PID CONTROL OF BRUSHLESS DC MOTOR AND ROBOT TRAJECTORY PLANNING AND SIMULATION WITH MATLAB®/SIMULINK®

This report presents a PID model of a brushless dc (BLDC) motor and a robot trajectory planning and simulation.
이 보고서는 브러시리스 DC (BLDC) 모터의 PID 모델과 로봇 궤도 계획 및 시뮬레이션을 제공합니다.

A short description of the brushless dc motor is given.
브러시리스 DC 모터에 대한 간단한 설명이 나와 있습니다.

For this work, mathematical models were developed and subsequently used in getting the simulation parameters.
이 작업에서는 수학적 모델을 개발하여 시뮬레이션 매개 변수를 가져 오는 데 사용했습니다.

The PID model is accomplished with the use of MATLAB®/SIMULINK®.
PID 모델은 MATLAB® / SIMULINK®를 사용하여 수행됩니다.

The operational parameters of the specific BLDC motor were modelled using the tuning methods which are used to develop subsequent simulations.
특정 BLDC 모터의 작동 매개 변수는 후속 시뮬레이션을 개발하는 데 사용되는 튜닝 방법을 사용하여 모델링되었습니다.

The best PID parameters were thereafter used for the robot trajectory and simulation over a football pitch model.
최상의 PID 파라미터는 이후 축구 피치 모델에 대한 로봇의 궤적과 시뮬레이션에 사용되었습니다.

INTRODUCTION 소개

The use of the general type dc motors has its long history.
일반형 DC 모터의 사용은 오랜 역사를 가지고 있습니다.

It has been used in the industries for many years now.
그것은 여러 해 동안 산업에서 사용되어 왔습니다.

They provide simple means and precise way of control [1].
그들은 간단한 수단과 정확한 제어 방법을 제공합니다 [1].

In addition, they have high efficiency and have a high starting torque versus falling speed characteristics which helps high starting torque and helps to prevent sudden load rise [2].

또한 높은 효율을 가지며 낮은 시동 토크와 높은 시동 토크를 제공하여 급격한 부하 상승 방지에 도움을줍니다 [2].

But with such characteristics, the dc motors have some deficiencies that needed to be attended to which gave rise to design of some other alternative types of dc motors.

그러나 이와 같은 특성으로 인해서 직류 전동기는 직면해야 할 몇 가지 결함이있어 다른 직류 전동기의 설계가 가능합니다.

For example, the lack of periodic maintenance, mechanical wear outs, acoustic noise, sparkling, brushes effects are some of the problems that were needed to overcome the defects in dc motors.

예를 들어,주기적인 유지 보수, 기계적 마모, 음향 잡음, 스파클링, 브러시 효과의 부족은 DC 모터의 결함을 극복하는 데 필요한 몇 가지 문제입니다.

As a result, emphatic studies have been made on synchronous dc motors with brushless commutators.

결과적으로, 브러시리스 (brushless) 정류자를 갖는 동기식 직류 모터에 대한 연구가 많이 이루어졌다.

So, current researches have been tailored towards developing brushless direct current motors, which are fast becoming alternatives to the conventional dc motor types.

따라서, 현재의 연구는 브러시리스 직류 모터의 개발에 맞춰져 있으며, 이는 종래의 직류 모터의 대안이되고있다.

The BrushLess Direct Current (BLDC) motors are gaining grounds in the industries, especially in the areas of appliances production, aeronautics, medicine, consumer and industrial automations and so on.

BrushLess Direct Current (BLDC) 모터는 특히 가전 제품 생산, 항공, 의학, 소비자 및 산업 자동화 분야에서 산업에서의 기반을 확보하고 있습니다.

The BLDC are typically permanent synchronous motors, they are well driven by dc voltage.

BLDC는 일반적으로 영구 동기 모터이며 DC 전압에 의해 잘 구동됩니다.

They have a commutation that is done mainly by electronics application.

그들은 주로 전자 응용 프로그램에 의해 행해지는 정류를 가지고 있습니다.

Some of the many advantages of a brushless dc motor over the conventional “brushed dc motors are highlighted below [3]:

기존 브러시 DC 모터에 비해 브러시리스 DC 모터의 많은 장점 중 일부는 다음과 같이 강조됩니다.

Better speed versus torque characteristics더 나은 속도 대 토크 특성
High dynamic response높은 동적 응답
High efficiency고효율
Long operating life긴 작동 수명
Noiseless operation소음없는 작동

Higher speed ranges높은 속도 범위
Low maintenance (in terms of brushes cleaning; which is peculiar to the brushed dc motors).
낮은 유지 보수 (브러시 세척과 관련하여 브러시가 달린 DC 모터 특유 임).
Another vital advantage is that the ratio of torque delivered to the size of the motor is higher, and this contributes to its usefulness in terms of space and weight consideration.
또 다른 중요한 이점은 모터의 크기에 전달되는 토크의 비율이 더 높다는 것이고, 이는 공간 및 중량 고려면에서 유용함을 의미합니다.

The BLDC motors come in different phases, for example, single phase, double-, and triple-types.
BLDC 모터는 단상, 이중 및 삼중 유형과 같이 서로 다른 단계로 제공됩니다.

In depth discussion would not be made in this regards, but the most commonly used of all these is the three phase type.
깊이있는 토론은이 점에서 이루어지지 않을 것입니다. 그러나이 모든 것들 중에서 가장 보편적으로 사용되는 것은 3 상 유형입니다.

For this purpose, a brief perspective will be considered on how the BLDC motors could be compensated in terms of control and stability.
이를 위해 제어 및 안정성면에서 BLDC 모터를 어떻게 보완 할 수 있는지에 대한 간단한 관점을 고려해야합니다.

Therefore, this report would presents a theoretical background of DC and BLDC motors, design of simple model of basic DC motors tailored towards developing a BLDC motor model.
따라서 이 보고서는 BLDC 모터 모델 개발에 적합한 기본 DC 모터의 간단한 모델을 설계하는 DC 및 BLDC 모터의 이론적 배경을 제시합니다

In addition, a brief introduction of a very essential tool of stability determinant would also be discussed under “PID auto-tuning”.
또한, 안정성 결정기의 필수 도구에 대한 간략한 소개는 "PID 자동 튜닝"에서 논의 될 것입니다.

Thereafter, a MATLAB®/SIMULINK® model of the BLDC motor would also be reported accordingly.
그 후 BLDC 모터의 MATLAB® / SIMULINK® 모델도 그에 따라보고됩니다.

The PID controller is applied in various fields of engineering, and it is also a very important tool in telecommunication system.
PID 컨트롤러는 다양한 엔지니어링 분야에 적용되며 통신 시스템에서 매우 중요한 도구입니다.

If there is a system and stability is desired, then PID could be very useful.
시스템이 있고 안정성이 필요한 경우 PID가 매우 유용 할 수 있습니다.

A simple systematic approach to these tasks is given in chapter format as given below.
이러한 작업에 대한 간단한 체계적인 접근 방식은 아래에 주어진 장 형식으로 제공됩니다.

The chapters 2 and 3 present the “DC motor and design concepts” while chapter 4 gives a brief introduction into the Brushless DC motor and its model concept.
2 장과 3 장은 "DC 모터와 설계 개념"을 제시하고 4 장은 브러시리스 DC 모터와 모델 개념에 대해 간략하게 소개합니다.

It also elaborates the basic concept of their mathematical representations in simple format.
또한 수학 표현의 기본 개념을 간단한 형식으로 자세히 설명합니다.

The particular BLDC motor used is a maxon motor and chapters 5 – 7 present the whole modelling idea of this specific motor and the open loop response analysis was also included as part of the pre-analysis needed for the subsequent control.
사용 된 특정 BLDC 모터는 맥슨 모터이며, 5 - 7 장 은 이 특정 모터의 전체 모델링 아이디어를 제시하고 개방 루프 응답 분석은 후속 제어에 필요한 사전 분석의 일부로 포함되었습니다.

Also, the idea of the PID (Proportional-Integral-Derivative) controller and its design concepts, control mechanism and tuning methods are presented under chapters 8 and 9.
또한 PID (Proportional-Integral-Derivative) 컨트롤러와 그 설계 개념, 제어 메커니즘 및 튜닝 방법에 대한 아이디어는 8 장과 9 장에 제시되어 있습니다.

Chapters 10 – 12 present the work done on the robot trajectory planning and simulation.
10 장부터 12 장까지는 로봇 궤도 계획 및 시뮬레이션에서 수행 된 작업을 나타냅니다.

The chapter 10 was used to elaborate the required standard football pitch layout model; chapter 11, for the analysis and computation for the robot four-wheeled motors and the chapter 12 gives the planning stages and corresponding coding schemes.
10 장은 필요한 표준 축구 피치 레이아웃 모델을 정교하게 만드는 데 사용되었습니다.
11 장은 로봇 4 륜 모터의 해석과 계산에 사용되며, 12 장에서는 계획 단계와 해당 코딩 체계에 대해 설명합니다.

The results analysis and discussion is presented under the 13th chapter; and finally the chapter fourteen focuses on the conclusion, challenges and recommendation and possible improvement needed in future works.
결과 분석과 토론은 제 13 장 아래 제시됩니다.
마지막으로 14 장은 결론, 도전 과제 및 권장 사항에 초점을두고 향후 작업에 필요한 개선 가능성을 제시합니다.

DC MOTOR

2.1 DC motors
DC 모터

A brief illustration and mathematical representation of DC motors will be discussed in the section based on the general concepts of electromagnetic induction.
DC 모터의 간단한 그림과 수학적 표현은 전자기 유도의 일반적인 개념에 기초한 절에서 논의 될 것입니다.

The DC motors are made of a number of components; some of which are [1]:
DC 모터는 여러 구성 요소로 이루어져 있습니다. 그 중 일부는 다음과 같습니다 [1] :

Frame 틀
Shaft 샤프트
Bearings 문장
Main field windings (Stator) 주요 자계권선(고정자)
Armature (Rotor) 전기자(로터)
Commutator 정류기
Brush Assembly 브러시 어셈블리 -DC와BLDC의 주요 차이점이다.
The most important part of these components that needs detail attention is the main field and the rotating windings (the stator and the rotor respectively).
세심한주의가 필요한 이들 부품의 가장 중요한 부분은 메인 필드와 회전 권선 (각각 고정자와 회 전자)입니다.

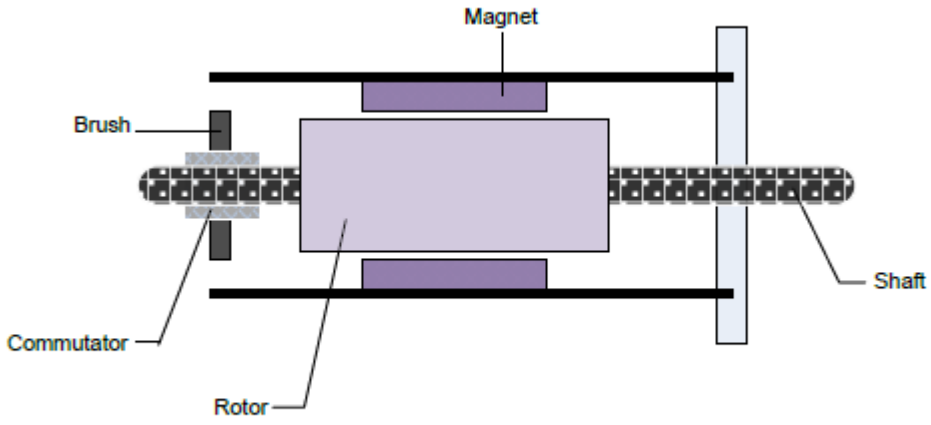

Figure 2.1 – Sectional illustration of a DC motor [2]

As shown in figure 2.1, the stator is formed by the metal carcass with a permanent magnet enclosure which a magnetic field inside the stator windings.
그림 2.1에서 보듯이, 고정자(stator)는 고정자 권선 내부에 자기장이있는 영구 자석 인클로저가있는 금속 중심재에 의해 형성됩니다.
==========
At one of the ends is the brush mountings and the brush gear which are used for electrical contacts with the armature (the rotor).
단부 중 하나에는 전기자 (로터)와의 전기 접촉에 사용되는 브러시 장착 및 브러시 기어가 있습니다.

The field windings are mounted on the poles pieces to create electromagnetism.
계자 권선(고정자)은 전자기를 생성하기 위해 극 조각에 장착됩니다.

The strength of this electromagnetic field is determined by the extent of interaction between the rotor and the stator.
이 전자기장의 강도는 회 전자(rotor)와 고정자 사이의 상호 작용의 정도에 의해 결정됩니다.

Also, the brushes serve as the contact- piece for the commutator to provide electrical voltage to the motor.
또한 브러시는 정류자가 모터에 전압을 공급하기위한 접점 역할을합니다.

Consistent dirt on the commutator causes disruption in the supply of dc voltage, which creates a number of maintenance applications.
정류자(commutator)에 일관된 먼지가 쌓이면 dc 전압 공급이 중단되어 많은 유지 관리 응용 프로그램이 만들어집니다.

This sometimes could lead to corrosion and sometimes sparks between the carbon made brushes and the commutator.
이것은 때로는 부식을 일으킬 수 있고 때로는 카본 제 브러시와 정류자 사이에서 불꽃을 일으킬 수 있습니다.

One of the major challenges is the control of the speed (speed precision); but this could be done by varying the applied voltage.
주요 과제 중 하나는 속도 (속도 정밀도) 제어입니다. 그러나 이것은 인가 된 전압을 변화시킴으로써 수행 될 수있다.

Varying the supply voltage might involve the use of a variable resistor (or a rheostat) which will be connected in tandem with the armature to form a series connection.
공급 전압을 변화 시키려면 일련의 연결을 형성하기 위해 전기자와 직렬로 연결될 가변 저항기 (또는 가변 저항기)를 사용해야합니다.

But this kind of arrangement is not efficient enough as a result of power dissipation.
그러나 이런 배열은 전력 낭비의 결과로는 충분히 효율적이지 못합니다.

In recent times, solid state electronics has made its implication in this regard through the use of controlled rectifiers and choppers.

최근에는 솔리드 스테이트 전자 장치가 제어 정류기 및 초퍼의 사용을 통해 이러한 측면에서 의미가 있습니다.

This arrangement could be efficient as they are used for highly efficient varying dc voltage.
이러한 배열은 고효율 가변 직류 전압에 사용되기 때문에 효율적일 수 있습니다.

In most cases, the most commonly used device is the thyristor (this allows for voltage variation by varying the firing angle of the thyristor in question) [4].
대부분의 경우, 가장 일반적으로 사용되는 장치는 사이리스터(전류 제어기능을지닌 반체소자)입니다
(이는 문제의 사이리스터의 점등 각도를 변경하여 전압 변동을 허용합니다) [4].

Consider the simple arrangement in figure 2.2.
그림 2.2의 간단한 배치를 고려하십시오.

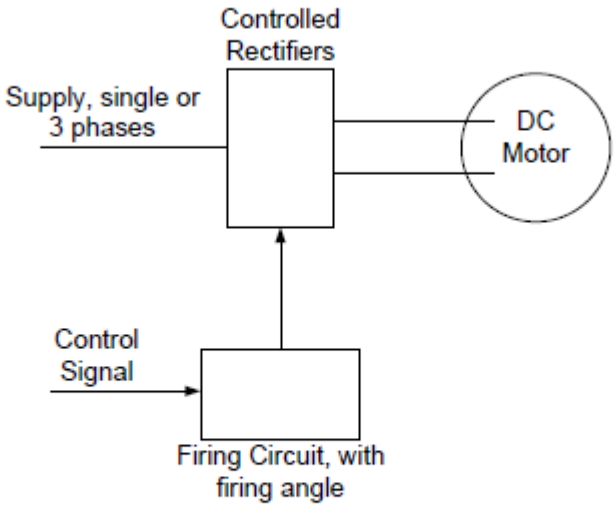

Figure 2.2 – A dc motor operation with a thyristor arrangement using the thyristor firing angle to vary the dc voltage [4].

그림 2.2 - 직류 전압을 변화시키기 위해 사이리스터 점화 각도를 사용하는 사이리스터 배열을 사용하는 직류 모터 작동 [4].

DC MOTOR MODEL
DC 모터 모델

Mathematical model of a typical DC motor
일반적인 DC 모터의 수학적 모델

A typical dc motor equivalent circuit is illustrated as shown in the circuit shown below in figure 3.1 and figure 3.2:

일반적인 직류 모터 등가 회로는 그림 3.1 및 그림 3.2에 표시된 회로와 같이 표시됩니다.

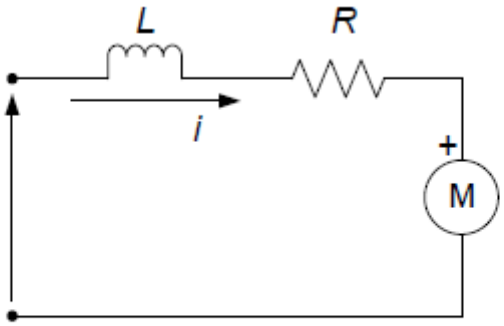

Figure 3.1 – A typical DC motor equivalent electrical circuit.

그림 3.1 - 일반적인 DC 모터와 동등한 전기 회로.

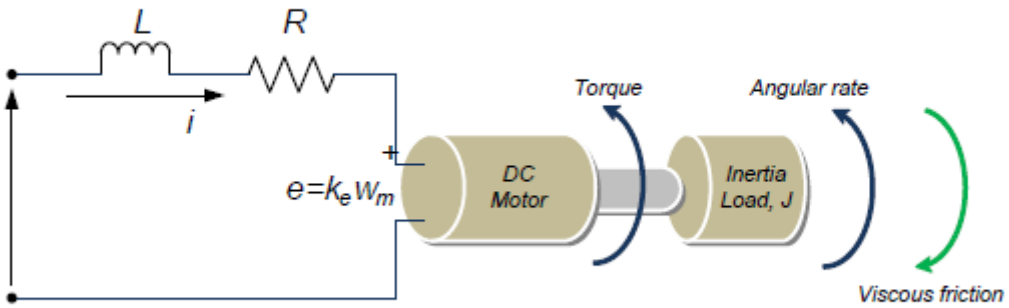

Figure 3.2 – A typical DC motor electromechanical system arrangement.

그림 3.2 - 일반적인 DC 모터 전기 기계 시스템 배치.

The basic component represented are the armature resistance, R and the armature inductance L; in addition, there is the back emf, e.
표시된 기본 구성 요소는 전기자 저항, R 및 전기자 인덕턴스 L입니다. 또한, 다시 emf, e.

From the in figure 3.1 and figure 3.2 above, the following equations are used to describe the relationship of operation.
위의 그림 3.1과 그림 3.2에서 다음 관계식을 사용하여 작업 관계를 설명합니다.

Using the Kirchhoff's Voltage Law, KVL, the following equation 3.1 is obtained:
Kirchhoff의 전압 법칙 KVL을 사용하여 다음 식 3.1을 얻습니다.

$$V_s = Ri + L\frac{di}{dt} + e \qquad (3.1)$$

At steady state (DC state of zero-frequency), $V_s = Ri + e.$
정상 상태 (제로 - 주파수의 DC 상태)에서,

Therefore, for the non steady-state, equation 3.1 is rearranged to make provision for the back emf, as shown in equation 3.2 below:
따라서, 비 정상 상태에 대해, 방정식 3.1은 다음 방정식 3.2와 같이 역전 류에 대한 규정을 만들기 위해 재배치된다.

$$e = -Ri - L\frac{di}{dt} + V_s \qquad (3.2)$$

Where,
□□ = the DC Source voltage DC 소스 전압
i = the armature current 전기자 전류

Similarly, considering the mechanical properties of the dc motor, from the Newton's second law of motion, the mechanical properties relative to the torque of the system arrangement in figure 3.1 and figure 3.2 would be the product of the inertia load, J and the rate of angular velocity, □□ is equal to the sum of all the torques; these follow with equation 3.3 and 3.4 accordingly.
유사하게, 뉴턴의 제 2 운동 법칙으로부터 직류 모터의 기계적 성질을 고려하면 그림 3.1과 그림 3.2에서 시스템 배열의 토크에 상대적인 기계적 특성은 관성 부하 J와 각속도 ω는 모든 토크의 합과 같습니다. 이것들은 식 3.3과 3.4에 따라 적절하게 따른다.

$$J\frac{d\omega_m}{dt} = \sum T_i \qquad (3.3)$$

$$T_e = k_f\omega_m + J\frac{d\omega_m}{dt} + T_L \qquad (3.4)$$

Where,
□□ = the electrical torque전기 토크
□□ = the friction constant 마찰 상수
□ = the rotor inertia 로터 관성
□□ = the angular velocity 각속도
□□ = the supposed mechanical load, 가정 된 기계적 부하
(분석을 위해 0이라고 가정 할 수 있습니다.),
Where the electrical torque and the back emf could be written as:
전기 토크와 역기전력은 다음과 같이 쓸 수 있습니다 :

$$e = k_e\omega_m \text{ and } T_e = k_t\omega_m \qquad (3.5)$$

Where,

□□ = the back emf constant 역기전력 상수

□□ = the torque constant 토크 상수

Therefore, re-writing equations 3.2 and 3.3, the equation 3.6 and 3.7 are obtained,
따라서, 방정식 3.2 및 3.3을 다시 쓰면 방정식 3.6 및 3.7이 얻어지고,

$$\frac{di}{dt} = -i\frac{R}{L} - \frac{k_e}{L}\omega_m + \frac{1}{L}V_s \qquad (3.6)$$

$$\frac{d\omega_m}{dt} = i\frac{k_t}{J} - \frac{k_f}{J}\omega_m + \frac{1}{J}T_L \qquad (3.7)$$

Using Laplace transform to evaluate the two equations 3.6 and 3.7, the following are obtained appropriately (all initial conditions are assumed to be zero): For equation 3.6,
Laplace transform을 사용하여 두 개의 방정식 3.6과 3.7을 평가하면 다음과 같이 적절하게 얻습니다 (모든 초기 조건은 0으로 가정 됨). 방정식 3.6의 경우,

$$\mathcal{L}\left\{\frac{di}{dt} = -i\frac{R}{L} - \frac{k_e}{L}\omega_m + \frac{1}{L}V_s\right\} \quad (3.8)$$

