

LEGO Control Systems Laboratory Manual

S.R.Manikandasriram

June 24, 2014

This report analyses the scope of using the LEGO Mindstorms NXT Kit as a platform for teaching Control Engineering concepts for both Undergraduate and Graduate students. A set of simple experiments using only a LEGO DC Motor and the “NXT Intelligent Brick” have been analysed. The experiments cover the introductory concepts in Control Engineering like Transfer Function, PID Control, System Identification and Bode Plots and is thus ideal for an Undergraduate Laboratory course on Control Systems. The later half of the report deals with building a 2-wheel Self-balancing Robot using LEGO Mindstorms kits and a HiTechnic Gyro sensor. The mathematical model of the *Segway* type robot is explained in detail and a Servo PID control for stabilising the *Inverted Pendulum* is studied. This would be ideal as a Course Project for first year Masters students.

ACKNOWLEDGEMENT

There is a number of publications on the use of LEGO NXT kits as an educational platform for undergraduate engineering students which provided valuable insights for our analysis. In particular, the first set of experiments discussed in this report is an extension of the paper *A LEGO based Undergraduate Control Systems Laboratory* by Sabiha A Wadoo and Rahul Jain. Also, the 2-wheel self-balancing robot project discussed in this report is an adaptation of the paper *NXTway-GS Model based design* by Y.Yamamoto in 2008.

1 INTRODUCTION

The use of LEGO Mindstorms series of kits as a platform for education has received widespread acceptance since the release of Mindstorms Robotics Invention System in 2000. The LEGO Mindstorms NXT kit is relatively cheap, robust, customizable, reprogrammable and induces enthusiasm and creativity in students. The official software for LEGO NXT is a NI LabView based programming language called NXT-G. However, enthusiasts have extended the software and hardware in various ways to make the LEGO NXT kits as a favourable Rapid prototyping platform for academic and research activities. In this report, we analyse the scope of using LEGO NXT kits for teaching Control Engineering.

A para on the hardware specs and the programming environments.

A para on the other work done on this field

A para on the layout of this report

2 ANALYSING THE LEGO DC MOTOR

The first half of the report explains relatively simple control experiments for analysing a DC Motor system. The analysis is divided into 4 parts - 1. Observing the in-built PID Controller, 2. Writing custom PID controller to meet design requirements, 3. Determining the Transfer Function of the DC Motor and 4. Designing PID controller using the derived Transfer Function. In order to avoid students from having to get used to a new programming language, ROBOTC - a C-like programming language developed by Carnegie Mellon University's Robotics Academy - will be used as software for this section.

2.1 OBSERVE PID CONTROLLER IN ACTION

In this first experiment, the students observe the functioning of the in-built PID Controller available in ROBOTC. The system under analysis is comprised of an *Interactive Servo motor* which is controlled by Pulse Width Modulation by the NXT Intelligent Brick. The feedback is provided by the *rotary encoder* present inside the *Interactive Servo motor* which gives 360 counts per single revolution of the motor shaft. The block diagram for the PID controller is shown in Figure 2.1. Using the feedback from the encoders, the “NXT Intelligent Brick” calculates the angular position of the motor (with 1 degree accuracy) and also the instantaneous speed of the motor. The formula used for speed calculation is

$$speed = \frac{\Delta\theta}{\Delta t} \text{degrees/sec} \quad (2.1)$$

The LEGO motors are assigned a power rating between -100 and 100 with a negative value denoting reverse direction. Under ideal conditions, the maximum speed of the LEGO motors is 1000 degrees per second when the power rating is set at 100% . But this value drops with decrease in battery voltage and increase in motor load.

To enable PID Speed Control and set the maximum regulated speed, the following commands were used

```
nMotorPIDSpeedCtrl[motorB] = mtrSpeedReg;
nMaxRegulatedSpeedNxt = 750;
```

The in-built PID controller provides consistent speed by continuously adjusting the raw power sent to the motor.

