상태귀환 제어기로 다음과 같다.

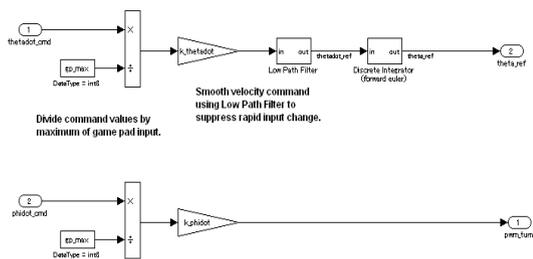
$$u = - \begin{bmatrix} 0 & k_2 & k_3 & k_4 \end{bmatrix} \begin{bmatrix} \theta \\ \dot{\psi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} + k_1(\theta_{ref} - \theta) \quad (17)$$

식 17의 이득 행렬은 LQR제어기를 이용하여 정했다. 행렬 A_1 , B_1 와 다음 식 18의 행렬 Q 와 R 을 시뮬레이션과 실험의 시행착오로 정하고, MATLAB의 lqr함수를 통해 이득 행렬 K 를 결정했다.

$$Q = \begin{pmatrix} 10^2 & 0 & 0 & 0 \\ 0 & 10^5 & 0 & 0 \\ 0 & 0 & 10 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}, R = \begin{pmatrix} 10 & 0 \\ 0 & 1 \end{pmatrix} \quad (18)$$

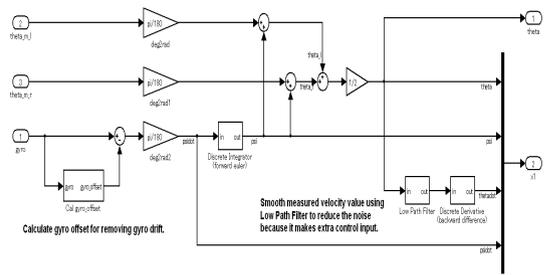
Gain = -(0.9534, 34.2069, 0.7641, 1.7537)
 lqr₁ = -0.9535
 lqr₂ = -(0, 34.2069, 0.7641, 1.7537)

그림 9는 상태변수 지령을 계산하는 블록이다. 바퀴 속도 지령 $\dot{\theta}_{cmd}$ 의 급격한 변화를 막기 위해 디지털 저역통과 필터를 거쳐서 지령신호를 계산한다. 로봇 몸체 경사각 ψ 와 각속도 $\dot{\psi}$ 는 이륜 도립진자 원리에 따라 지령 값이 0으로 나간다.

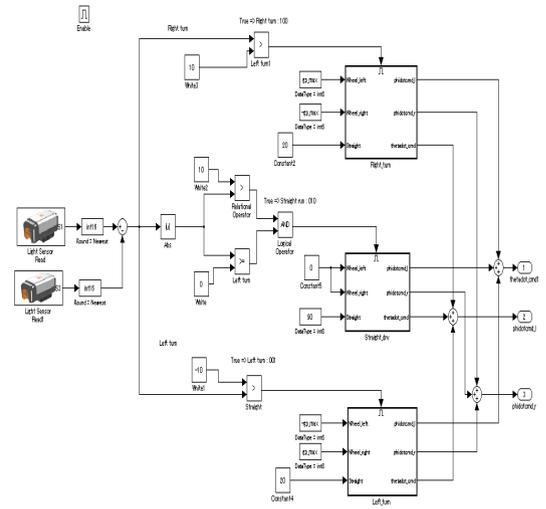


[그림 9] 지령 생성

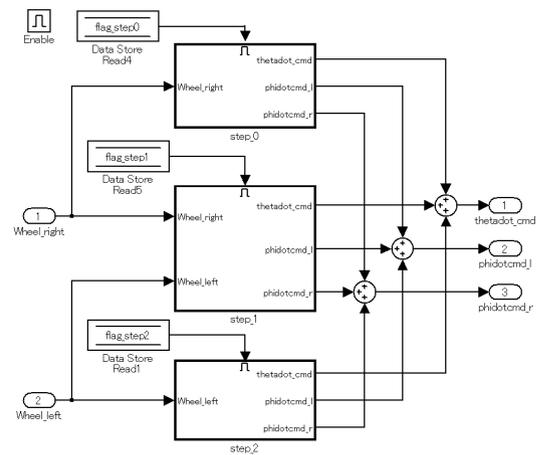
그림 10은 로봇의 센서를 통한 계측 데이터를 바탕으로 현재 상태변수를 계산하는 블록이다. Gyro Calibration에서 자이로 오프셋을 계산하였지만 주행 및 제어 프로그램 수행단계에서도 계속 오프셋이 발생하기 때문에 디지털 저역통과 필터로 오프셋을 계산해내어 자이로 계측 값에서 빼주어야 한다. 이 블록에서도 $\dot{\theta}$ 을 계산할 때 디지털 저역통과 필터를 사용해야 한다.