This implies,

$$si = -i\frac{R}{L} - \frac{k_e}{L}\omega_m + \frac{1}{L}V_s \quad (3.9)$$

For equation 3.7,

$$\mathcal{L}\left\{\frac{d\omega_m}{dt} = i\frac{k_t}{J} - \frac{k_f}{J}\omega_m + \frac{1}{J}T_L\right\} \quad (3.10)$$

This implies,

$$s\omega_m = i\frac{k_t}{J} - \frac{k_f}{J}\omega_m + \frac{1}{J}T_L \quad (3.11)$$

At no load (for $T_L = 0$); equation 3.11 becomes:

$$s\omega_m = i\frac{k_t}{J} - \frac{k_f}{J}\omega_m \quad (3.12)$$

From equation 3.12, i is made the subject for a substitute into equation 3.9.

$$i = \frac{s\omega_m + \frac{k_f}{J}\omega_m}{\frac{k_t}{J}} \quad (3.13)$$

$$\left(\frac{s\omega_m + \frac{k_f}{J}\omega_m}{\frac{k_t}{J}}\right)\left(s + \frac{R}{L}\right) = -\frac{k_e}{L}\omega_m + \frac{1}{L}V_s \quad (3.14)$$

Equation 3.14 becomes:

$$\left\{\left(\frac{s^2J}{k_t} + \frac{sk_f}{k_t} + \frac{sRJ}{k_tL} + \frac{k_fR}{k_tL}\right) + \frac{k_e}{L}\right\}\omega_m = \frac{1}{L}V_s \quad (3.15)$$

And equation 3.15 finally resolved to 3.16:

$$V_s = \left\{\frac{s^2JL + sk_fL + sRJ + k_fR + k_ek_t}{k_t}\right\}\omega_m \quad (3.16)$$

The transfer function is therefore obtained as follows using the ratio of and the angular velocity, □□ to source voltage, Vs.
That is,
그러므로 전달 함수는 다음과 같이 ω와 ω의 비율을 이용하여 다음과 같이 얻어진다.
그건,

$$G(s) = \frac{\omega_m}{V_s} = \frac{k_t}{s^2JL + sk_fL + sRJ + k_fR + k_ek_t} \quad (3.17)$$

From these, the transfer function could be derived accordingly as follows: That is,

이들로부터, 전달 함수는 다음과 같이 유도 될 수있다. 즉,

$$G(s) = \frac{\omega_m}{V_s} = \frac{k_t}{s^2JL + (RJ + k_f L)s + k_f R + k_e k_t} \qquad (3.18)$$

Considering the following assumptions:
1.The friction constant is small, that is, □□ tends to 0, this implies that;
2. □□ ≫ □□ □, and
3. □□ □□ ≫□□□

And the negligible values zeroed, the transfer function is finally written as;
그리고 무시할 수있는 값은 제로가되고, 전달 함수는 마침내 다음과 같이 쓰여집니다.

$$G(s) = \frac{\omega_m}{V_s} = \frac{k_t}{s^2JL + RJs + k_e k_t} \qquad (3.19)$$

So by re-arrangement and mathematical manipulation on “JL”, by multiplying top and bottom of equation 3.19 by:
따라서 "JL"에 대한 재 배열과 수학적 조작에 의해 식 3.19의 위와 아래에 다음을 곱하면됩니다.

$$\frac{R}{k_e k_t} \times \frac{1}{R}$$

Equation 3.20 is obtained after the manipulation,
식 3.20은 조작 이후에 얻어지며,

$$G(s) = \frac{\frac{1}{k_e}}{\frac{RJ}{k_e k_t} \cdot \frac{L}{R} \cdot s^2 + \frac{RJ}{k_e k_t} \cdot s + 1} \qquad (3.20)$$

From equation 3.13, the following constants are gotten, The mechanical (time constant),
식 3.13으로부터 다음의 상수들이 얻어진다. The mechanical (time constant),

$$\tau_m = \frac{RJ}{k_e k_t} \qquad (3.21)$$

The electrical (time constant),
전기 (시간 상수),

$$\tau_e = \frac{L}{R} \qquad (3.22)$$

Substituting the equations 3.21 and 3.22 into equation 3.20, it yields;
방정식 3.21과 3.22를 방정식 3.20에 대입하면 방정식 3.20이 나온다.

$$G(s) = \frac{\frac{1}{k_e}}{\tau_m \cdot \tau_e \cdot s^2 + \tau_m \cdot s + 1} \qquad (3.23)$$

4.BRUSHLESS DC MOTOR AND MODEL CONCEPT
브러시리스 DC 모터 및 모델 개념

One of the major differences between the DC motor and the BLDC is implied from the name.
DC 모터와 BLDC의 주요 차이점 중 하나는 이름에서 암시됩니다.

The conventional DC motor has brushes that are attached to its stator while the “brushless” DC motor does not.
기존의 DC 모터에는 "브러시리스"DC 모터가없는 동안 고정자에 부착 된 브러시가 있습니다.

Also, unlike the normal DC motor, the commutation of the BLDC could be done by electronic control [3].
또한 일반적인 DC 모터와는 달리 BLDC의 정류는 전자 제어로 수행 될 수있다 [3].

Under the BLDC motor, the stator windings are energised in sequence for the motor to rotate.
BLDC 모터 아래에서, 고정자 권선은 모터가 회전하도록 순차적으로 활성화됩니다.

More also, there is no physical contact whatsoever between the stator and the rotor.
또한 고정자와 회 전자 사이에는 물리적 접촉이 전혀 없습니다.

Another vital part of the BLDC is the hall sensor(s);
BLDC의 또 다른 중요한 부분은 홀 센서입니다.

these hall sensors are systematically attached to the rotor and they are used as major sensing device by the Hall Effect sensors embedded into the stator [3].
이 홀 센서는 로터에 체계적으로 부착되며 고정자에 내장 된 홀 효과 센서에 의해 주요 감지 장치로 사용됩니다 [3].

This works based on the principle of Hall Effect.
이것은 Hall Effect의 원리를 기반으로 작동합니다.

The BLDC motor operates in many modes (phases), but the most common is the 3-phase.
BLDC 모터는 여러 모드 (위상)에서 작동하지만 가장 일반적인 것은 3 상입니다.
The 3-phase has better efficiency and gives quite low torque.
3 상은 효율이 좋고 토크가 매우 낮습니다.

Though, it has some cost implications, the 3-phase has a very good precision in control [6].
그러나, 이것은 약간의 비용 함의를 가지고 있지만, 3상은 제어에서 매우 우수한 정밀도를 갖는다 [6].

And this is needful in terms of control of the stator current.
그리고 이것은 고정자 전류의 제어 측면에서 필요합니다.
====

4.1 Mathematical model of a typical BLDC motor
일반적인 BLDC 모터의 수학적 모델

Typically, the mathematical model of a Brushless DC motor is not totally different from the conventional DC motor.
일반적으로 브러시리스 DC 모터의 수학적 모델은 기존의 DC 모터와 완전히 다르지 않습니다.

The major thing addition is the phases involved which affects the overall results of the BLDC model.
주요한 사항은 BLDC 모델의 전체 결과에 영향을 미치는 단계입니다.

The phases peculiarly affect the resistive and the inductive of the BLDC arrangement.
이 단계들은 BLDC 장치의 저항성 및 유도성에 특유하게 영향을 미친다.

For example, a simple arrangement with a symmetrical 3-phase and “wye” internal connection could give a brief illustration of the whole phase concept.
예를 들어, 대칭형 3 상 및 "예"내부 연결을 갖는 간단한 배열은 전체 위상 개념에 대한 간단한 설명을 제공 할 수 있습니다.

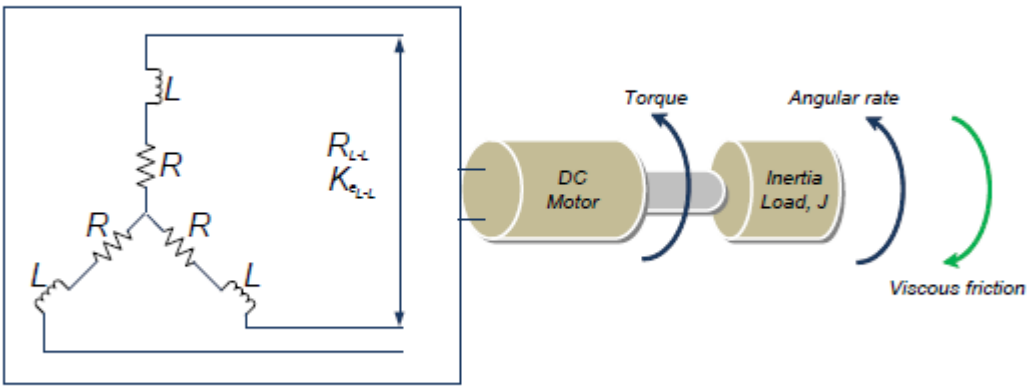

Figure 4.1 – Brushless DC motor schematic diagram

So from the equations 3.20 – 3.22, the difference in the DC and BLDC motors will be shown.
따라서 방정식 3.20 - 3.22에서 DC 및 BLDC 모터의 차이가 표시됩니다.

This difference will affect primarily the mechanical and electrical constants as they are very important parts of modelling parameters.
이 차이는 모델링 매개 변수의 중요한 부분이므로 주로 기계 및 전기 상수에 영향을 미칩니다.

For the mechanical time constant (with symmetrical arrangement), equation 3.21 becomes:
기계식 시정 수 (대칭 배열)의 경우, 식 3.21은 다음과 같습니다.

$$\tau_m = \sum \frac{RJ}{K_e K_t} = \frac{J \sum R}{K_e K_t} \qquad (4.1)$$

The electrical (time constant),
전기 (시간 상수),

$$\tau_e = \sum \frac{L}{R} = \frac{L}{\sum R} \qquad (4.2)$$

Therefore, since there is a symmetrical arrangement and a three phase, the mechanical (known) and electrical constants become:
따라서 대칭 배열과 3상이 있기 때문에 기계적 (알려진) 전기 상수는 다음과 같이됩니다.

Mechanical constant,
기계 상수,

$$\tau_m = \frac{J.3R}{K_e K_t} \qquad (4.3)$$

Electrical constant,
전기적 상수,

$$\tau_e = \frac{L}{3.R} \qquad (4.4)$$

Considering the phase effects,
위상 효과를 고려할 때,

$$\tau_m = \frac{3.R_{\emptyset}.J}{\left(K_{e(L-L)}/\sqrt{3}\right).K_t} \qquad (4.5)$$

Quation 4.5 now becomes:
Quation 4.5는 이제 다음과 같이됩니다

$$\tau_m = \frac{3.R_{\emptyset}.J}{K_e.K_t} \qquad (4.6)$$

Where □□ is the phase value of the EMF (voltage) constant;
여기서 □□는 EMF (전압) 상수의 위상 값입니다.

$$K_e = K_{e(L-L)}/\sqrt{3}$$

Also, there is a relationship between □□ and □□ ; using the electrical power (left hand side) and mechanical power (right hand side) equations;
that is:
또한 □□와 □□ 사이에는 관계가있다. 전력 (왼쪽)과 기계적 힘 (오른쪽) 방정식을 사용하여;
그건:

$$\sqrt{3} \times E \times I = \frac{2\pi}{60} \times N \times T$$

$$\frac{E}{N} = \frac{T}{I} \times \frac{2\pi \times 1}{60 \times \sqrt{3}}$$

$$K_e = K_t \times \frac{2\pi \times 1}{60 \times \sqrt{3}}$$

$$K_e = K_t \times 0.0605 \qquad (4.7)$$

Where,

$K_e = \left[\frac{\mathrm{v-secs}}{\mathrm{rad}}\right]$: the electrical torque

$K_t = \left[\frac{\mathrm{N-m}}{\mathrm{A}}\right]$: the torque constant

□□는 전기토크
□t는 토크상수

Therefore, the equation for the BLDC can now be obtained as follow from equation 3.23 by considering the effects of the constants and the phase accordingly.
따라서 BLDC의 방정식은 상수 및 위상의 영향을 고려하여 식 3.23에서와 같이 얻을 수 있습니다.

$$G(s) = \frac{\frac{1}{K_e}}{\tau_m \cdot \tau_e \cdot s^2 + \tau_m \cdot s + 1} \qquad (4.8)$$

5 MAXON BLDC MOTOR

5.1 Maxon EC 45 flat ø45 mm, brushless DC motor
Maxon EC 45 평형 ø45 mm, 브러시리스 DC 모터

The BLDC motor provided for this thesis is the EC 45 flat ø45 mm, brushless, 30 Watt from Maxon motors [8].
이 논문을 위해 제공된 BLDC 모터는 Maxon 모터에서 30 와트의 EC 45 flat ø45 mm이며 브러시리스 모터이다 [8].

The order number of the motor is 200142.
모터의 주문 번호는 200142입니다.

The parameters used in the modeling are extracted from the datasheet of this motor with corresponding relevant parameters used.
모델링에 사용 된 파라미터는 이 모터의 데이터 시트에서 사용 된 해당 파라미터를 사용하여 추출됩니다.

Find below in Table 5.1 the major extracted parameters used for the modeling task.
아래 표 5.1에서 모델링 작업에 사용 된 주요 추출 매개 변수를 찾습니다.

	Maxon Motor Data	Unit	Value
	Values at nominal voltage		
1	Nominal Voltage	V	12.0
2	No load Speed	rpm	4370
3	No load Current	mA	151
4	Nominal Speed	rpm	2860
5	Nominal Torque (max. continuous torque)	mNm	59.0
6	Nominal Current (max. continuous current)	A	2.14
7	Stall Torque	mNm	255
8	Starting Current	A	10.0
9	Maximum Efficiency	%	77
	Characteristics		
10	***Terminal Resistance phase to phase***	Ω	1.20
11	***Terminal Inductance phase to phase***	mH	0.560
12	***Torque Constant***	mNm/A	25.5
13	***Speed Constant***	rpm/V	37.4
14	Speed/Torque Gradient	rpm/mNm	17.6
15	***Mechanical time constant***	ms	17.1
16	***Rotor Inertia***	gcm^2	92.5
17*	Number of phases		3

Table 5.1 – BLDC motor parameters used [8]

BLDC Maxon 모터 수학 모델

The mathematical model of the BLDC motor is modelled based on the parameters from table 5.1 using the equation 4.23. This is illustrated below:
BLDC 모터의 수학적 모델은 방정식 4.23을 사용하여 표 5.1의 파라미터를 기반으로 모델링됩니다. 아래에 그 예가 나와 있습니다.

$$G(s) = \frac{\frac{1}{K_e}}{\tau_m \cdot \tau_e \cdot s^2 + \tau_m \cdot s + 1} \qquad (6.1.)$$

So the values for □□ , □□ and □□ need to calculated to obtain the motor model. From equation 4.4,
그래서 모터 모델을 얻기 위해 □□, τ□ 및 τ values의 값을 계산해야합니다. 식 4.4로부터,

$$\tau_e = \frac{L}{3.R}$$

$$\tau_e = \frac{0.560 \times 10^{-3}}{3 \times 1.20}$$

$$\boldsymbol{\tau_e} = 155.56 \times 10^{-6} \qquad (6.2.)$$

But □□ is a function of R, J, □□ and □t , 그러나 □□는 R, J,□□ 및 □t의 함수이다.
Where,
R = □ø = 1.2 Ω;

$$J_{Rotor} = 92.5\ \text{gcm}^2 = 9.25 \times 10^{-6}\ \text{Kgm}^2;$$

$$K_t = 25.5 \times 10^{-3}\text{Nm/A}$$

□□ = 0.0171 secs
From equation 4.6, □□ could be obtained:
That is,
방정식 4.6으로부터 □□가 얻어 질 수있다 : 즉,

$$\boldsymbol{\tau_m} = \frac{3.R_{ø}.J}{K_e.K_t} = 0.0171$$

$$\boldsymbol{K_e} = \frac{3.R_{ø}.J}{\tau_m.K_t} = \frac{3 \times 1.2 \times 9.25 \times 10^{-6}}{0.0171 \times 25.5 \times 10^{-3}} = 0.0763\ \frac{\text{v} - \text{secs}}{\text{rad}}$$

Therefore, the G(s) becomes:
따라서 G (s)는 다음과 같이됩니다.

$$G(s) = \frac{13.11}{155.56 \times 10^{-6} \times 0.0171 \cdot s^2 + 0.0171 \cdot s + 1}$$

$$G(s) = \frac{13.11}{2.66 \times 10^{-6} \cdot s^2 + 0.0171 \cdot s + 1} \qquad (6.3.)$$

The G(s) derived above in the equation 6.3 is the open loop transfer function of the Brushless DC maxon motor using all necessarily sufficient parameters available.
방정식 6.3에서 위에 파생 된 G (s)는 사용 가능한 모든 필수 매개 변수를 사용하여 브러시리스 DC 맥슨 모터의 개방 루프 전달 함수입니다.

7 OPEN LOOP ANALYSIS OF THE MAXON MOTOR MODEL

The open loop analysis would be done using the MATLAB®/SIMULINK®.
개방 루프 분석은 MATLAB® / SIMULINK®를 사용하여 수행됩니다.

And the corresponding stability analysis is given likewise to see the effect thereafter when there is closed loop system incorporation.
그리고 유사한 안정성 분석은 폐쇄 루프 시스템 통합이있을 때 그 이후의 효과를보기 위해 마찬가지로 주어집니다.

7.1 Open Loop Analysis using MATLAB m-file
MATLAB m- 파일을 사용한 개방 루프 분석

With the aid of the BLDC motor parameters provided, the open loop analysis is done by considering the stability factors and making the necessary plots for this analysis.
제공된 BLDC 모터 파라미터의 도움으로 개방 루프 분석은 안정성 요인을 고려하고 이 분석에 필요한 플롯을 작성함으로써 수행됩니다.

Some of the plots include the step response, root locus, nyquist diagram, and bode plot diagram.
플롯 중 일부는 스텝 응답, 근원 궤적, 나이 퀴 스트 다이어그램 및 보드 선도 도표를 포함합니다.

For this, separate m-files were created for the constants, evaluated constants and the main files
이를 위해 상수, 평가 된 상수 및 주 파일에 대해 별도의 m 파일이 생성되었습니다

constants.m

```
%
% Start of code
% Maxon flat motor parameters used in the modeling
%
% Characteristics parameters
R = 1.2;          % Ohms, Terminal Resistance phase to phase
L = 0.560e-3;     % Henrys, Terminal Inductance phase to phase
Kt = 25.5e-3;     % Nm/A, Torque constant
Ks = 37.4         % rpm/V, Speed constant
tm = 17.1e-3;     % seconds, s, Mechanical Time constant
J = 92.5e-7;      % kg.m^2, Rotor inertia, given in gcm^2
p = 3;            % Number of phases
%
% End of code
```

evaluatedconstants.m

```
%
% Start of code
%
% Evaluated parameters not given
%
constants
te = L/(p*R);                 % seconds, s, Electrical Time constant
Ke = (3*R*J)/(tm*Kt);         % Back emf constant
% End of code
```

topenloop.m

```
%
% Start of code
%
% includes constant parameters
constants

% includes evaluated constants
evaluatedconstants

% Transfer function
G = tf([1/Ke],[tm*te tm 1]);

% Plots the Step Response diagram
figure;
step(G, 0.5);
title('Open Loop Step Response diagram');
xlabel('Time, secs')
ylabel('Voltage, volts')
grid on;

% plots the Root-locus
figure;
rlocus(G);
title('Open Loop Root Locus diagram');
grid on;

% plots the Nyquist diagram
figure;
nyquist(G);
title('Open Loop Nyquist diagram');
grid on;

% plots the Bode Plot
figure;
bode(G);
title('Open Loop Bode plot diagram 1');
grid on;

% plots the Bode Plot
figure;
bode(G,{0.1 , 100})
title('Open Loop Bode plot diagram with wider frequency spacing');
grid on;

% plots the Bode Plot
figure;
GD = c2d(G, 0.5)
bode(G,'r', GD,'b--')
title('Open Loop Bode plot diagram with discretisied response');
grid on;
% End of code
```

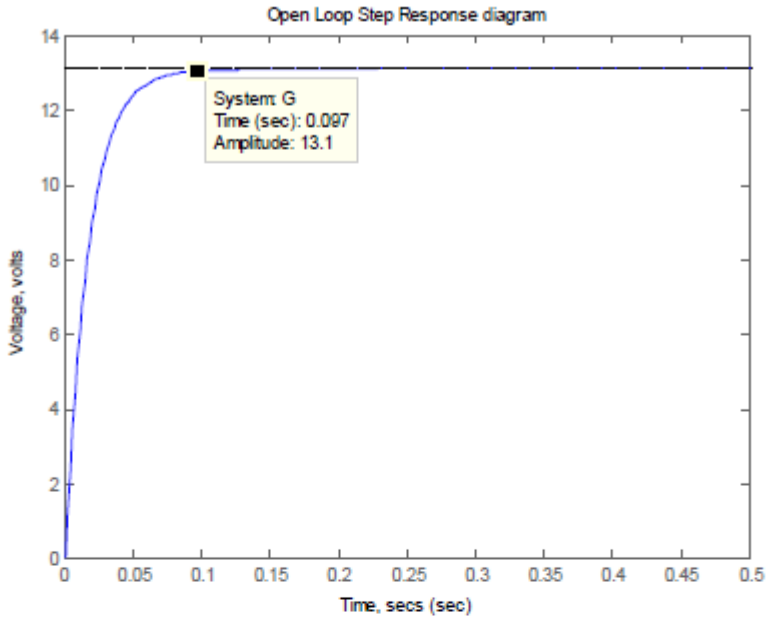

Figure 7.1 – Open Loop Step Response

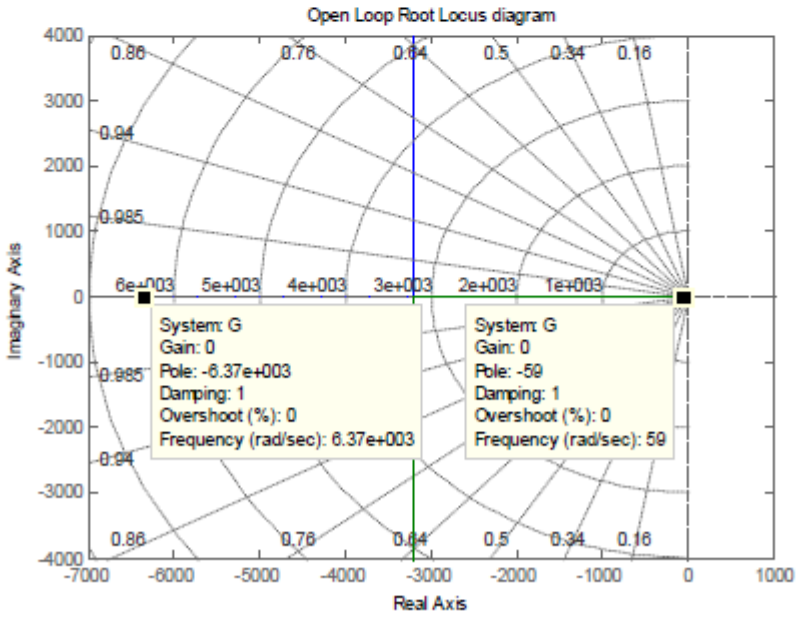

Figure 7.2 – Open Loop Step Root Locus with Gain = 0, Overshoot % = 0 and Damping = 1 for both poles