The LEGO NXT has a *Datalog* feature which allows sensor data and status information to be stored in memory as (Key, Value) pairs, which can later be exported to a PC for post-processing. In this experiment, the encoder counts and Motor PWM level are logged every Δt ms. The ROBOTC code for this experiment is given below as the motive of this experiment is to get the students acquainted with the software and hardware interfaces.

```
task main()
{
    nMaxRegulatedSpeedNxt = 500;
    // Reset the Motor Encoder
    nMotorEncoder[motorB] = 0;
    nMotorPIDSpeedCtrl[motorB] = mtrSpeedReg;
```

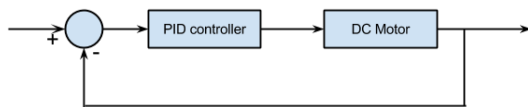


Figure 2.1: Block Diagram of the in-built closed loop PID Controller



Figure 2.2: Experimental setup for observing in-built PID Controller

```

int motorRAWpower, motorDegrees;
motor[motorB] = 50;    // 50/100
time1[T1] = 0;        // Timer

// Allocate memory for datalog
nDatalogSize = 1600;

while (time1[T1] < 10000) {
    motorDegrees = nMotorEncoder[motorB];
    motorRAWpower = motorPWMLevel[motorB];
    // store value to Datalog
    AddToDatalog(1,motorPIDdegrees);
    AddToDatalog(2,motorPIDpower);
    wait1Msec(50);
}
motor[motorB] = 0;    // Stop the motors
SaveNxtDatalog();
}

```

In order to observe the PID control action, the students are asked to run the above program and apply friction on the motor. The in-built PID controller will then step in and increase the motor PWM level in order to maintain the speed. This can also be quantitatively verified from the log file. The datalog file (which would be named as *DATAAnnnn.rdt*) can be exported to a PC using the *File Management Utility* available in the ROBOTC Development Environment under *Robot→NXT Brick→File Management Utility*. The *Spreadsheet Upload* feature transfers the datalog file to the PC and additionally converts the *.rdt* file into a *.csv* file which can then be processed using any Spreadsheet Processors.

As mentioned earlier, the data is stored as *Key, Value* pairs which constitute the two columns in the exported CSV file. The datalog sequentially stores data with every call to the `AddToDatalog(<index>,<data>);` creating a new entry in the file with a *key* which is proportional to the *index* and the *data* stored as a 16-bit unsigned integer. Due to this implicit type casting, negative values would get “wrapped” around. Hence appropriate post-processing has to be done to handle the same.

The students can now plot the speed in degrees per second and motor PWM level in a single graph to observe the PID action. One such plot is shown in Figure 2.3.

As can be seen from the plot, there is a dip in the speed of the motor between 3 and 5 seconds which caused an increase in the PWM power level applied to the motor. When the friction is removed, the speed shoots up due to higher power level before settling down to the desired speed of 250 deg/sec.

2.2 DESIGN AND TUNING OF PID CONTROLLER

how to disable internal PID controller. writing your own PID controller. Zeigler-Nichols tuning Critical damping

2.3 SYSTEM IDENTIFICATION USING BODE PLOTS

brief on Bode plots obtaining bode plots from experimental data system identification using bode plots

2.4 SYSTEM IDENTIFICATION AND VERIFICATION USING MATLAB

Introduction on System Identification. brief on System identification Toolbox in MATLAB Simulink model for identification and verification.

3 CONCLUSIONS FOR ANALYSIS OF LEGO DC MOTOR

summary of concepts learned through this set of experiments comparison to current practices

4 NXTWAY-GS - A TWO WHEELED SELF-BALANCING ROBOT

The second half of the report details the building of a 2-wheel Self-balancing robot using the LEGO NXT kit and a HiTechnic Gyro Sensor. The 2-wheel Self-balancing robot can be modelled as a *2D Inverted pendulum* system which is a well studied classic problem in Control Engineering. This makes it ideal for a 1 month course project in a first year Masters programme in Control Engineering. The analysis is divided into 3 parts - 1. Mathematical Modelling, 2. Servo PID Controller design and 3. Experimental Results. MATLAB provides official Simulink support package for LEGO Mindstorms NXT hardware which allows the students to use the various toolboxes available in Simulink and deploy the model to the LEGO NXT hardware.