[그림 10] 현재 상태 변수 계산



[그림 11] 라인트레이싱 - Line_driv 블록



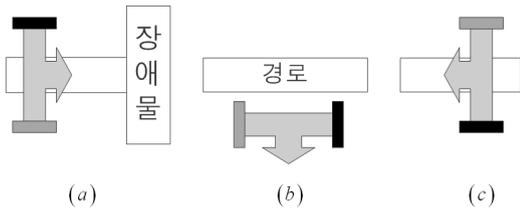
[그림 12] 장애물 탐지시 회피 프로그램

그림 11은 라인주행 알고리즘에 따라 설계한 Line_driv 블록 내부 프로그램이다. 여기서는 경계 값을 10으로 설

정하였다. 두 광센서 계측 값 차이의 절대 값이 10보다 작을 경우 직선라인으로 인식해 직선주행을 하게 되고 계측 값 차이가 10보다 크고 -10보다 작을 경우 곡선라인으로 인식해 커브를 돌게 된다.

그림 12 는 라인 주행 중 장애물 탐지 시 장애물을 회피하여 반대 방향으로 주행하는 알고리즘을 나타낸다. 장애물 탐지 거리는 사용자가 설정 할 수 있다.

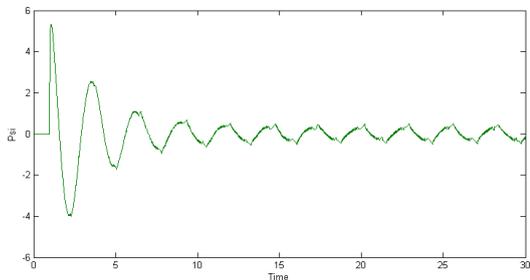
장애물 회피 알고리즘은 그림 13을 따른다. 초음파 센서로 30cm 이내의 장애물을 탐지하면 정지 후 왼쪽 바퀴만 구동하고 정지 후 다시 오른쪽 바퀴만 구동하여 제자리에서 반대 방향으로 위치시킨 뒤에 라인 검사 후 반대 방향으로 주행을 시작한다.



[그림 13] 장애물 탐지시 회피동작

5. 시뮬레이션 및 실험 결과

5.1 시뮬레이션 결과



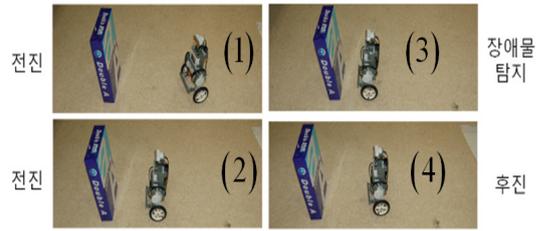
[그림 14] 몸체 경사각 제어 시뮬레이션 결과

실제 실험 전 Embedded Coder Robot NXT에서 제공하는 모델을 이용하여 시뮬레이션을 수행 하였다. 그림 14는 시간에 대한 로봇 몸체 경사각의 시뮬레이션 결과를 보여준다.

5.2 실험 결과

그림 15는 조이스틱으로 무선조종 하는 실험 결과를 보여준다. 초음파로 약 30cm 이내의 전방 장애물을 탐지

할 수 있다.



[그림 15] 장애물 탐지 및 회피

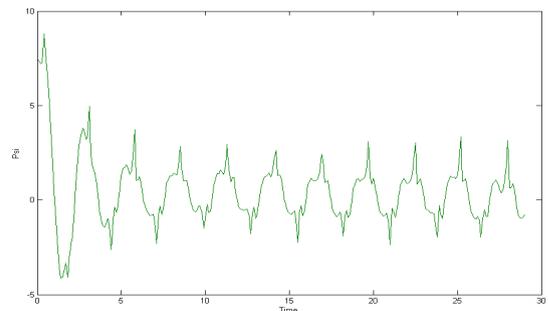
장애물 탐지시 조이스틱의 지령 보다 장애물 회피가 우선하여 20ms 마다 수행되는 Obstacle Check 프로그램이 장애물 탐지를 못할 때 까지 로봇은 후진한다.