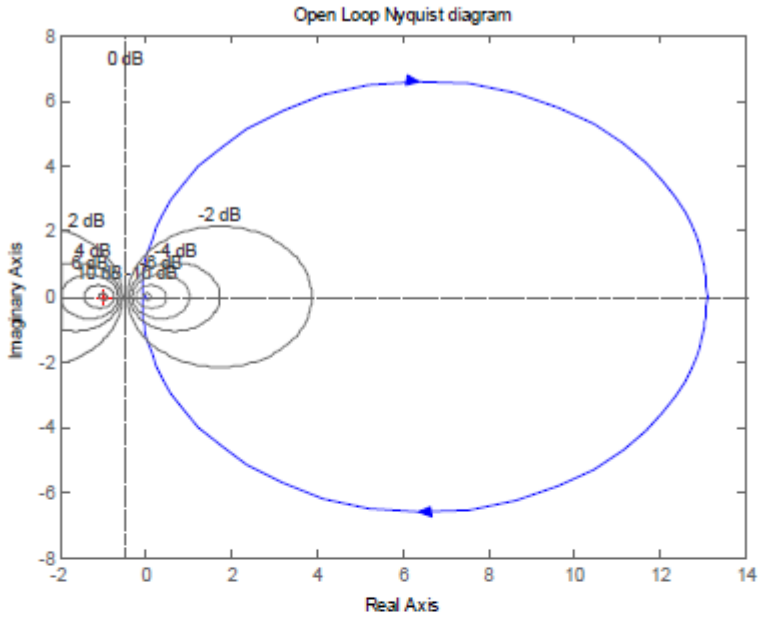

Figure 7.3 – Open Loop Step Nyquist Diagram

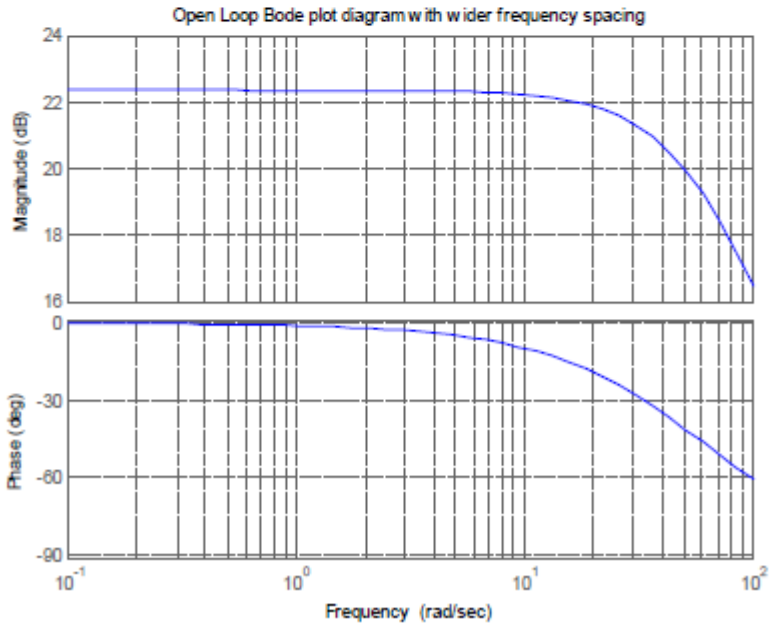

Figure 7.4 – Open Loop Step Bode Plot Diagram

7.2 Open Loop Analysis using SIMULINK

SIMULINK를 이용한 개방 루프 분석

Alternatively, the open loop step response could be done by using the SIMULINK tools as shown in figure 7.5 below.
다른 방법으로는 그림 7.5와 같이 SIMULINK 도구를 사용하여 개방 루프 스텝 응답을 수행 할 수 있습니다.

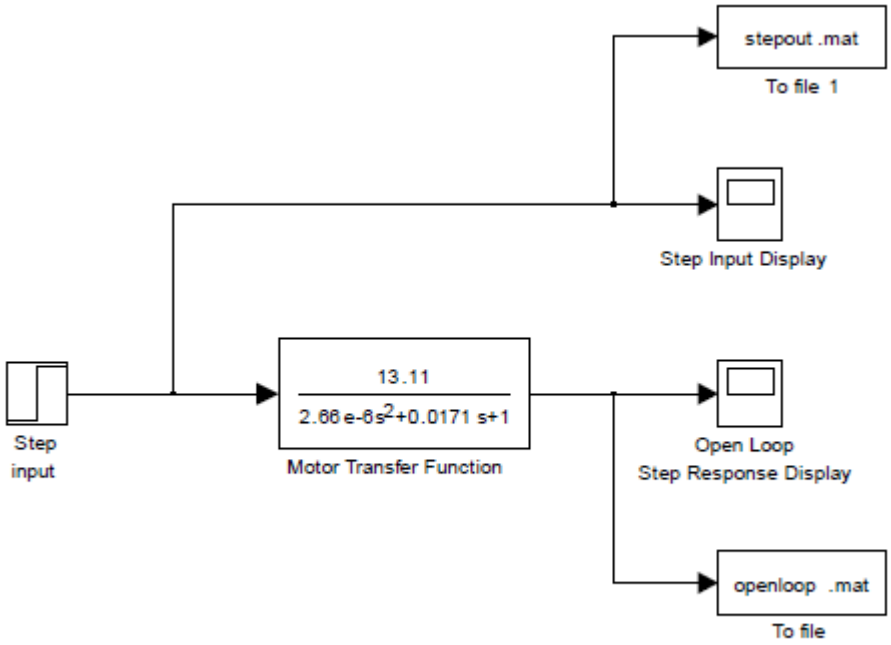

Figure 7.5 – Open loop step response simulink arrangement

From the simulation of figure 7.5 and using a step input of at t=1, the following were obtained.
그림 7.5의 시뮬레이션과 t = 1에서의 스텝 입력을 사용하여 다음을 얻었다.

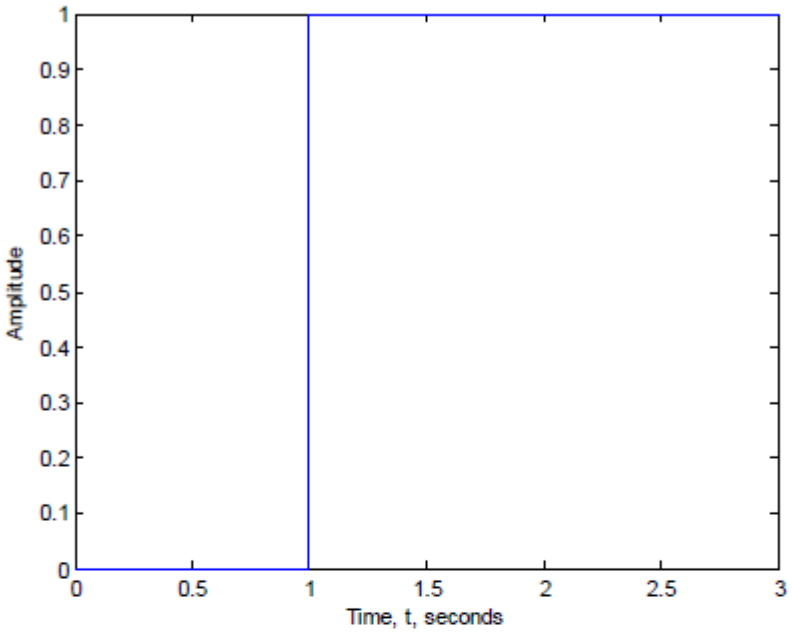

Figure 7.6 – Step input for the open loop simulink arrangement (at t=1)

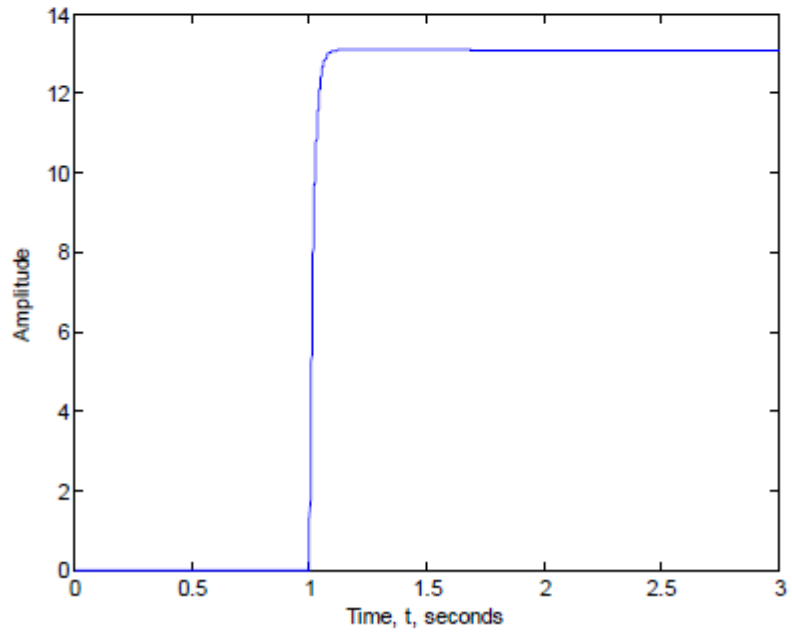

Figure 7.7 – Open loop step response output for the simulink arrangement

With the step response moved to 0.05 for a better display, a joint output of the step input and open loop step response was simulated to give figure 7.8 below.
보다 나은 디스플레이를 위해 스텝 응답을 0.05로 이동 시키면 스텝 입력 및 오픈 루프 스텝 응답의 조인트 출력이 시뮬레이션되어 그림 7.8과 같습니다.

This shows the effect of the system model on the step input.
이것은 스텝 입력에 대한 시스템 모델의 효과를 보여줍니다.

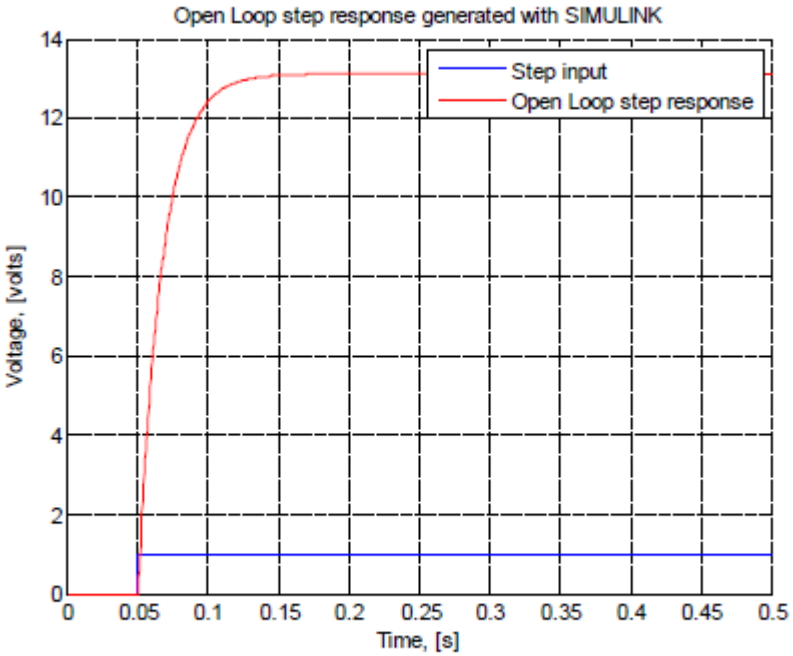

Figure 7.8 – Combined step input and open loop step response span over t=0.5 s

8 PID DESIGN CONCEPT
PID 설계 개념

The Proportional-Integral-Derivative (PID) controller is about the most common and useful algorithm in control systems engineering [7].
Proportional-Integral-Derivative (PID) 제어기는 제어 시스템 엔지니어링에서 가장 보편적이며 유용한 알고리즘입니다 [7].

In most cases, feedback loops are controlled using the PID algorithm.
대부분의 경우 피드백 루프는 PID 알고리즘을 사용하여 제어됩니다.

The main reason why feedback is very important in systems is to be able to attain a set-point irrespective of disturbances or any variation in characteristics of any form.
시스템에서 피드백이 매우 중요한 주된 이유는 모든 형태의 특성의 교란 또는 임의의 변동에 관계없이 설정 점을 달성 할 수 있어야한다는 것입니다.

The PID controller is always designed to correct error(s) between measured process value(s) and a particular desired set-point in a system.
PID 컨트롤러는 항상 측정 된 프로세스 값과 시스템의 특정 설정 점 사이의 오류를 수정하도록 설계되었습니다.

A simple illustration on how the PID works is given below:
PID 작동 방식에 대한 간단한 그림이 아래에 나와 있습니다.

Consider the characteristics parameters – proportional (P), integral (I), and derivative (D) controls, as applied to the diagram below in figure 8.1, the system, S is to be controlled using the controller, C;
아래의 그림 8.1에 적용된 특성 매개 변수 - 비례 (P), 적분 (I) 및 미분 (D) 제어를 고려하면, 시스템은 컨트롤러를 사용하여 제어됩니다.

where controller, C efficiency depends on the P, I and D parameters [8].
제어기, C 효율은 P, I 및 D 파라미터에 의존한다 [8].

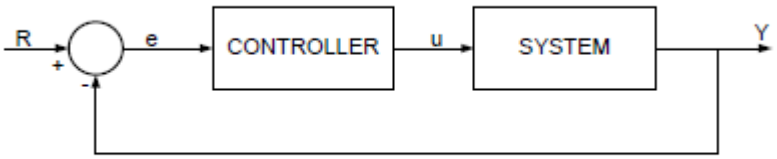

Figure 8.1 – A typical system with a controller [8]

The controller provides the excitation needed by the system and it is designed to control the overall behaviour of the system.
제어기는 시스템에 필요한 구동을 제공하며 시스템의 전체 동작을 제어하도록 설계되었습니다.

The PID controller has several categories of structural arrangements.
PID 제어기는 몇 가지 구조 배열 범주를 가지고 있습니다.

The most common of these are the series and parallel structures and in some cases, there are the hybrid form of the series and the parallel structures.
이들 중 가장 공통적 인 것은 직렬 및 병렬 구조이고 경우에 따라 시리즈 및 병렬 구조의 하이브리드 형식이 있습니다.

The following shows the typical illustrative diagrams of common PID controller structures.
다음은 일반적인 PID 컨트롤러 구조의 전형적인 설명도를 보여줍니다.

Typically, the function of the form shown in equation 8.1 is applicable in this kind of PID controller design.
전형적으로, 식 8.1에 나타낸 형태의 함수는 이러한 종류의 PID 제어기 설계에 적용 가능합니다.

$$K_P + \frac{K_I}{s} + K_D \cdot s = \frac{K_D s^2 + K_P s + K_I}{s} \qquad (8.1)$$

Where,

K_P = Proportional gain

K_I = Integral gain

K_D = Derivative gain

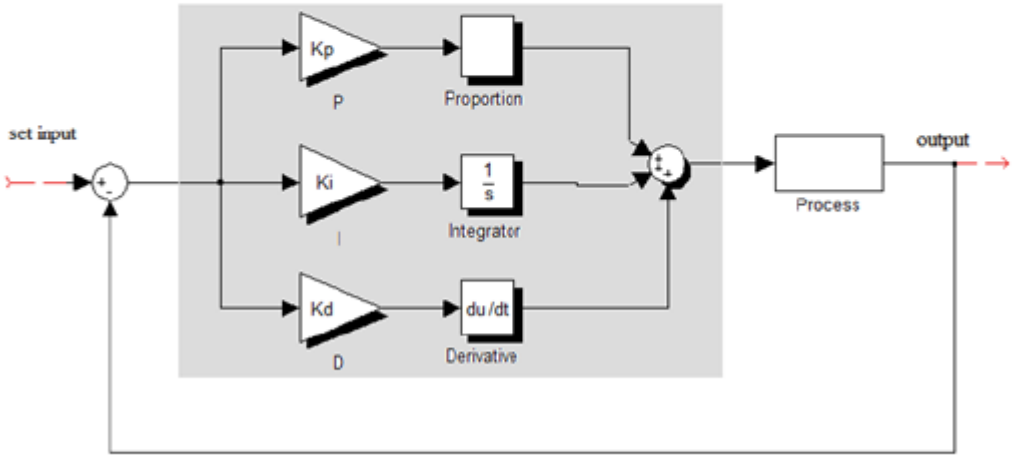

Figure 8.2 – PID parameters schematics

==========

Considering the figure 8.1, variable, e is the sample error, and it is the difference between the desired input value, R and the actual output, Y.
그림 8.1을 보면 변수 e는 표본 오차이며, 이는 원하는 입력 값 R과 실제 출력 Y의 차이입니다.

In a closed loop, e will be sent to the controller, and the controller will perform the integral and derivative computation on the error signal.
폐 루프에서, e는 컨트롤러로 보내지고 컨트롤러는 에러 신호에 대한 적분 및 미분 계산을 수행합니다.

Thereafter, the signal, u which is the output of the controller is now equal to the sum of [the product of proportional gain, KP and the magnitude of the error], [the product of the integral gain, KI and the integral of the error] and [the product of the derivative gain, KD and the derivative of the error].
That is,
그 후 제어기의 출력 인 신호 u는 [비례 게인의 곱, KP와 오차의 크기]의 합과 동일하게됩니다. [적분 게인의 곱 KI와 오차]와 [미분 게인과 KD 및 미분의 미분의 곱]을 사용합니다.
그건,

$$u = K_P e + K_I \int e dt + K_D \frac{de}{dt} \qquad (8.2)$$

The signal value, u is sent continuously to the plant with every corresponding new output, Y being obtained as the process continues.
신호 값 u는 해당되는 모든 새로운 출력이있는 플랜트로 연속적으로 전송되며, Y는 프로세스가 계속 진행될 때 얻어집니다.

The output, Y is sent back and subsequently new error signal, e is found and the same process repeats itself on and on.
출력 Y가 반송되고 새로운 오류 신호 e가 발견되고 동일한 프로세스가 계속 반복됩니다.

Also, it is very typical to have the PID transfer function written in several forms depending on the arrangement structure.
또한 배열 구조에 따라 PID 전달 함수를 여러 가지 형식으로 작성하는 것이 일반적입니다.

The following equation shows one of these (a parallel structure):
다음 방정식은 이들 중 하나 (병렬 구조)를 보여줍니다.

$$K_P + \frac{K_I}{s} + K_D \cdot s = K_P \times \left(1 + \frac{1}{T_I \cdot s} + T_D \cdot s\right) \qquad (8.3)$$

Where,

K_P = Proportional gain

T_I = Integral time or Reset time = $\frac{K_P}{K_I}$

T_D = Derivative time or Rate time

8.1 Some characteristics effects of PID controller parameters
PID 제어기 매개 변수의 몇 가지 특성 효과

The proportional gain □□ , will reduce the rise time and might reduce or remove the steady-state error of the system.
비례 이득 □□은 상승 시간을 줄여 시스템의 정상 상태 오류를 줄이거 나 제거 할 수 있습니다.

The integral gain □□, will eliminate the steady- state error but it might a negative effect on the transient response (a worse response might be produced in this case).
적분 게인 □□은 정상 상태 오류를 제거하지만 과도 응답에 부정적인 영향을 줄 수 있습니다 (이 경우에는 더 나쁜 응답이 생성 될 수 있음).

And the derivative gain □□ , will tend to increase the stability of the system, reducing overshoot percentage, and improving the transient response of the system.
그리고 미분 게인 □□은 시스템의 안정성을 증가시키고 오버 슛 비율을 줄이며 시스템의 과도 응답을 향상시키는 경향이 있습니다.

In all, the table below will give comprehensive effects of each of the controllers on a typical closed-loop system.
아래 표는 일반적인 폐쇄 루프 시스템에서 각 컨트롤러의 포괄적 인 효과를 제공합니다.

Parameter	Rise time	Overshoot	Settling time	Steady-state error
K_P	↓	↑	small change	↓
K_I	↓	↑	↑	eliminate
K_D	small change	↓	↓	small change

Legend	↓	*Decrease*
	↑	*Increase*

Table 8.1 – PID controller parameter characteristics on a typical system [8]

The ability to blend these three parameters will make a very efficient and stable system.
이 세 가지 매개 변수를 혼합하는 기능은 매우 효율적이고 안정적인 시스템을 만들 것입니다.

It should be noted that the relationship between the three controller parameters may not exactly be accurate because of their interdependency.
3 개의 제어기 매개 변수 사이의 상호 관계가 상호 의존성 때문에 정확히 정확하지 않을 수 있음을 알아야합니다.

Therefore, it is very possible to compute particular parameters which effects would be noticed on the other two.
따라서 특정 매개 변수를 계산하는 것이 가능하며 다른 매개 변수에 영향을 미칠 수 있습니다.