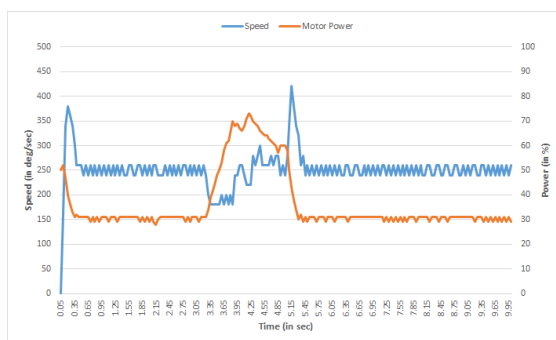


Figure 2.3: Plot of Speed vs PWM level of the LEGO motor

4.1 MODELLING THE 2-WHEEL SELF-BALANCING ROBOT

The 2-wheel Self-balancing Robot can be modelled as a 2-wheeled Inverted Pendulum system as shown in Figure 3.1. The mathematical model of such a system is derived here from first principles. The physical parameters used in the model with their nominal



Figure 4.1: 2-wheeled Inverted Pendulum

values have been tabulated in Table ?? in Appendix B.

The equations of motion for the 2-wheeled inverted pendulum can be derived using the Lagrangian method based on the coordinate axis adopted in Figure 3.1. If direction of two-wheeled inverted pendulum is positive x-axis at $t = 0$, then:

$$(\theta, \phi) = \left(\frac{\theta_l + \theta_r}{2}, \frac{R(\theta_r - \theta_l)}{W} \right) \quad (4.1)$$

Parameter	Nominal Value	Remark
g	$9.81m/sec^2$	Accel. due to Gravity
m	$0.03kg$	Wheel weight
R	$0.04m$	Wheel radius
J_w	$\frac{mR^2}{2}kgm$	Wheel inertia moment
M	$0.6kg$	Body weight
W	$0.14m$	Body width
D	$0.04m$	Body depth
H	$0.144m$	Body height
L	$\frac{H}{2}m$	Dist. of CoM from wheel axle
J_ψ	$\frac{ML^2}{3}kgm^2$	Body pitch inertia moment
J_ϕ	$\frac{M(W^2+D^2)}{12}kgm^2$	Body yaw inertia moment
J_m	$1.5 \times 10^{-5}kgm^2$	DC motor inertia moment
R_m	6.69Ω	DC motor resistance
K_b	$0.468Vsec/rad$	DC motor back EMF constant
K_t	$0.317Nm/A$	DC motor torque constant
n	1	Gear ratio
f_m	0.0022	Friction coeff. b/w body and DC motor
f_w	0	Friction coeff. b/w wheel and floor

4.2 CONTROLLER DESIGN

Linearization about $\psi = 0$ State space representation Servo controller Feedback gain calculations - LQR concepts Integral gain to regulate steady state errors.

4.3 SIMULATION AND VERIFICATION

Simulation results experimental results graphs and diagrams

4.4 FUTURE WORK

Possible extensions to the project

5 CONCLUSIONS FOR TWO-WHEELED SELF-BALANCING ROBOT

Summary of concepts learned through this project. Time frame for the project. Possible alterations

6 APPENDIX

6.1 ZEIGLER-NICHOLS TUNING

- Set K_i and K_d to zero.

- Slowly increase K_p to a value K_u at which sustained oscillations - constant amplitude and periodic - are observed.
- Note period of oscillation
- Refer table below for initial values

Z-N Model	K_p	K_i	K_d
P controller	$0.5 \cdot K_u$	0	0
PI controller	$0.455 \cdot K_u$	$0.833 \cdot T_u$	0
PID controller	$0.588 \cdot K_u$	$0.5 \cdot T_u$	$0.125 \cdot T_u$

Since Z-N method usually results in aggressive tuning, alternatively Tyreus-Luyben method can be adopted.

T-L Model	K_p	K_i	K_d
PI controller	$0.312 \cdot K_u$	$2.2 \cdot T_u$	0
PID controller	$0.454 \cdot K_u$	$2.2 \cdot T_u$	$0.159 \cdot T_u$