그림 16은 라인주행 중 장애물을 탐지하면 장애물 회피 알고리즘대로 회피한 후 반대방향으로 진행하는 것을 볼 수 있다.



[그림 16] 라인주행 및 장애물 회피

그림 17은 로봇의 전진 주행 실험 시 경사각 제어의 결과 그래프이다.



[그림 17] 실제 실험시 몸체 경사각 제어

5. 결론

본 논문에서는 NXT 마인드스톰을 이용하여 세그웨이형 이륜 도립진자 로봇을 제작하고 모델기반 제어기 설계 및 구현을 통해서 성능을 실험해 보았다.

제어 프로그램을 설계하고 실제실험을 하기 전 모델기반설계의 장점인 시뮬레이션을 통해 시행착오를 줄일 수 있었으며, 제어프로그램을 설계한 후 MATLAB/SIMULINK로 설계된 이륜 도립진자 로봇 시뮬레이션 모델에 통합하면 곧바로 시뮬레이션이 가능하고 결과를 확인할 수 있었다. 충분한 시뮬레이션을 통해 제어 프로그램을 검증하고 실제 NXT에 다운로드 할 때 C 언어로 다시 작성할 필요가 없이 MATLAB/SIMULINK의 자동 코드 생성 기능을 이용하여 다운로드 할 수 있었다.

NXT 마인드 스톰은 저연령층이나 대학생들을 위한 교육용 목적의 로봇이다. 이미 준비되어 있는 부품들을 이용해서 로봇을 쉽게 구현시킬 수 있고 MATLAB/SIMULINK 이나 LABVIEW 같은 모델 기반 설계 개발 환경을 이용하여 제어 프로그램을 개발 할 수 있다. 이런 특징들은 대학생들이 모델 기반 설계 기법을 배우고, 실제 제어 시스템 설계의 예로 로봇 설계를 쉽게 할 수 있어 학습 효과를 향상 시킬 수 있었다.

본 논문을 통해서 NXT 마인드스톰과 모델기반설계 기법을 통해 로봇을 개발할 경우 실제 로봇 제작의 전 단계로서 제어기와 센서 시험이 가능한 테스트 베드로서 유용해 로봇 상업화에 도움이 될 수 있었음을 확인했다.

참고문헌

- [1] 최동일, "2륜자기균형이동차의 개발 및 외란 측정기를 이용한 강인제어", 한국과학기술원 기계공학전공 석사학위논문, 2007.
- [2] 신영훈, "관성 센서를 이용한 두 바퀴 도립 진자 제어에 관한 연구", 울산대학교 대학원 전기전자정보시스템공학과 석사학위논문, 2004.
- [3] 김수환, 장순호, "LEGO MINDSTORM을 이용한 로봇 구현", 순천향대학교 정보기술공학부 학사학위논문, 2008.
- [4] 홍선하, 송선미, "C언어로 즐기는 LEGO MINDSTORMS NXT", 이지테크, 2007.
- [5] Takashi Chikamasa, Tomoki Fukuda, "Embedded Coder Robot NXT Modeling Tips", 2009.
- [6] Yori-hisa Yamamoto, NXTway-GS Model-Based Design

구 대 관(Dae-Kwan Ku)

[준회원]



- 2010년 2월 : 순천향대학교 정보기술공학부 (공학사)
- 2010년 3월 ~ 현재 : 순천향대학교 대학원 전기통신시스템 공학과 (공학석사 과정)

<관심분야>

전력전자, 전기기계, 전력변환, 전동기제어

지 준 근(Jun-Keun Ji)

[정회원]



- 1986년 2월 : 서울대학교 전기공학과 (공학사)
- 1988년 2월 : 서울대학교 대학원 전기공학과 (공학석사)
- 1994년 2월 : 서울대학교 대학원 전기공학과 (공학박사)
- 1994년 9월 ~ 현재 : 순천향대학교 전기통신공학과 교수

<관심분야>

전력전자, 전기기계, 전력변환, 전동기 제어, 전원장치, 전력품질제어, 모션제어, 제어응용

차 귀 수(Guesoo Cha)

[정회원]



- 1978년 2월 : 서울대학교 공업교육학과 (공학사)
- 1982년 2월 : 서울대학교 대학원 전기공학과 (공학석사)
- 1987년 2월 : 서울대학교 대학원 전기공학과 (공학박사)
- 1987년 3월 ~ 현재 : 순천향대학교 전기통신공학과 교수

<관심분야>

전기기계, 전력전자, 전력변환, 전동기 제어, 전원장치