PID controller design tips
PID 컨트롤러 설계 팁

Designing a PID controller might require some of the following steps to obtain a more efficient and stable system [5]:
PID 제어기를 설계하려면보다 효율적이고 안정적인 시스템을 얻기 위해 다음 단계 중 일부가 필요할 수 있습니다 [5].

It is advisable to obtain the open-loop response of the system first and subsequently determine what to improve;
먼저 시스템의 개 루프 응답을 얻은 다음 개선 할 내용을 결정하는 것이 좋습니다.

Add a proportional gain control to improve the rising time;
상승 시간을 개선하기 위해 비례 게인 컨트롤을 추가하십시오.

Then, add a derivative gain to improve the overshoot percentage;
그런 다음 미분 게인을 추가하여 오버 슛 비율을 향상시킵니다.

And perhaps, add the integral control to eliminate the steady-state error;
그리고 정상 상태 오류를 제거하기 위해 적분 제어를 추가하십시오.

Thereafter, adjust each of the parameters might be important to achieve an overall desired performance (or output).
그 후, 각각의 매개 변수를 조정하여 전반적인 원하는 성능 (또는 출력)을 달성하는 것이 중요할 수 있습니다.

And most importantly, all the three PID controller parameters might not be necessarily used in some cases. In most cases, the tuning stops at the PI – control combination.
가장 중요한 것은 3 가지 PID 제어기 매개 변수가 모두 사용되지 않을 수도 있다는 것입니다. 대부분의 경우, PI 제어 조합에서 튜닝이 중지됩니다.

More also, it should be noted that the major goal of the PID parameters is to obtain a fast rise time with minimum overshoot and no (almost no) steady-state error.
또한, PID 파라미터의 주요 목표는 최소 오버 슛과 안정 상태 오류가없는 빠른 상승 시간을 얻는 것입니다.

9.PID CONTROLLER TUNING PARAMETERS
PID 제어기 튜닝 파라미터

Under this section a critical analysis would be done on the PID tuning criteria and the parameters involved.
이 섹션에서는 PID 튜닝 기준과 관련된 파라미터에 대한 중요한 분석을 수행합니다.

Before a detail analysis is done, a quick look at the tuning methods is considered first and thereafter, specific tuning parameters are computed for the BLDC maxon motor.
상세 분석을하기 전에 먼저 튜닝 방법을 간략하게 살펴본 후 BLDC maxon 모터에 대한 특정 튜닝 매개 변수를 계산합니다.

Some of the generally used tuning methods are the Trial and Error method, the Ziegler-Nichols method (1st), Improved Ziegler-Nichols method (2nd), Cohen-Coon method, Genetic Algorithms and so on.
일반적으로 사용되는 튜닝 방법에는 Trial and Error 방법, Ziegler-Nichols 방법 (1 차), Ziegler-Nichols 방법 개선 (2 차), Cohen-Coon 방법, Genetic 알고리즘 등이있다.

For this work, the Ziegler-Nichols tuning method would be given a priority.
이 작업을 위해서는 Ziegler-Nichols 튜닝 방법이 우선 순위가 부여됩니다.

9.1The PID arrangement
PID 배열

As a general form, a full schematic of the PID controller arrangement with the System model arrangement is displayed in figure 9.1 as a start for the tuning procedure.
일반적인 형태로서, 시스템 모델 배열을 가진 PID 컨트롤러 배열의 전체 도식이 그림 9.1에 튜닝 절차의 시작으로 표시됩니다.

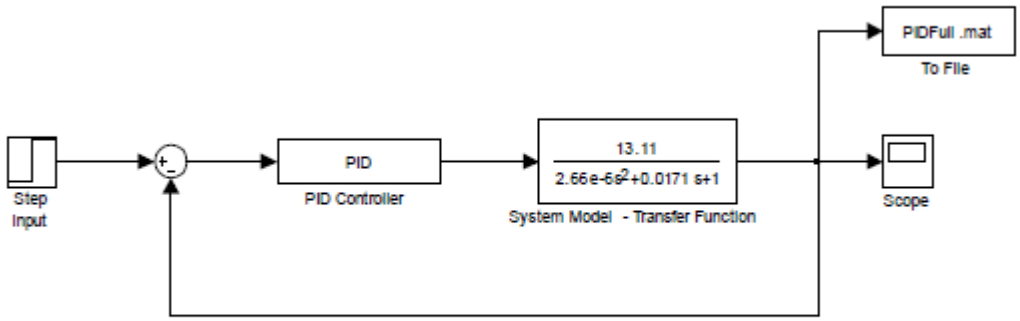

Figure 9.1 – PID Schematic for a full PID Controller with System model arrangement

The figure 9.1 is under no saturation, but the saturation is included in figure 9.2.
그림 9.1은 채도가 없지만 채도는 그림 9.2에 포함되어 있습니다.
Both figures would be used for our analysis.
두 수치 모두 우리의 분석에 사용됩니다.

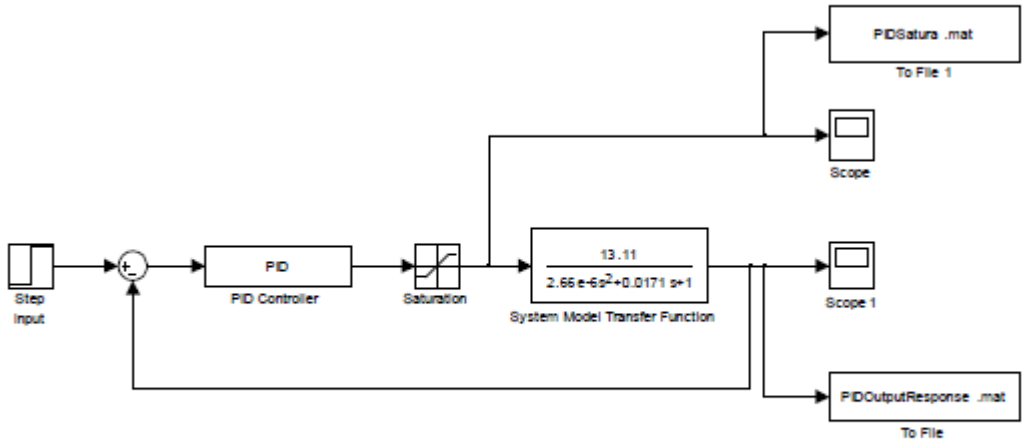

Figure 9.2 – PID Schematic for a full PID Controller (with saturation) and system model arrangement

For an initial computation, P, PI and PID would be considered in that order to observe the best part for the PID parameters to be obtained.
초기 계산을 위해 P, PI 및 PID가 순서대로 고려되어 PID 매개 변수에 대한 최상의 부분을 얻습니다.

9.2 Trial and Error tuning methods
시험 및 오류 조정 방법

This method is crude but could help in getting an overview of what the PID parameters could be like and their effects on the whole system model.
이 방법은 매우 어렵지만 PID 매개 변수가 무엇인지, 전체 시스템 모델에 미치는 영향에 대한 개요를 파악하는 데 도움이 될 수 있습니다.

It is particularly time consuming because of its trial and format.
그것의 예심 및 체재 때문에 특히 시간이 걸린다.

But a computational stability rule was needed to set a mark for the trial and effect.
그러나 시험 및 효과에 대한 표시를 설정하려면 계산 안정성 규칙이 필요했습니다.

This is done by using the Routh-Hurwitz stability rule as shown below. Under this, emphasis would be mainly on the PID combination.
이것은 아래와 같이 Routh-Hurwitz 안정성 규칙을 사용하여 수행됩니다. 이 점에서 강조점은 주로 PID 조합에 있습니다.

9.2.1 The Routh-Hurwitz stability rule
Routh-Hurwitz 안정성 규칙

From the various designs needed for this trial, a brief stability check is needed to make the trial and error at the first instance.
이 재판에 필요한 다양한 설계에서 첫 번째 인스턴스에서 시행 착오를하기 위해 간단한 안정성 검사가 필요합니다.

It would be observed that the only design near the perfect (open-loop – which is without compensation or controller) is the PID.
완벽 (개방 루프 - 보상 또는 컨트롤러가없는) 근처의 유일한 설계가 PID임을 알 수 있습니다.

To have a more appropriate trial and error value, the following steps would be followed for only the PID structure.
보다 적절한 시행 착오 값을 가지기 위해서는 PID 구조에 대해서만 다음 단계를 따라야합니다.

From the PID controller equation 9.1,
PID 제어기 방정식 9.1에서,

$$K_P + \frac{K_I}{s} + K_D \cdot s = K_P \times \left(1 + \frac{1}{T_I \cdot s} + T_D \cdot s\right) \qquad (9.1)$$

Similarly,

$$K_P + \frac{K_I}{s} + K_D \cdot s = \frac{K_P \cdot s + K_I + K_D.s^2}{s} \qquad (9.2)$$

This is used in the m-file tclosedloopPID_TrialError4.
이것은 m- 파일 tclosedloopPID_TrialError4에서 사용됩니다.

m and it is convuled with the motor model.
그것은 모터 모델로 해석됩니다.

Keeping the KP part, with TI and TD set to infinity and zero respectively.
KP 부분을 유지하면서 TI와 TD를 각각 무한대와 0으로 설정합니다.

A controller gain, KC could be obtained that would sustain the oscillation output.
컨트롤러 이득, KC는 발진 출력을 유지할 수 있습니다.

This value serves as the ultimate gain, KCU.
이 값은 궁극적 인 이득 인 KCU 역할을합니다.

For a proper oscillation, KC is set to be less than KCU.
적절한 발진을 위해 KC는 KCU보다 작게 설정됩니다.

Assumed the figure 9.9 below with a gain of KCU and the system model:
KCU와 시스템 모델의 이득으로 아래 그림 9.9를 가정합니다.

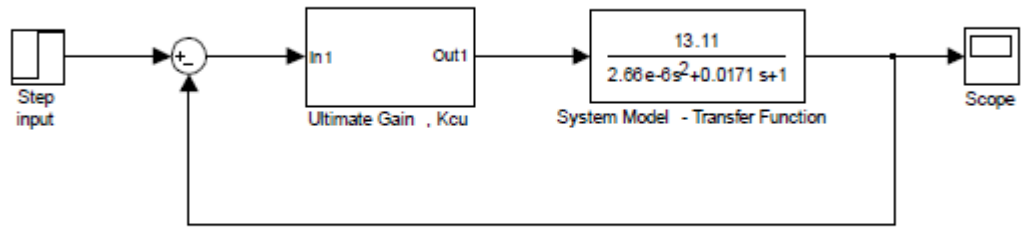

Figure 9.3 – Trial and Error PID computation diagram

By obtaining the characteristics equation of the figure 9.9, a limiting gain could be obtained just before sustained oscillation and this is assumed as the KCU.
그림 9.9의 특성 방정식을 얻음으로써 지속적인 진동이 일어나기 직전에 한계 이득을 얻을 수 있으며 이는 KCU로 가정됩니다.

tclosedloopPID_TrialError4.m

```
% Start of code
clear
close all
% includes constant parameters
constants
% includes evaluated constants
evaluatedconstants

num = [1/Ke];
den = [tm*te tm 1];

%Ziegler-Nichols parameter computed
Kp = 13.11;     %Proportional gain
Ki = 0%1310.6;          %Integral gain
Kd = 0%0.0763;          %Derivative gain
% For the PID equation
numc = [Kd Kp Ki];
denc = [1 0];

% convule "num with numc" and "den with demc"
numa = conv(num, numc);
dena = conv(den, denc);

% For the closed-loop transfer function, the following is obtained
% numac and denac used for the overall closed tranfer function in
this case
[numac, denac] = cloop(numa, dena);
% Plotting the new step-response
t = 0:0.00001:0.5;
step(numac, denac, t);      % across 0.01 seconds timing
title('Closed loop step response for ZN - Kp, Ki and Kd');
xlabel('Time, [s]')
ylabel('Voltage, [volts]')
%grid on;

% New G1 for overall closed loop trasnfer function
G1 = tf(numac, denac);
% End of code
```

Therefore, we have:

$$1 + K_{CU} \cdot G(s) = 0 \tag{9.3}$$

$$1 + K_{CU} \cdot \frac{13.11}{2.66 \times 10^{-6} \cdot s^2 + 0.0171 \cdot s + 1} = 0 \tag{9.4}$$

Equation 9.4 becomes,

식 9.4는,

$$2.66 \times 10^{-6} \cdot s^2 + 0.0171 \cdot s + 1 + 13.11 \cdot K_{CU} = 0 \qquad (9.5)$$

So for stability purposes, KCU's range of values could be obtained by using the Routh-Hurwitz condition of stability. This is computed below:
따라서 안정성을 위해 RUSA-HURWITS 안정성 조건을 사용하여 KCU의 값 범위를 얻을 수있었습니다. 이것은 다음과 같이 계산됩니다.

s^2	2.66×10^{-6}	$1 + 13.11 \cdot K_{CU}$
s^1	0.0171	0
s^0	$\mathbf{1 + 13.11 \cdot K_{CU}}$	−

According to Routh-Hurwitz condition, the obtained characteristics equation 9.5 should be spread into column as shown above and the s0 is evaluated as follows (because it has the assumed unknown KCU which would be evaluated):
Routh-Hurwitz 조건에 따라 얻은 특성 방정식 9.5를 위의 그림과 같이 열에 표시하고 s0는 다음과 같이 계산됩니다 (평가할 수있는 KCU라고 가정하기 때문에).

$$s^0(\text{1st row}) = -\frac{\begin{vmatrix} 2.66 \times 10^{-6} & 1 + 13.11 \cdot K_{CU} \\ 0.0171 & 0 \end{vmatrix}}{1 + 13.11 \cdot K_{CU}}$$

$$s^0(\text{1st row}) = -\frac{(2.66 \times 10^{-6} \times 0) - (1 + 13.11 \cdot K_{CU})(0.0171)}{0.0171} = 1 + 13.11 \cdot K_{CU}$$

For stability sake, the 1st column after the s-column must not have any sign change (that is, no change from + to – or – to +). Therefore, s0(1st row), must be greater than zero.
This implied that,
안정성을 위해 s- 열 뒤의 첫 번째 열에는 부호 변경이 없어야합니다 (즉, +에서 + 또는 -로 변경되지 않아야 함). 따라서 s0 (1 행)은 0보다 커야합니다.

$$1 + 13.11 \cdot K_{CU} > 0$$

$$13.11 \cdot K_{CU} > -1$$

$$K_{CU} > \frac{-1}{13.11} = -0.0763$$

이것은,
This implies that KCU has its main value in the positive range.
이는 KCU의 주요 가치가 양수 범위에 있음을 의미합니다.

With a rough trial and error tuning, KP, can be fixed to full value of the system model numerator, which is 13.11.
거친 시행 착오를 통해 KP는 시스템 모델 분자의 전체 값인 13.11로 고정 될 수 있습니다.

The KI and KD were set initially to zero to see the effect of the KP on the system.
KI 및 KD는 시스템에 대한 KP의 영향을보기 위해 초기에는 0으로 설정되었습니다.

This resulted into the figure KI about the inverse of 0.0763 = 13.106, and KD = 0.0763. After this,
이것은 0.0763 = 13.106 및 KD = 0.0763의 역수에 대한 수치 KI로 귀결된다. 금후,

9.2.2 Proportional control
비례 제어

Based on the M-file – “tclosedloopP.m”, the following figure 9.3, figure 9.4, figure 9.5 and figure 9.6 were obtained as an improvement to the open-loop system.
M- 파일 "tclosedloopP.m"을 기반으로 다음 그림 9.3, 그림 9.4, 그림 9.5 및 그림 9.6이 개방 루프 시스템에 대한 개선 사항으로 확보되었습니다.

By making an initial raw guess of the value of KP just before applying the Routh-Hurwitz condition.
Routh-Hurwitz 조건을 적용하기 직전에 KP의 가치에 대한 최초의 추측을합니다.

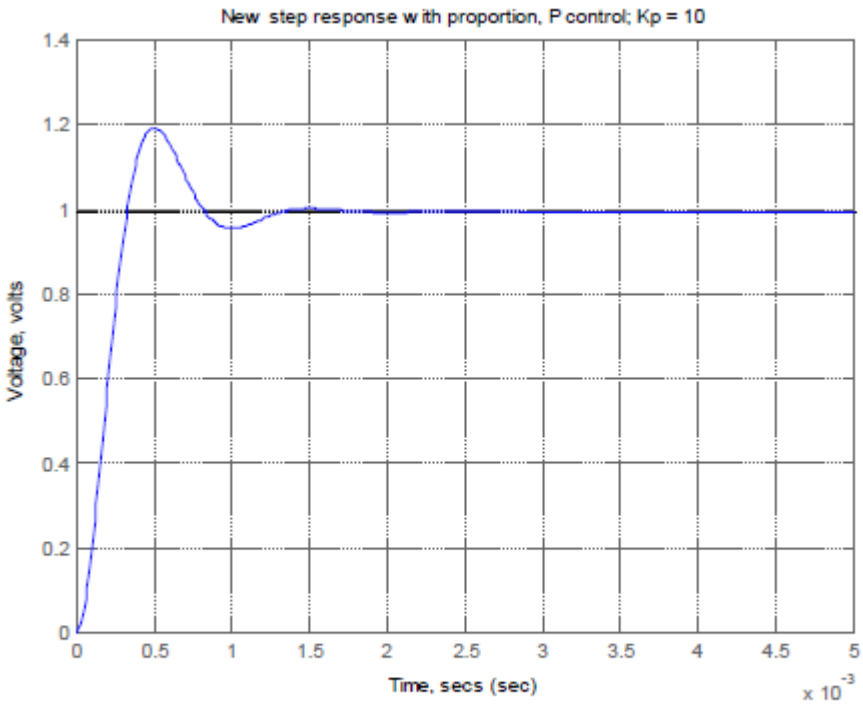

Figure 9.4 – Proportional controller gain effect on the system

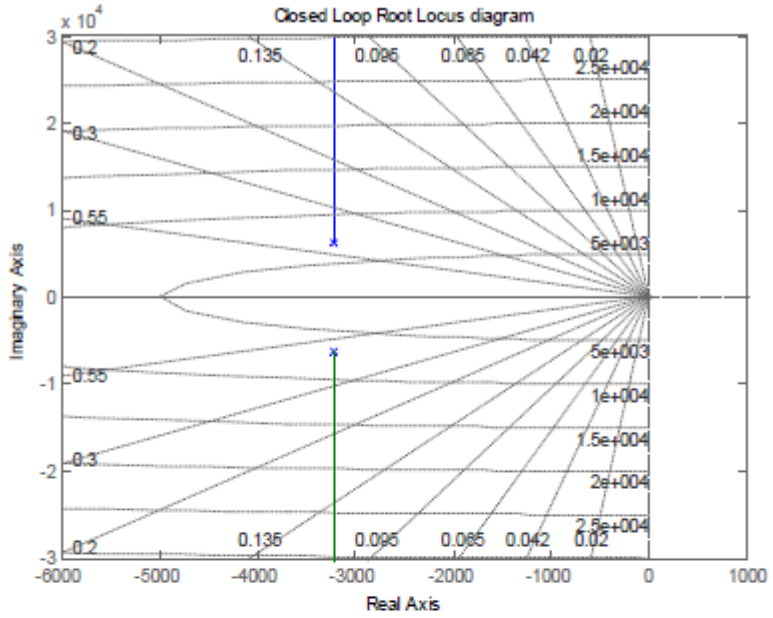

Figure 9.5 – Root locus diagram for the proportional controller gain effect

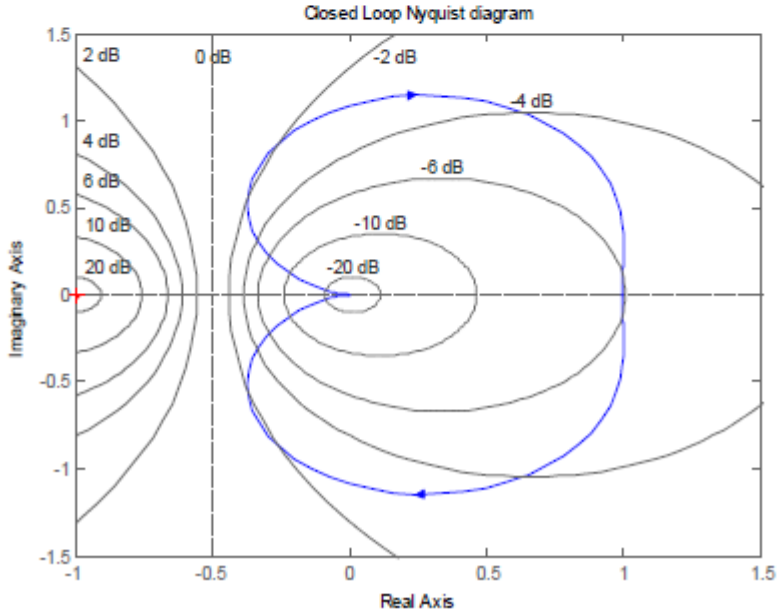

Figure 9.6 – Nyquist diagram for the proportional controller gain effect

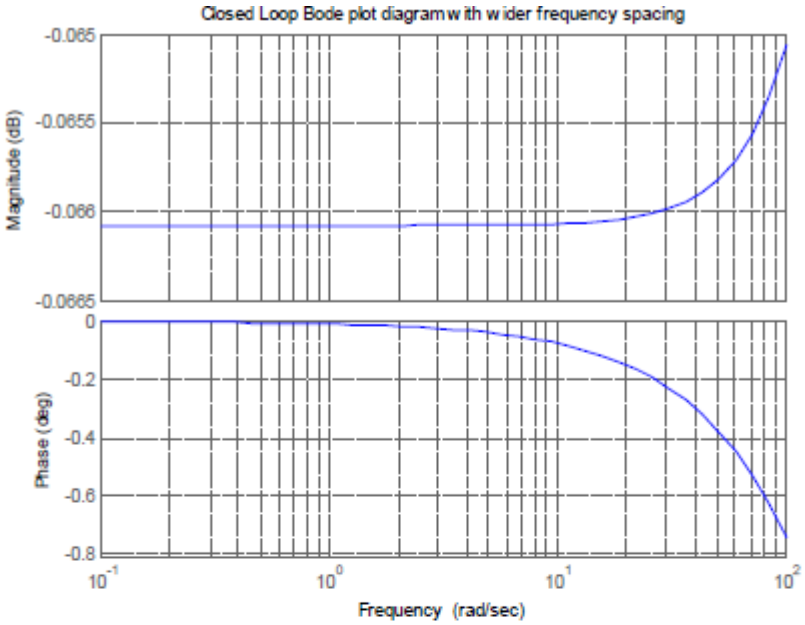

Figure 9.7 – Bode plot for the proportional controller gain effect

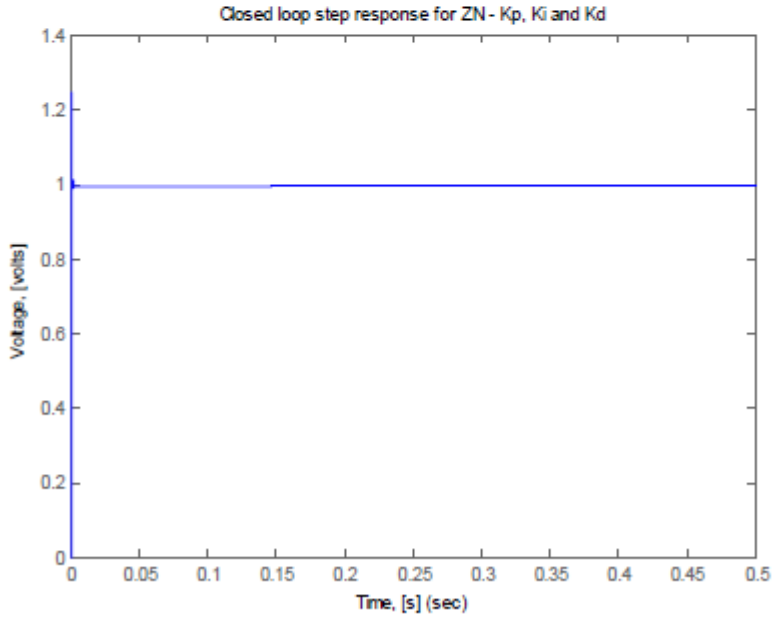

Figure 9.8 – Trial and error value used for the P parameters output, with K_I and K_D set to zero

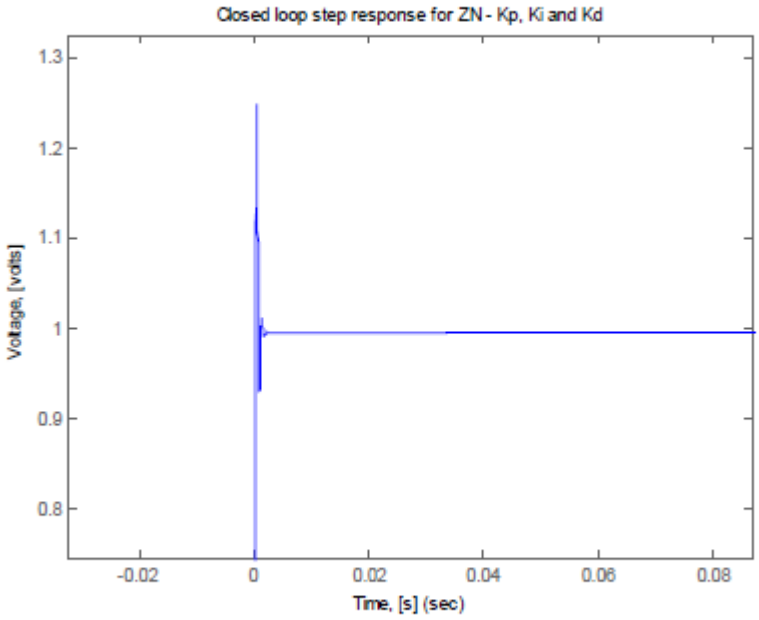

Figure 9.9 – Trial and error value used for the P parameters output, with K_I and K_D set to zero (zoomed display)

The above figure 9.3 – 9.7 show how the proportional controller has reduced the rising time and the steady-state error, the overshoot is reasonably increased but the settling time is also decreased slightly.
위 그림 9.3 - 9.7은 비례 제어기가 상승 시간과 정상 상태 오차를 어떻게 감소 시켰는지 보여주며, 오버 슛은 합리적으로 증가하지만 안정 시간도 약간 감소합니다.

The subsequent figures show the effects of the trial and error method of tuning applied. The detail analysis would be under the results and analysis section.
다음 그림은 시행 된 시행 착오 방법의 효과를 보여줍니다. 세부 분석은 결과 및 분석 섹션 아래에 있습니다.

9.2.3 Proportional-Integral control
비례 적분 적분 제어

To improve on effect of the KP, an additional KI was also set based on the Routh- Hurwitz condition used above.
KP의 효과를 향상시키기 위해 위에 사용 된 Routh-Hurwitz 조건을 기반으로 추가 KI가 설정되었습니다.

This is implemented with the same m-file– “tclosedloopPID_TrialError4.m”, the following figures 9.10 – 9.11 was obtained as an added improvement.
이것은 동일한 m- 파일 "tclosedloopPID_TrialError4.m"로 구현됩니다. 다음 그림 9.10 - 9.11이 추가 개선 사항으로 확보되었습니다.

To make a more visible on the step response, the integral parameter was scaled by 1000 to see its effects, that is, KI= 1310.6. And another “supposed” improvement was also obtained (figures 9.12 – 9.13).

계단 응답에서 더 눈에 띄기 위해 적분 매개 변수의 효과를 보려면 KI = 1310.6을 1000으로 조정했습니다. 그리고 또 다른 "가정"개선이 이루어졌다 (그림 9.12 - 9.13).

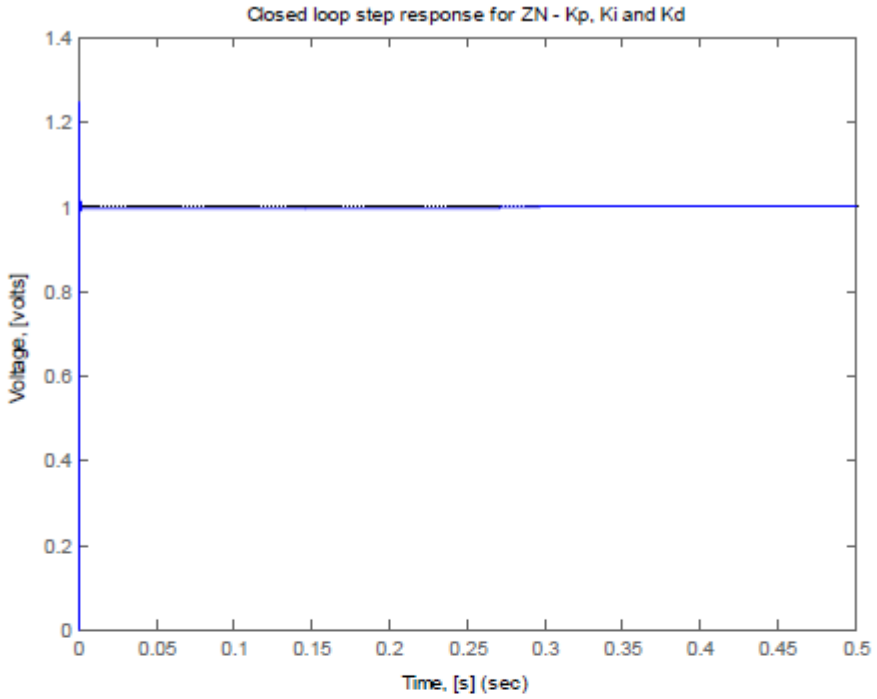

Figure 9.10 – Trial and error values used for the PI parameters output

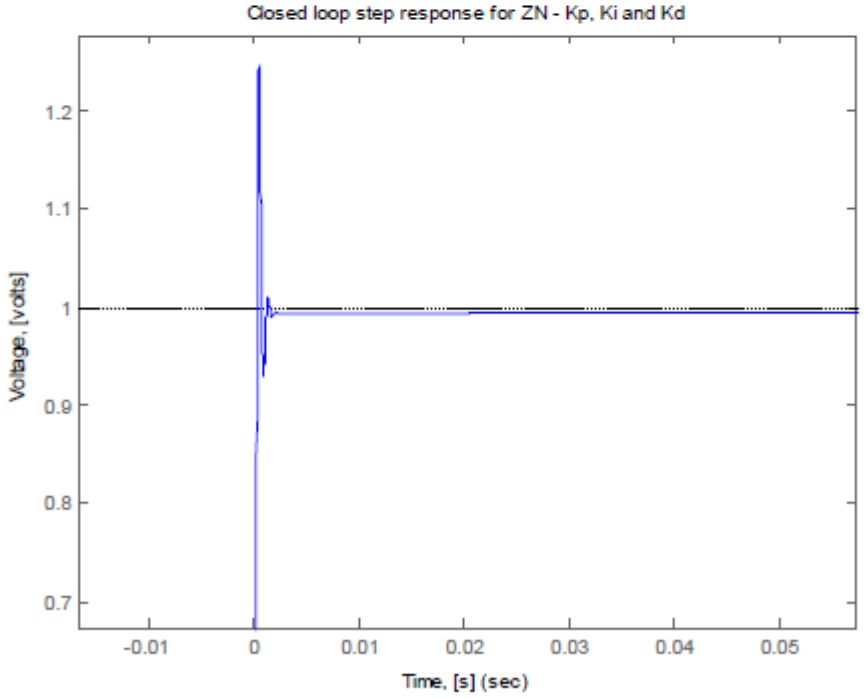

Figure 9.11 – Trial and error values used for the PI parameters output with Kd=0 (zoomed)

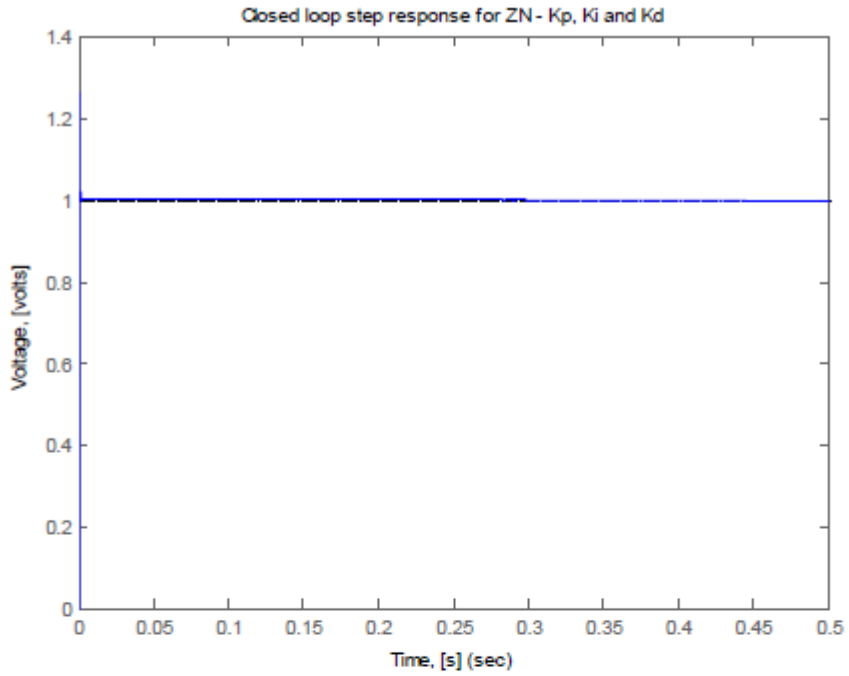

Figure 9.12 – Trial and error values used for the PI parameters output with Ki multiplied 1000 and Kd=0

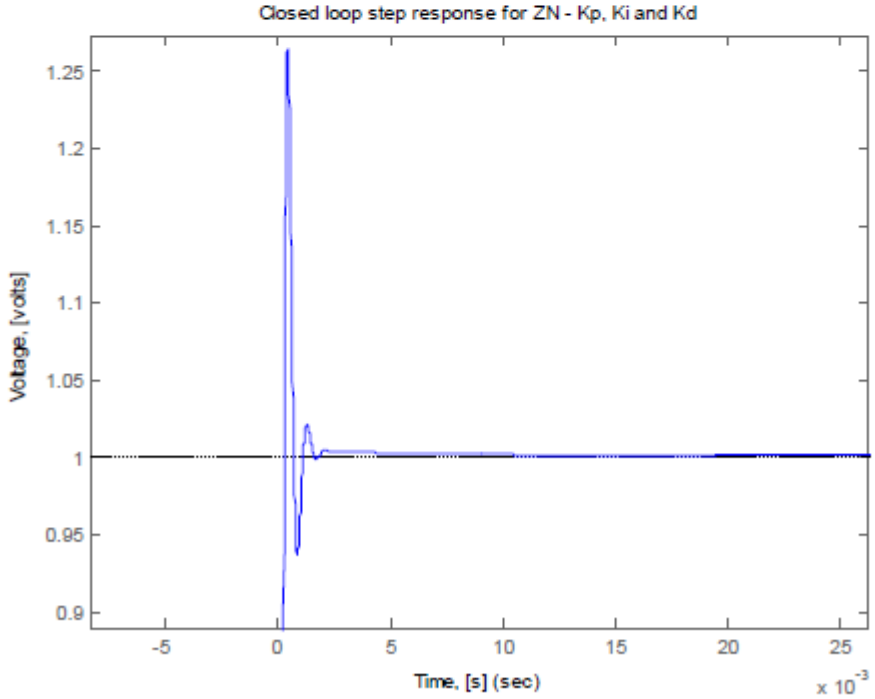

Figure 9.13 – Trial and error values used for the PI parameters output with Ki multiplied 1000 and Kd=0 (zoomed)

9.2.4 Proportional-Integral-Derivative control
비례 - 적분 - 미분 제어

But for a more critical assessment of the trial and error method, the M-file – “tclosedloopPID_TrialError4.m”, was used to obtain a more perfect output for the system response as shown in the following figure 9.8.
그러나 시행 착오적 인 방법에 대한보다 중요한 평가를 위해 M- 파일 "tclosedloopPID_TrialError4.m"을 사용하여 다음 그림 9.8에서와 같이 시스템 응답에 대한보다 완벽한 출력을 얻습니다.

Though, all the PID parameters might not be needed sometimes, but it needful to examine it to check the effect and the difference from the other P and PI combinations.
그러나 모든 PID 매개 변수는 때때로 필요하지 않을 수 있지만 다른 P 및 PI 조합과의 효과 및 차이를 확인하기 위해 검사해야합니다.

For the implementation of the PID guessed parameters based in the trial and error, the KI and KD were set to 1310.6 and 0.0763 respectively.
시행 착오에 근거한 PID 추측 파라미터의 구현을 위해, KI 및 KD는 각각 1310.6 및 0.0763으로 설정되었다.

On the first trial the figure – was obtained.
첫 번째 시험에서 그 그림이 얻어졌다.

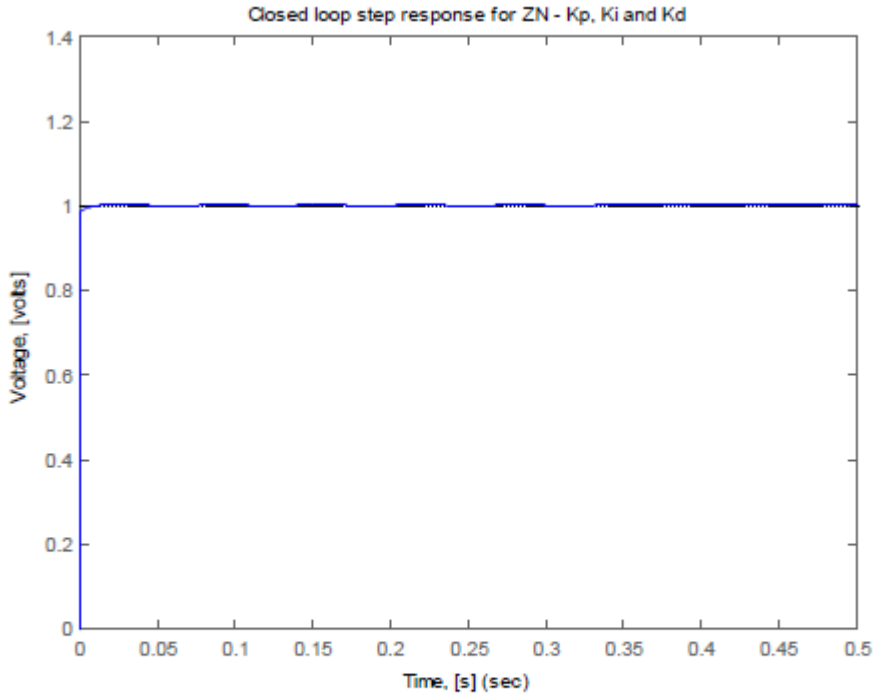

Figure 9.14 – Trial and error method for PID – control effect on the system response (first trial with Kd set at 0.0763)

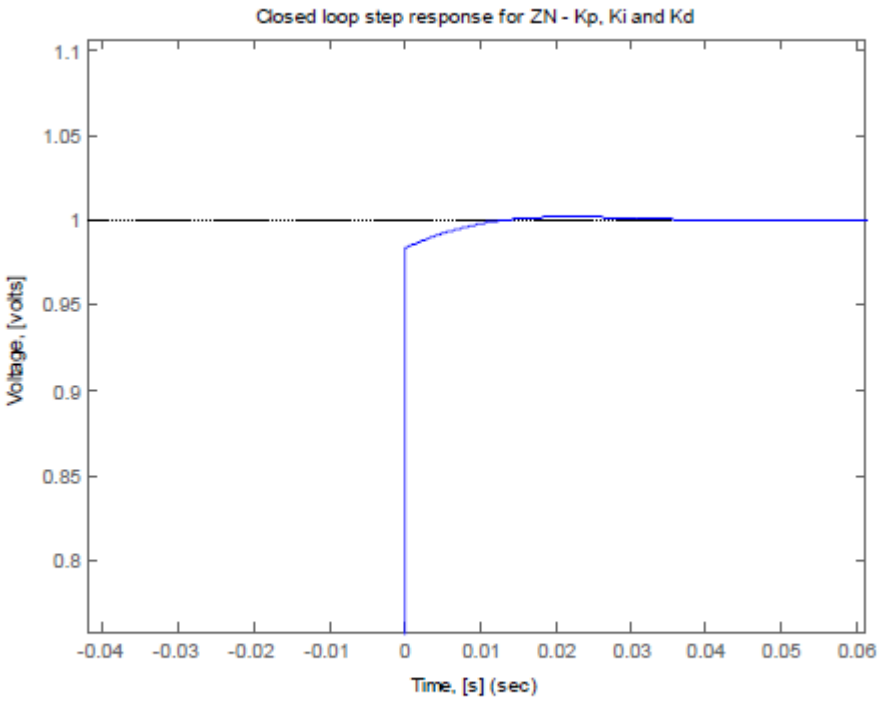

Figure 9.15 – Trial and error method for PID – control effect on the system response (first trial with Kd set at 0.0763, zoomed)

The trial and error gave a reasonable level comfort but it is time consuming and requires extra techniques to be able to have guesses that are appropriate and near efficient.
시행 착오는 합리적인 수준의 편안함을 주었지만, 시간이 많이 걸렸으며 적절하고 근사한 추측을 할 수있는 추가 기술이 필요했습니다.

For an overall assessment of the P, PI and PID parameters effect, the following figure was generated for appropriate comparison effects using the UpdatedPPIPID_TrialError.m.
P, PI 및 PID 매개 변수 효과의 전반적인 평가를 위해 다음 그림이 UpdatedPPID_TrialError.m을 사용하여 적절한 비교 효과에 대해 생성되었습니다.

	PID Type	K_P	K_I	K_D
1.	P	13.11	0	0
2.	PI	13.11	1310.6	0
3.	PID	13.11	1310.6	0.0763

Table 9.1 – Results of the Trial and Error method for PID controller parameters

UpdatedPPIPID_TrialError.m

```
% Start of code
clear
close all

% includes constant parameters
constants
% includes evaluated constants
evaluatedconstants
num = 1/Ke;
den = [tm*te tm 1];

%----P starts
% assumed Kp = 13.11
Kp1 = 13.11;
numa1 = Kp1 * num;
dena1 = den;

% For the closed-loop transfer function, the following is obtained
% numac and denac used for the overall closed tranfer function in
this case
[numac1, denac1] = cloop(numa1, dena1);
%----P ends

%----PI Starts
%Trial and Error tuning parameter Kp and Ki
Kp2 = 13.11;
Ki2 = 1310.6;

% For the PI equation
numc2 = [Kp2 Ki2];
denc2 = [1 0];

% convule "num with numc" and "den with demc"
numa2 = conv(num, numc2);
dena2 = conv(den, denc2);

% For the closed-loop transfer function, the following is obtained
% numac and denac used for the overall closed tranfer function in
this case
[numac2, denac2] = cloop(numa2, dena2);
%----PI ends

%----PID Starts
%Trial and Error parameter guessed with support of RH
Kp3 = 13.11;    %Proportional gain
Ki3 = 1310.6;   %Integral gain
Kd3 = 0.0763;    %Derivative gain
% For the PID equation
numc3 = [Kd3 Kp3 Ki3 ];
denc3 = [1 0];
```

UpdatedPPIPID_TrialError.m (contd.)

```
% convule "num with numc" and "den with demc"
numa3 = conv(num, numc3);
dena3 = conv(den, denc3);

% For the closed-loop transfer function, the following is obtained
% numac and denac used for the overall closed tranfer function in
this case
[numac3, denac3] = cloop(numa3, dena3);
%----PID ends

% Plotting the new step-response
t = 0:0.00001:0.01;

% New G1 for overall closed loop transfer function
G1 = tf(numac1, denac1);

G2 = tf(numac2, denac2);

G3 = tf(numac3, denac3);

% Plots the Step Response diagram
figure;
hold on
step(G1, t);
hold on
step(G2, t);
hold on
step(G3, t);
legend('P', 'PI', 'PID');
title('Closed Loop PID Trial and Error step response generated for
```

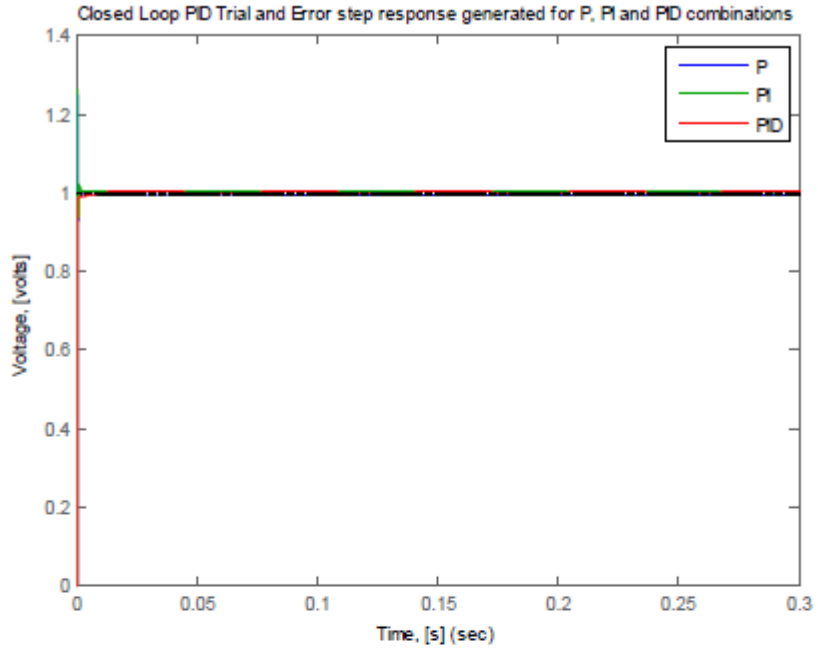

Figure 9.16 – Trial and error method for P, PI and PID – control effect on the system response (t-max=0.3s)

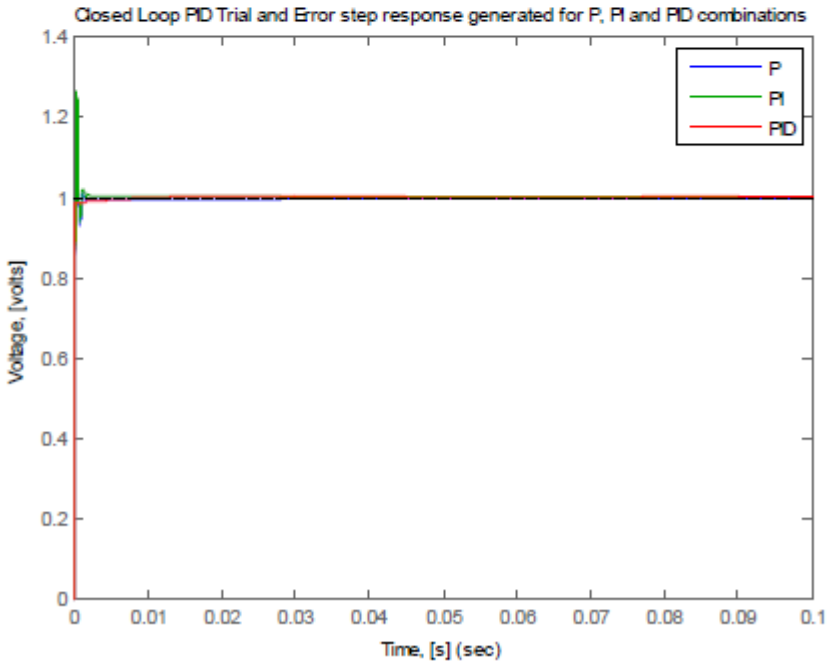

Figure 9.17 – Trial and error method for P, PI and PID – control effect on the system response (t-max=0.1s)

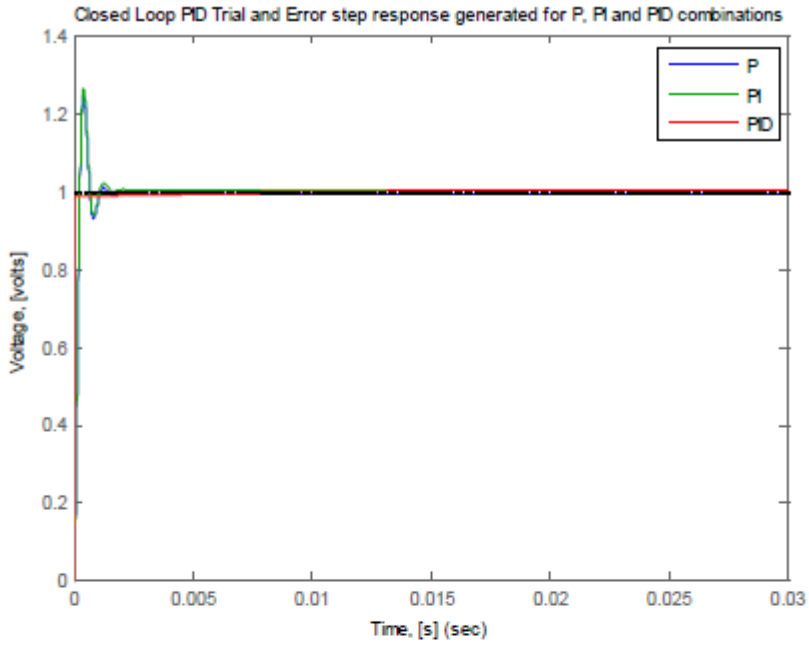

Figure 9.18 – Trial and error method for P, PI and PID – control effect on the system response (t-max=0.03s)

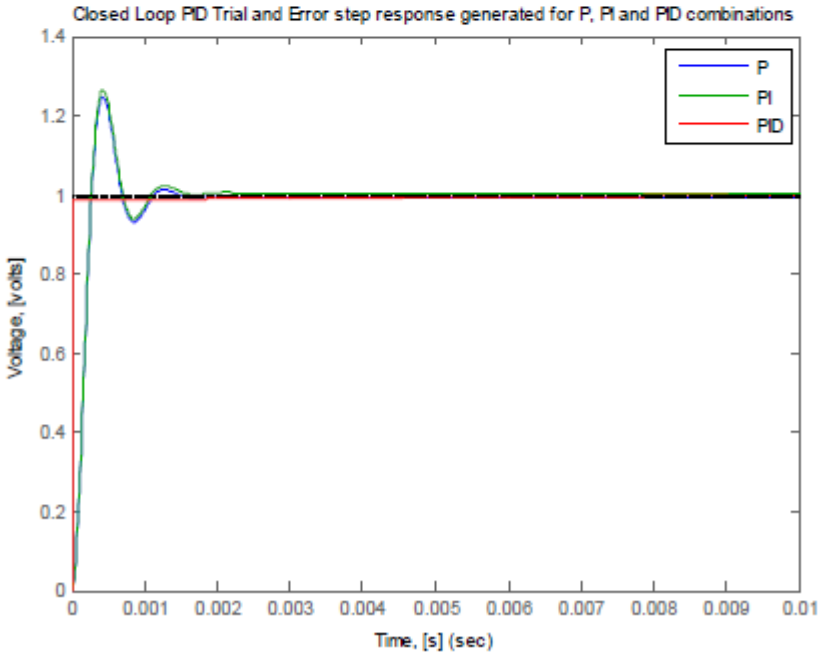

Figure 9.19 – Trial and error method for P, PI and PID – control effect on the system response (t-max=0.01s)

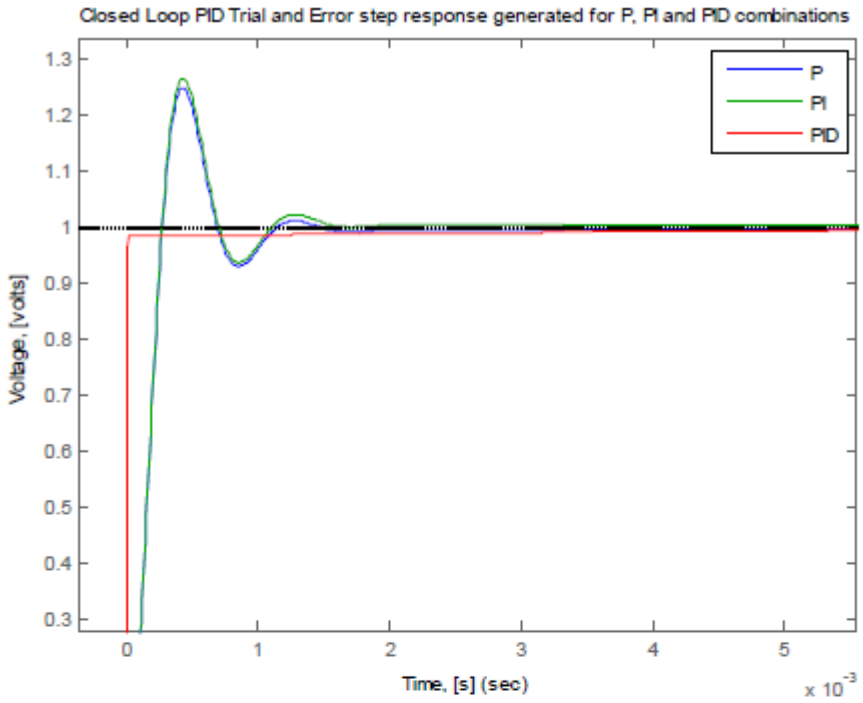

Figure 9.20 – Trial and error method for P, PI and PID – control effect on the system response (1st zooming)

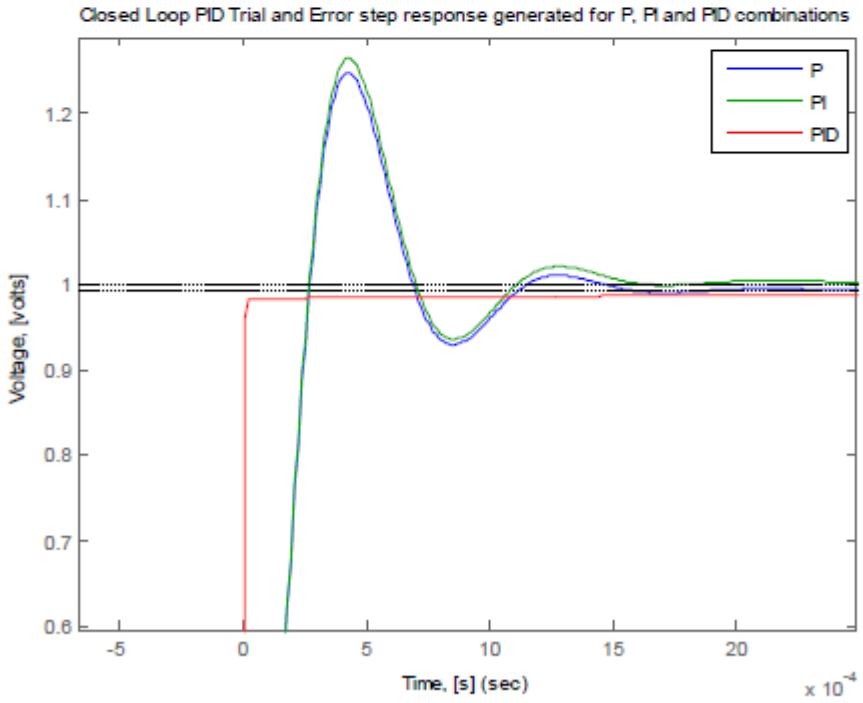

Figure 9.21 – Trial and error method for P, PI and PID – control effect on the system response (2nd zooming)

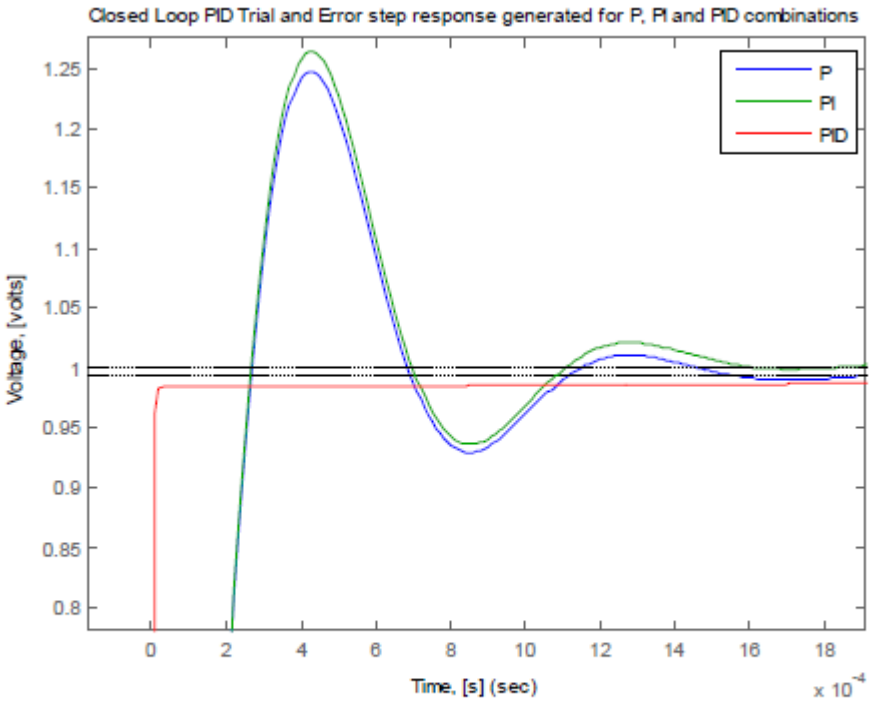

Figure 9.22 – Trial and error method for P, PI and PID – control effect on the system response (3rd zooming)

9.3 Ziegler-Nichols tuning methods
지글러-니콜스 튜닝방법

The Ziegler-Nichols method used was done based on obtaining the open loop transfer function and thereafter obtaining the necessary parameter values needed for the various evaluation of the P, PI and PID parameters.
사용 된 Ziegler-Nichols 방법은 개방 루프 전달 함수를 얻은 다음에 P, PI 및 PID 매개 변수의 다양한 평가에 필요한 필요한 매개 변수 값을 얻는 것에 기초하여 수행되었습니다.

The steps taken involve the files topenloop.m used in conjunction with the openloop.mdl model.
수행되는 단계에는 openloop.mdl 모델과 함께 사용되는 topenloop.m 파일이 포함됩니다.

So, for the Ziegler-Nichols method analysis the m-file topenloop_zn.m was used accordingly.
그래서 지글러 - 니콜스 방법 분석을 위해 m- 파일 인 topenloop_zn.m이 적절하게 사용되었습니다.

The open loop step response is characterized by two main parameters, the L (delay time parameter) and T (time constant).
개방 루프 스텝 응답은 L (지연 시간 파라미터)과 T (시간 상수)의 두 가지 주요 파라미터로 특징 지어집니다.

These two parameters are computed by drawing tangents to the open loop step response at its point of inflections (basically two points.
이 두 매개 변수는 굴곡 지점에서 열린 루프 스텝 응답에 대한 접선을 그려서 계산됩니다 (기본적으로 두 점).

The inflection points are particularly done so that there would be an intersection with the vertical (voltage axis, which correlates with the steady-state value) and horizontal (time axis) axes.
변곡점은 특히 수직 (전압 - 정상 상태 값과 상관 관계가있는 축)과 수평 (시간 축) 축의 교차점이있을 수 있도록 수행됩니다.

	PID Type	K_P	$T_I = \frac{K_P}{K_I}$	$T_D = \frac{K_D}{K_P}$
1.	P	$\frac{T}{L}$	∞	0
2.	PI	$0.9\times\frac{T}{L}$	$\frac{L}{0.3}$	0
3.	PID	$1.2\times\frac{T}{L}$	$2\times L$	$0.5\times L$

Table 9.2 – Ziegler-Nichols PID controller parameters model [10]

Based on the Ziegler-Nichols, the following were derived to obtain the control parameters based on the required model:
지글러 - 니콜스 (Ziegler-Nichols)에 기초하여, 요구되는 모델에 기초한 제어 파라미터를 얻기 위해 다음을 유도 하였다 :

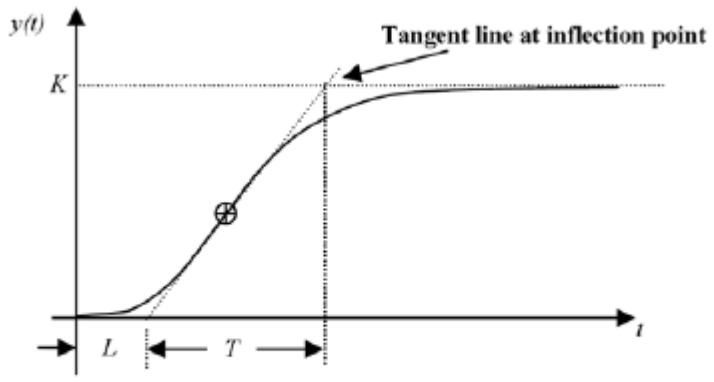

Figure 9.23 – Ziegler-Nichols step response tuning method [10]

From the figure 9.23, the target is on how to evaluate the two parameters (L and T) needed. This is done as follows with the illustration.
그림 9.23에서, 목표는 필요한 두 매개 변수 (L과 T)를 평가하는 방법입니다. 이것은 그림과 같이 다음과 같이 수행됩니다.

topenloop_zn.m

```
%
% Start of code
%
% includes constant parameters
clear
close all

%motor constants
constants

% includes evaluated constants
evaluatedconstants

% Transfer function
G = tf([1/Ke],[tm*te tm 1]);

% Plots the Step Response diagram
figure;
step(G, 0.5);
title('Open Loop Step Response diagram');
xlabel('Time, secs')
ylabel('Voltage, volts')
%grid on;

format long
load openloop.mat
coeff_x=polyfit([6 10 12],openloop(2,[6 10 12]),1)
coeff_y=polyfit([700:900],openloop(2,[700:900]),1)

for n=1:100
    zn_line_x(n)=coeff_x(1)*n+coeff_x(2);
end

for n=1:900
    zn_line_y(n)=coeff_y(1)*n+coeff_y(2);
end

figure(2)
hold on
plot(openloop(2,:),'red')
plot(zn_line_x);
plot((zn_line_y), 'green');
legend('1step response','line');
grid on
axis([0 400 0 14]);
l=length(openloop(2,:))
L_samples=roots(coeff_x)

%inflecton_point=intersect(zn_line_x,zn_line_y)
[a,b,c]=intersect(zn_line_x,zn_line_y)

% End of code
```

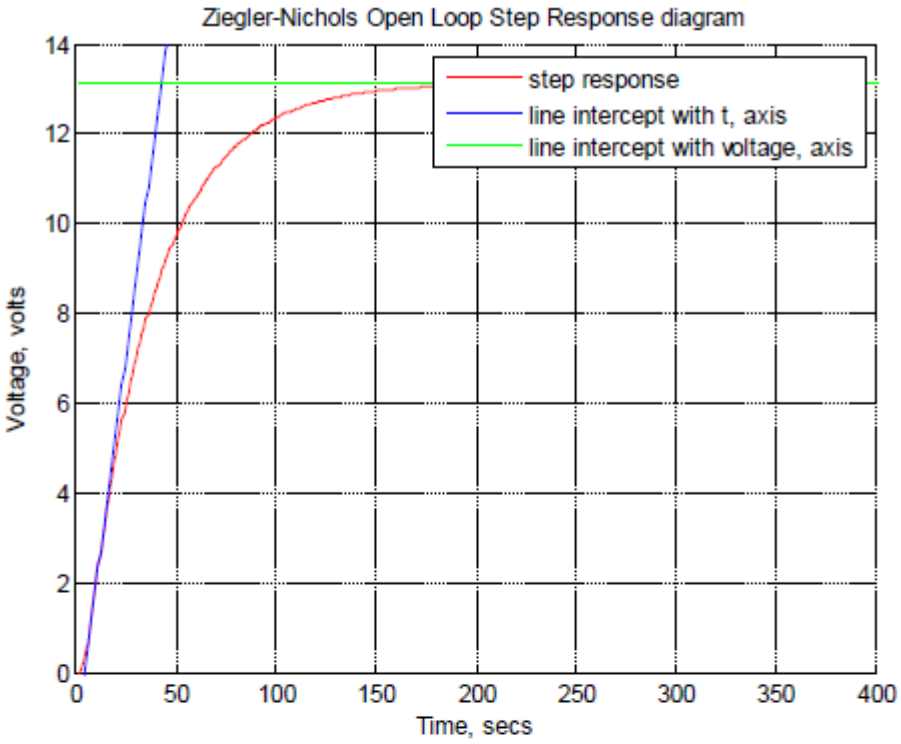

Figure 9.24 – Ziegler-Nichols open step response plot computation

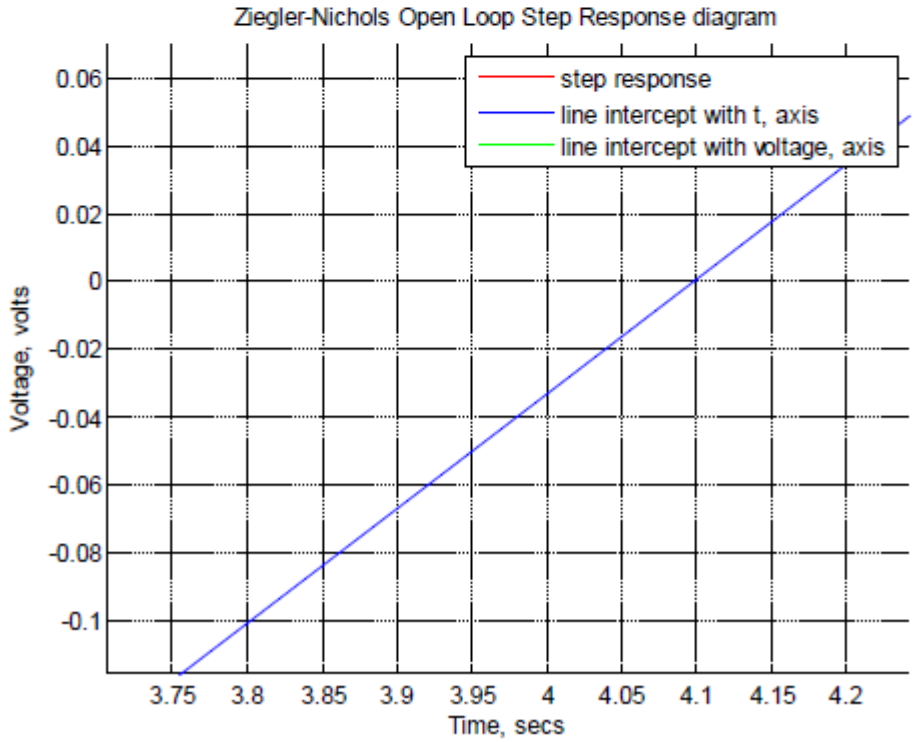

Figure 9.25 – Ziegler-Nichols open step response horizontally zoomed

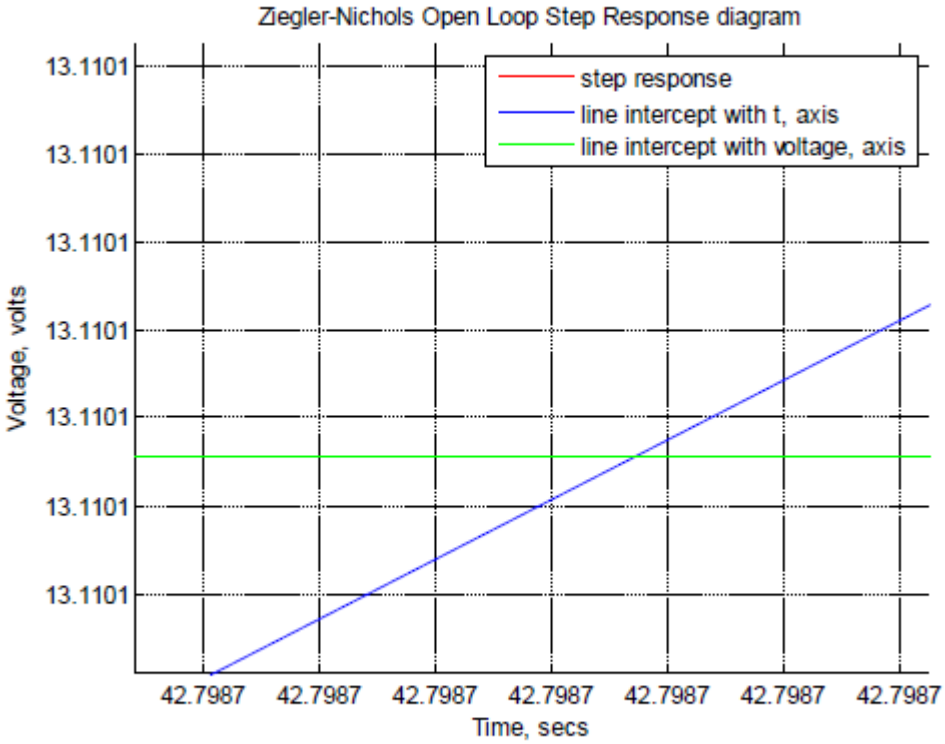

Figure 9.26 – Ziegler-Nichols open step response vertically zoomed

Therefore, from the figure 9.24, figure 9.25 and figure 9.26, the values of the L and T could be computed as follows:
그러므로 그림 9.24, 그림 9.25 및 그림 9.26에서 L과 T의 값은 다음과 같이 계산할 수 있습니다.

An assumed sample rate of 1000 was used for the topenloop_zn.m plots Point of interception of the horizontal line ≈ 4.1 (voltage = 0)
Coordinate of the point of interception of the two lines ≈ (T*, K) = (42.7987, 13.1101);
가정 된 샘플 속도 1000은 topenloop_zn.m 도표에 사용되었습니다. 수평선의 가로 채기 지점 ≈ 4.1 (전압 = 0) 두 선의 가로 채점의 좌표 ≈ (T *, K) = (42.7987, 13.1101);

Where,
T* is horizontal trace of the interception on the tangent lines drawn L = 4.1;
어디에,
T *는 그려진 접선에서 가로 채기의 흔적입니다. L = 4.1;

K = 13.1101;
T = T* – L = 4.1 = 42.7987 - 38.6987 ≈ 38.70
This implies that we have: L = 0.0041;
K = 13.1101;
T = 0.0387

With the above computation, the P, PI and PID computation was done to get the best suited parameters combination desired.
위의 계산을 사용하여 P, PI 및 PID 계산을 수행하여 원하는 최적의 매개 변수 조합을 얻습니다.

So the updated table 9.1 would be table 9.2 shown below:
따라서 업데이트 된 표 9.1은 아래 표 9.2와 같습니다.

-

	PID Type	K_P	$T_I = \frac{K_P}{K_I}$	$T_D = \frac{K_D}{K_P}$
1.	P	9.439	∞	0
2.	PI	8.495	0.0137	0
3.	PID	11.327	0.0082	0.00205

Table 9.3 – Results of the Ziegler-Nichols method for PID controller parameters

From table 9.2, the following parameters are obtained based on the equation format (from equation 7.3 above) to become equation 9.1 below:
표 9.2에서 아래의 방정식 9.1이되도록 방정식 형식 (위의 방정식 7.3)을 기반으로 다음 매개 변수가 얻어집니다.

For P only,

$$K_P + \frac{K_I}{s} + K_D \cdot s = 9.439 + \frac{K_I}{s} + K_D \cdot s \qquad (9.1)$$

For PI only,

$$K_P + \frac{K_I}{s} + K_D \cdot s = 8.495 + \frac{620.07}{s} + K_D \cdot s \qquad (9.2)$$

For PID only,

$$K_P + \frac{K_I}{s} + K_D \cdot s = 11.327 + \frac{1381.34}{s} + 0.0232 \cdot s \qquad (9.3)$$

Using the figure 9.1 (above) and m-file tclosedloopP_zn.m, tclosedloopPI_zn.m and tclosedloopPID_zn.m, the outputs of the various PID combinations could be obtained as given below:
그림 9.1 (위)과 m 파일 tclosedloopP_zn.m, tclosedloopPI_zn.m 및 tclosedloopPID_zn.m을 사용하여 다양한 PID 조합의 출력을 다음과 같이 얻을 수 있습니다.

tclosedloopP_zn.m

```
%start of code
clear
close all

% includes constant parameters
constants
% includes evaluated constants
evaluatedconstants

num = 1/Ke;
den = [tm*te tm 1];

% assumed Kp = 10
Kp = 9.439;
numa = Kp * num;
dena = den;

% For the closed-loop transfer function, the following is obtained
% numac and denac used for the overall closed transfer function in
this case
[numac, denac] = cloop(numa, dena);

% Plotting the new step-response
t = 0:0.00001:0.005
step(numac, denac, t);      % across 0.01 seconds timing
title('Closed step response with proportion, P control; Kp =
9.439');
xlabel('Time, [s]')
ylabel('Voltage, [volts]')
grid on;

% New G1 for overall closed loop trasnfer function
G1 = tf(numac, denac);

% plots the Root-locus
figure;
rlocus(G1);
title('Closed Loop Root Locus diagram');
grid on;

% plots the Nyquist diagram
figure;
nyquist(G1);
title('Closed Loop Nyquist diagram');
grid on;

% plots the Bode Plot
figure;
bode(G1,{0.1 , 100})
title('Closed Loop Bode plot diagram with wider frequency
spacing');
grid on;

%end of code
```

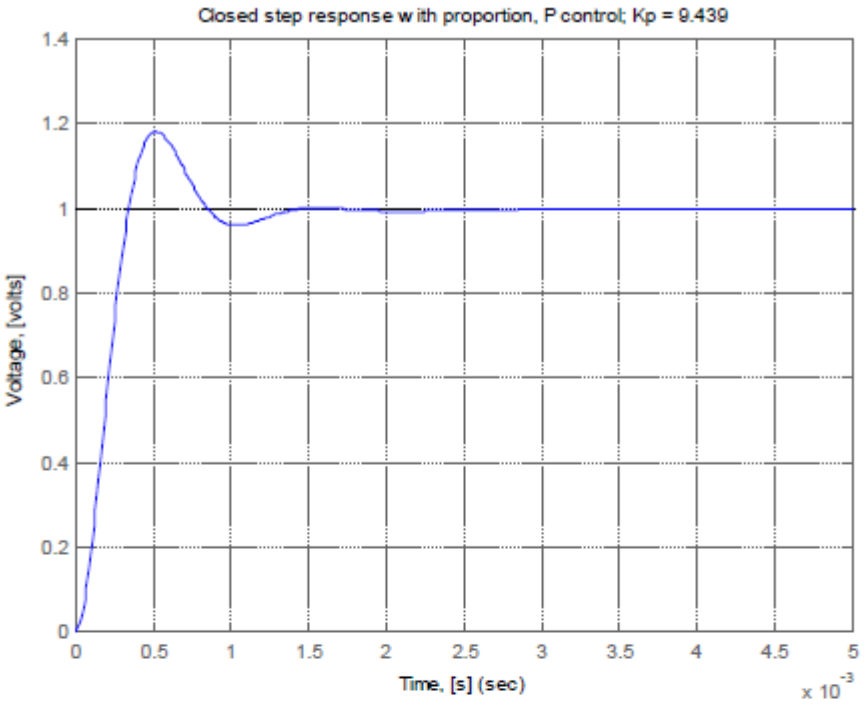

Figure 9.27 – P output for the Ziegler-Nichols tuning method

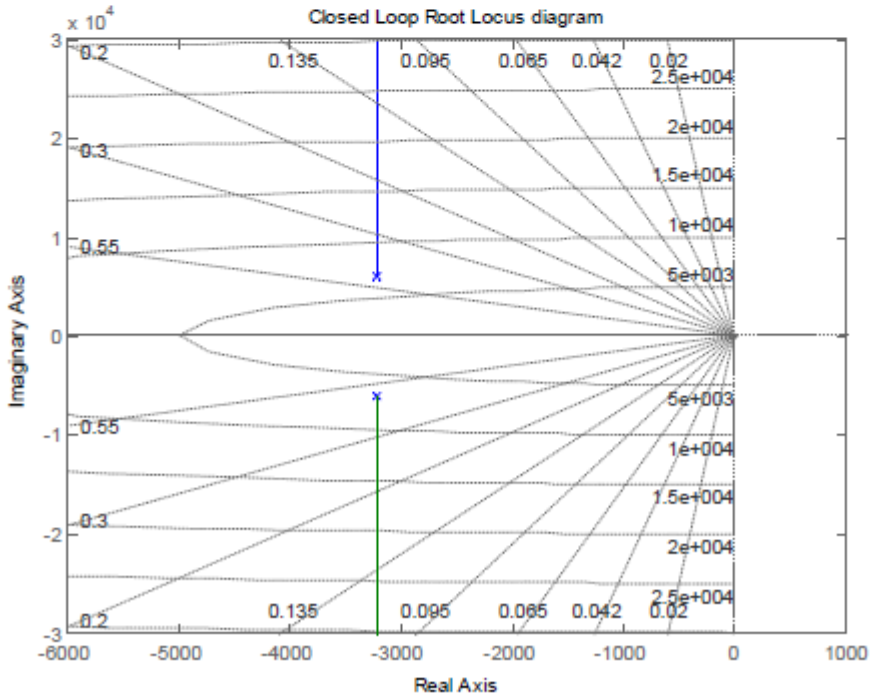

Figure 9.28 – P output for the Ziegler-Nichols tuning method root locus output

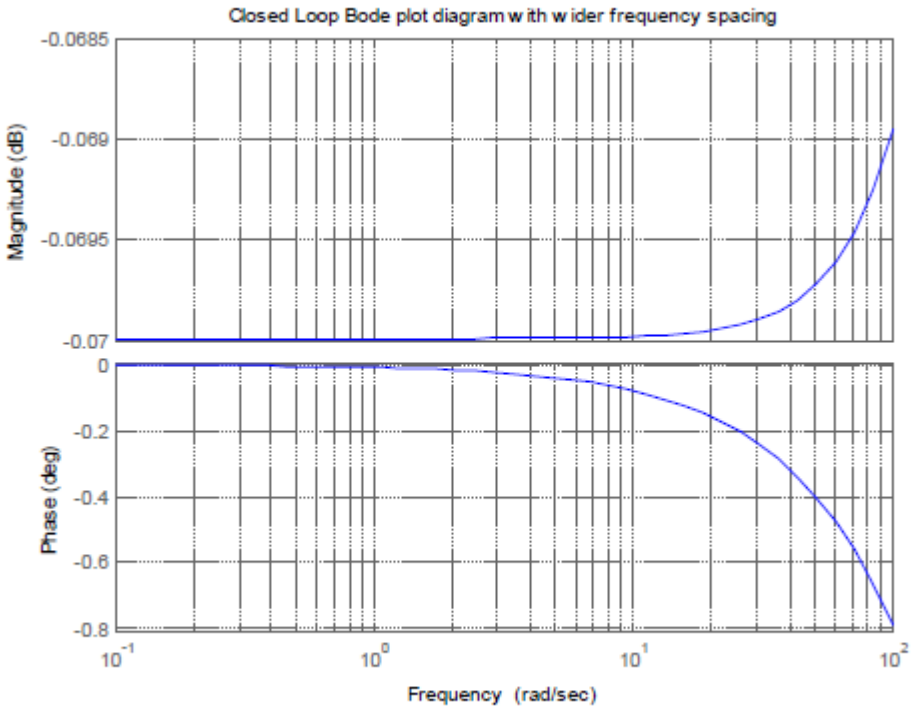

Figure 9.29 – P output for the Ziegler-Nichols tuning method Bode plot output

tclosedloopPI_zn.m

```
% Start of code
clear
close all
% includes constant parameters
constants
% includes evaluated constants
evaluatedconstants
num = 1/Ke;
den = [tm*te tm 1];

%Ziegler-Nichol tuning parameter Kp and Ki
Kp = 8.495;
Ki = 620.07;

% For the PI equation
numc = [Kp Ki];
denc = [1 0];
% convule "num with numc" and "den with demc"
numa = conv(num, numc);
dena = conv(den, denc);

% For the closed-loop transfer function, the following is obtained
% numac and denac used for the overall closed tranfer function in
this case
[numac, denac] = cloop(numa, dena);

% Plotting the new step-response
t = 0:0.00001:0.005
step(numac, denac, t);      % across 0.01 seconds timing
title('Closed step response with proportion, P control; Kp = 8.495
and Ki = 620.07');
xlabel('Time, [s]')
ylabel('Voltage, [volts]')
grid on;

% New G1 for overall closed loop trasnfer function
G1 = tf(numac, denac);
% plots the Root-locus
figure;
rlocus(G1);
title('Closed Loop Root Locus diagram');
grid on;
% plots the Nyquist diagram
figure;
nyquist(G1);
title('Closed Loop Nyquist diagram');
grid on;
% plots the Bode Plot
figure;
bode(G1,{0.1 , 100})
title('Closed Loop Bode plot diagram with wider frequency
spacing');
grid on;
%end of code
```

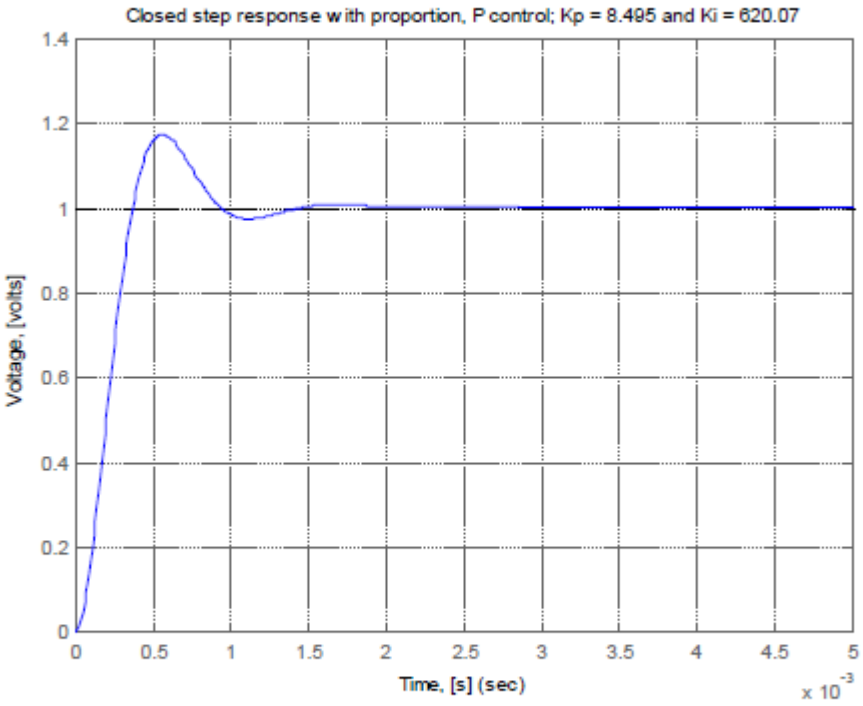

Figure 9.30 – PI output for the Ziegler-Nichols tuning method

tclosedloopPID_zn.m

```
% Start of code
% includes constant parameters
constants
% includes evaluated constants
evaluatedconstants
num = 1/Ke;
den = [tm*te tm 1];

%Ziegler-Nichols parameter computed
Kp = 11.327;    %Proportional gain
Ki = 1381.34;   %Integral gain
Kd = 0.0232;    %Derivative gain

% For the PID equation
numc = [Kd Kp Ki ];
denc = [1 0];

% convule "num with numc" and "den with demc"
numa = conv(num, numc);
dena = conv(den, denc);

% For the closed-loop transfer function, the following is obtained
% numac and denac used for the overall closed tranfer function in
this case
[numac, denac] = cloop(numa, dena);

% Plotting the new step-response
t = 0:0.00001:0.3;
step(numac, denac, t);      % across 0.01 seconds timing
title('Closed loop step response for ZN - Kp, Ki and Kd');
xlabel('Time, [s]')
ylabel('Voltage, [volts]')
%grid on;

% New G1 for overall closed loop trasnfer function
G1 = tf(numac, denac);

% plots the Root-locus
figure;
rlocus(G1);
title('Closed Loop Root Locus diagram');
grid on;
% plots the Nyquist diagram
figure;
nyquist(G1);
title('Closed Loop Nyquist diagram');
grid on;
% plots the Bode Plot
figure;
bode(G1,{0.1 , 100})
title('Closed Loop Bode plot diagram with wider frequency
spacing');
grid on;
%% End of code
```

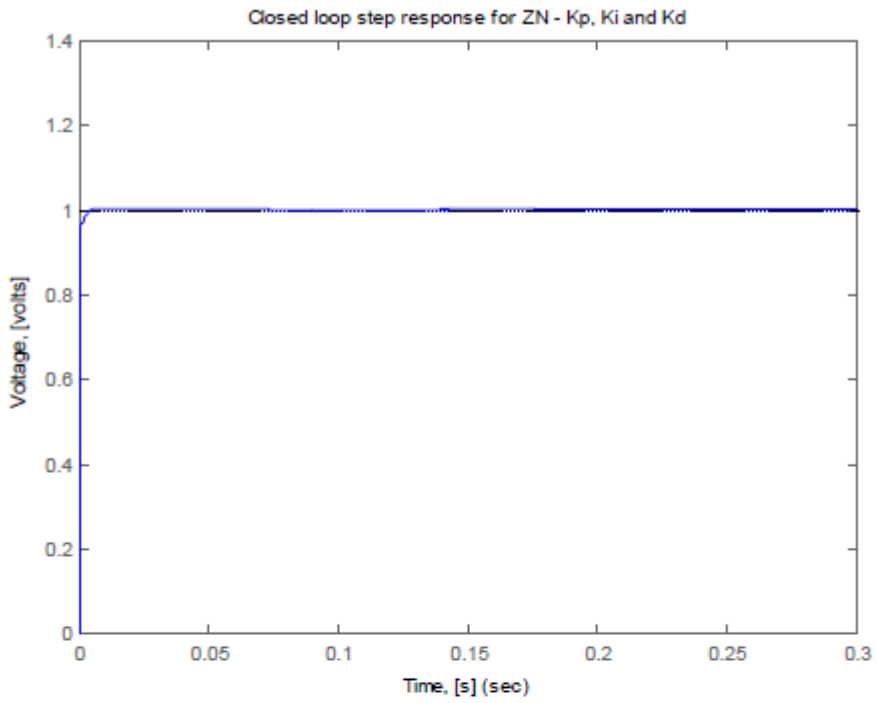

Figure 9.31 – Auto-scaled PID output for the Ziegler-Nichols tuning method

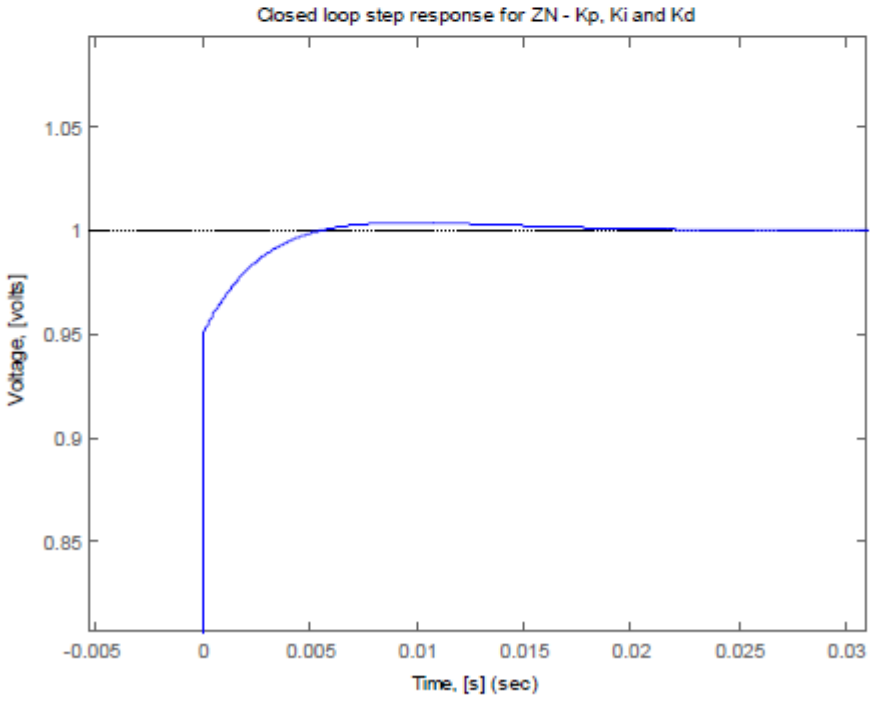

Figure 9.32 – Auto-scaled PID output for the Ziegler-Nichols tuning method (zoomed overshoot point)

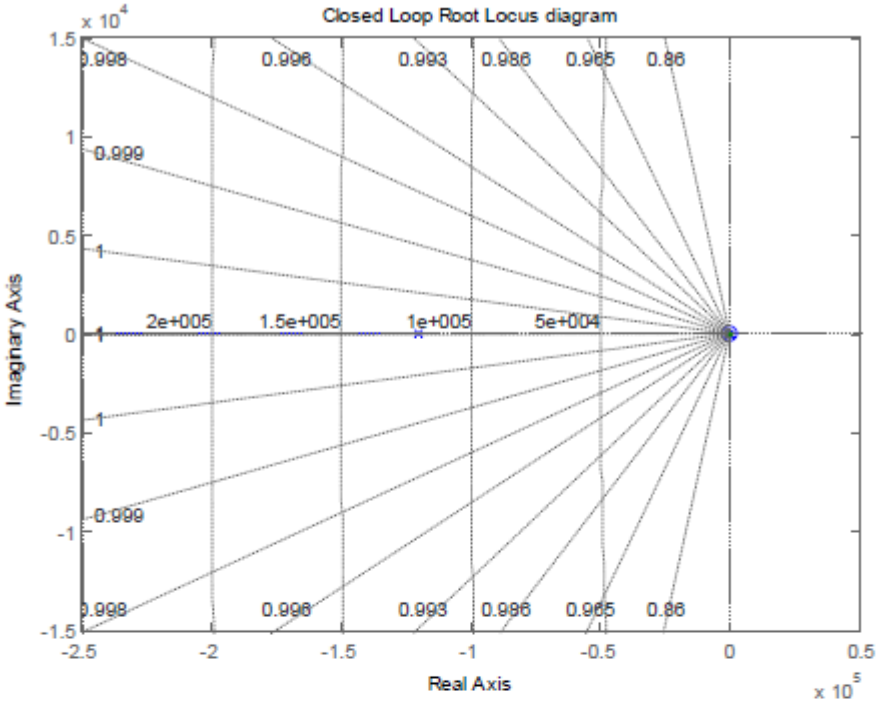

Figure 9.33 – PID Ziegler-Nichols tuning method Root locus diagram

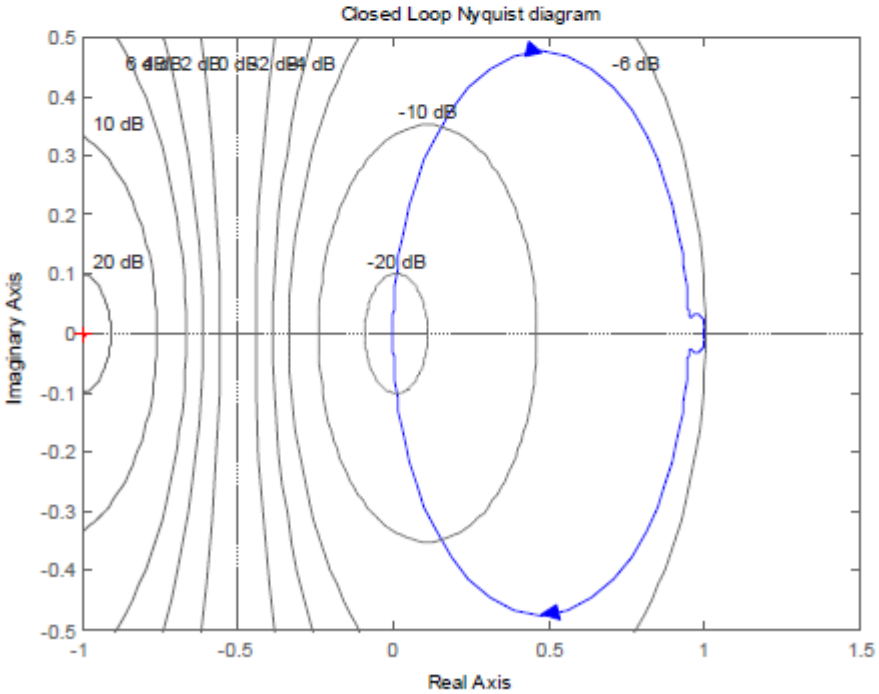

Figure 9.34 – PID Ziegler-Nichols tuning method Nyquist diagram

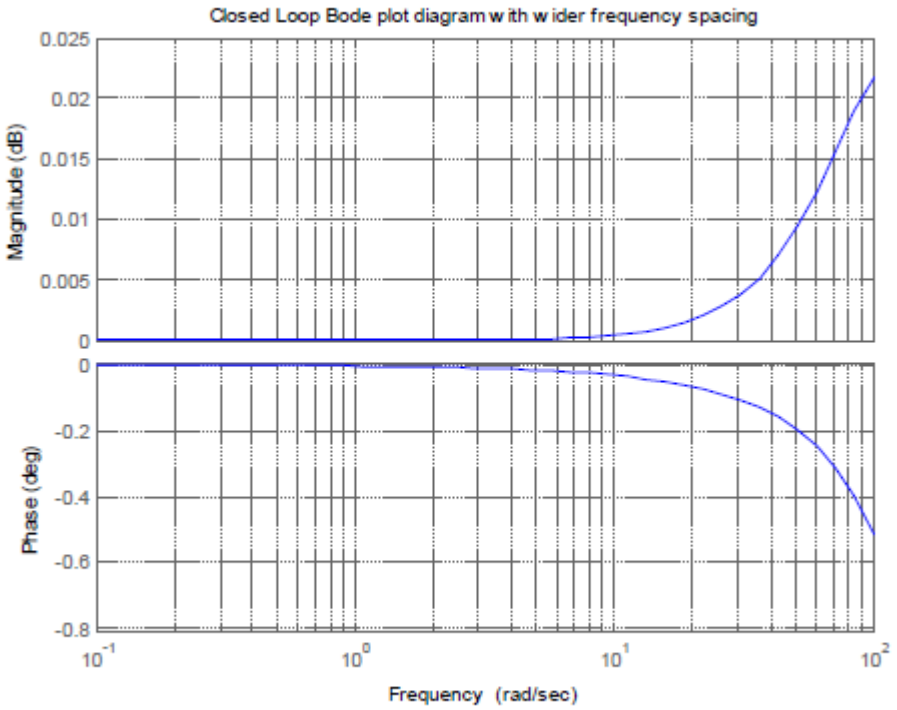

Figure 9.35 – PID Ziegler-Nichols tuning method Bode plot diagram

For a combined comparison of the Ziegler-Nichols tuning methods for the P, PI and PID, a separate m-file, UpdatedPPIPID_znj.m was created to execute the combination and this was done over different time spans (0.01, 0.03, 0.1 and 0.3).
P, PI 및 PID에 대한 Ziegler-Nichols 튜닝 방법의 결합 된 비교를 위해 별도의 m- 파일 인 UpdatedPPID_znj.m이 조합을 실행하기 위해 만들어졌으며 이것은 다른 시간 범위 (0.01, 0.03, 0.1 및 0.3).

The various outputs figures are shown in figures 9.23, 9.24, 9.25 and 9.26.
다양한 출력 수치는 그림 9.23, 9.24, 9.25 및 9.26에 나와 있습니다.

UpdatedPPIPID_znj.m

```
%
% Start of code
%
clear
close all

% includes constant parameters
constants
% includes evaluated constants
evaluatedconstants
num = 1/Ke;
den = [tm*te tm 1];


%----P starts
% assumed Kp = 10
Kp1 = 9.439;
numa1 = Kp1 * num;
dena1 = den;

% For the closed-loop transfer function, the following is obtained
% numac and denac used for the overall closed tranfer function in
this case
[numac1, denac1] = cloop(numa1, dena1);
%----P ends

%----PI Starts
%Ziegler-Nichols parameter computed
%Ziegler-Nichol tuning parameter Kp and Ki
Kp2 = 8.495;
Ki2 = 620.07;

% For the PI equation
numc2 = [Kp2 Ki2];
denc2 = [1 0];

% convule "num with numc" and "den with demc"
numa2 = conv(num, numc2);
dena2 = conv(den, denc2);

% For the closed-loop transfer function, the following is obtained
% numac and denac used for the overall closed tranfer function in
this case
[numac2, denac2] = cloop(numa2, dena2);
%----PI ends

%----PID Starts
%Ziegler-Nichols parameter computed
Kp3 = 11.327;    %Proportional gain
Ki3 = 1381.34;   %Integral gain
Kd3 = 0.0232;    %Derivative gain
```

UpdatedPPIPID_znj.m (contd.)

```
% For the PID equation
numc3 = [Kd3 Kp3 Ki3 ];
denc3 = [1 0];

% convule "num with numc" and "den with demc"
numa3 = conv(num, numc3);
dena3 = conv(den, denc3);

% For the closed-loop transfer function, the following is obtained
% numac and denac used for the overall closed tranfer function in
this case
[numac3, denac3] = cloop(numa3, dena3);
%----PID ends

% Plotting the new step-response
t = 0:0.00001:0.3;

% New G1 for overall closed loop transfer function
G1 = tf(numac1, denac1);

G2 = tf(numac2, denac2);

G3 = tf(numac3, denac3);

% Plots the Step Response diagram
figure;
hold on
step(G1, t);
hold on
step(G2, t);
hold on
step(G3, t);
legend('P', 'PI', 'PID');
title('Closed Loop PID ZN step response generated for P, PI and
PID combinations');
xlabel('Time, [s]')
ylabel('Voltage, [volts]')
% End of code
```

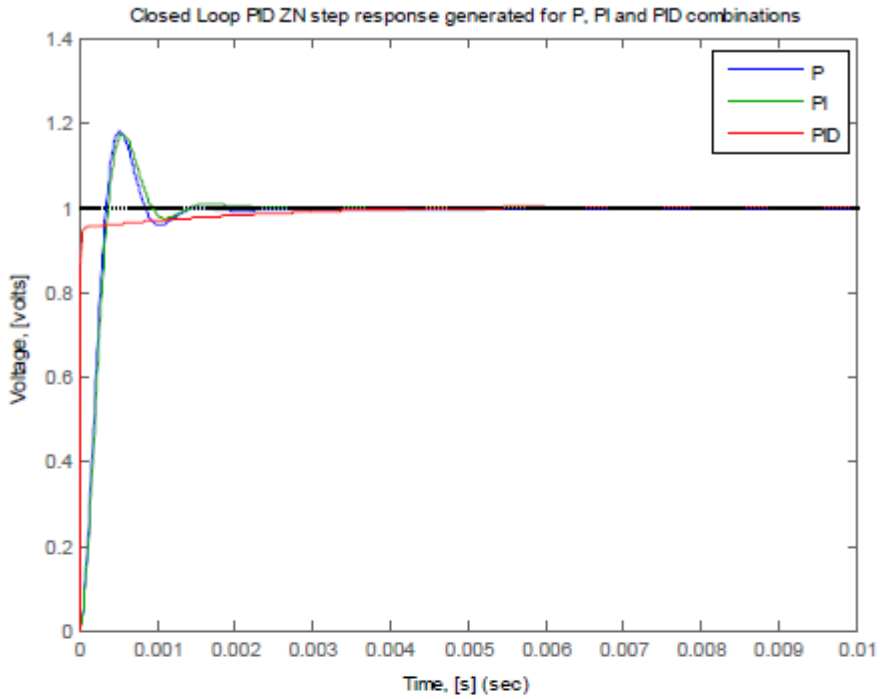

Figure 9.36 – Closed loop PID response for P, PI and PID with t-max=0.01s

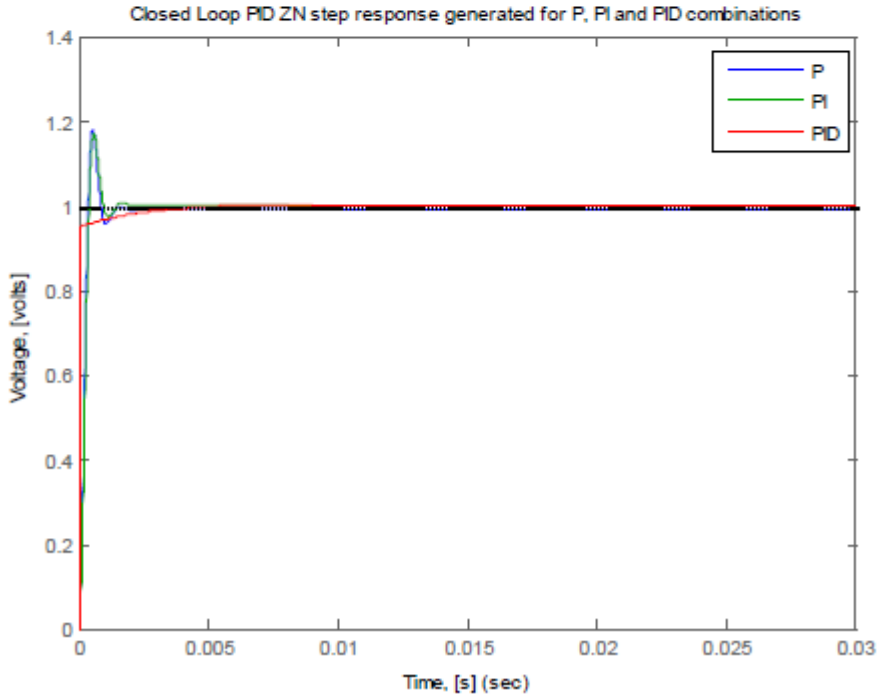

Figure 9.37 – Closed loop PID response for P, PI and PID with t-max=0.03

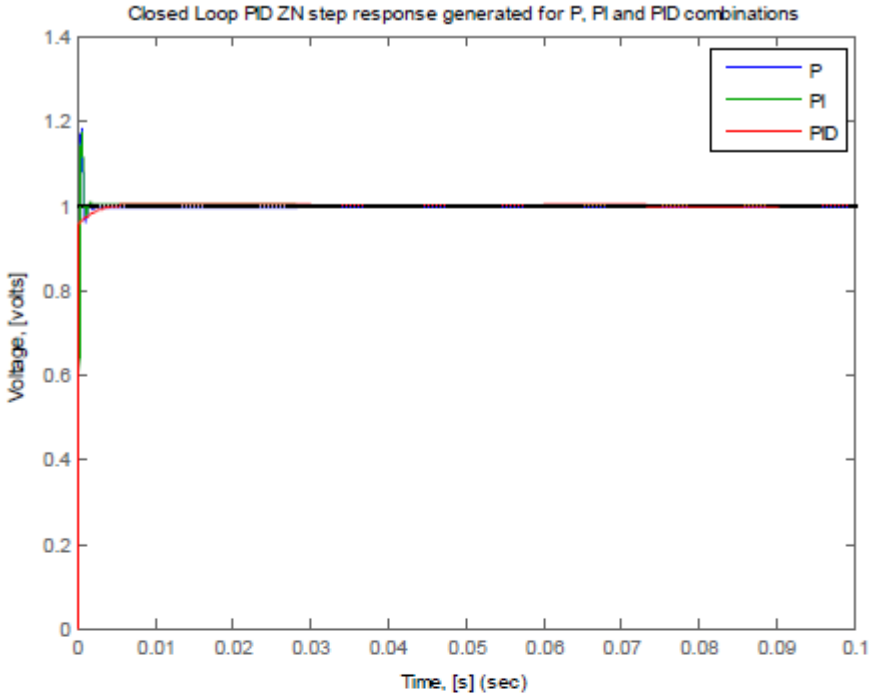

Figure 9.38 – Closed loop PID response for P, PI and PID with t-max=0.1

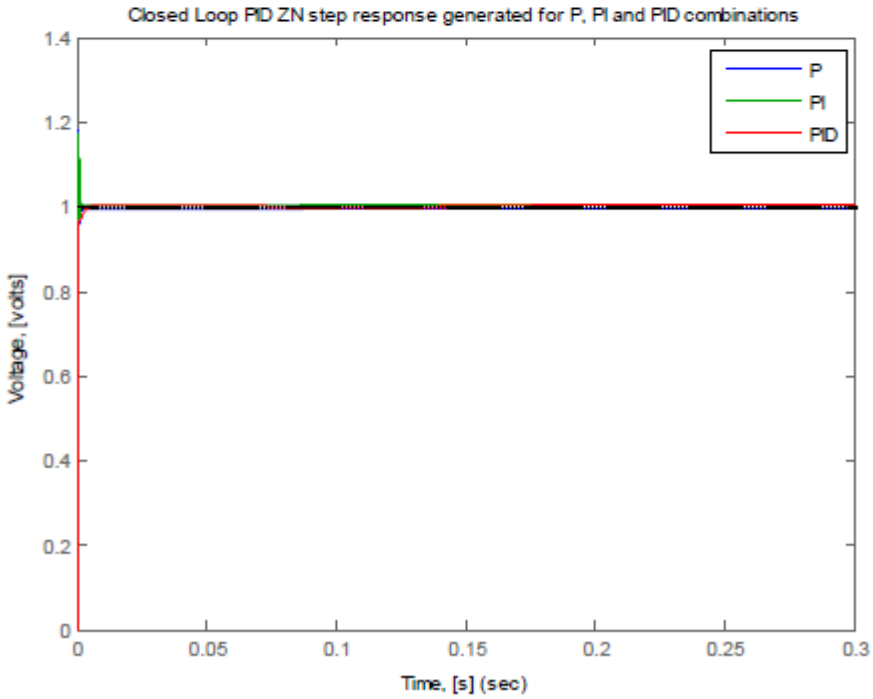

Figure 9.39 – Closed loop PID response for P, PI and PID with t-max=0.3

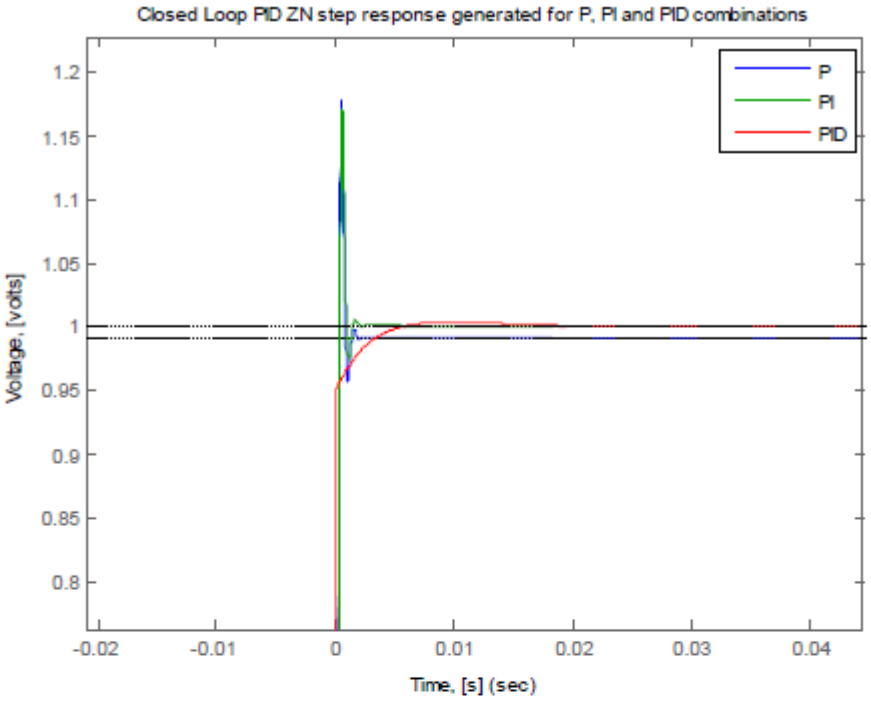

Figure 9.40 – Closed loop PID response for P, PI and PID (1st Zoom)

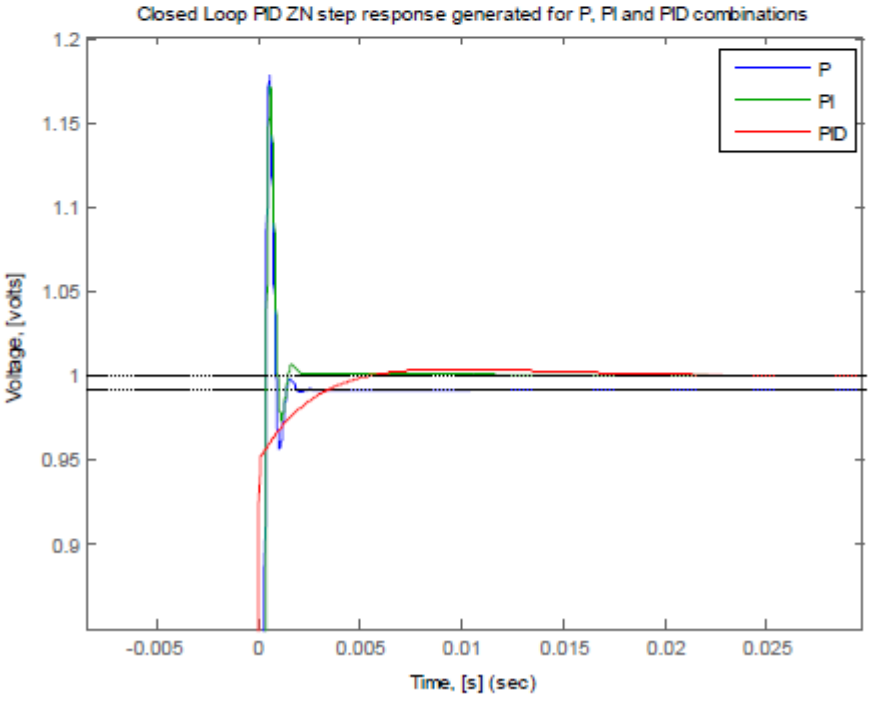

Figure 9.41 – Closed loop PID response for P, PI and PID (2nd Zoom)

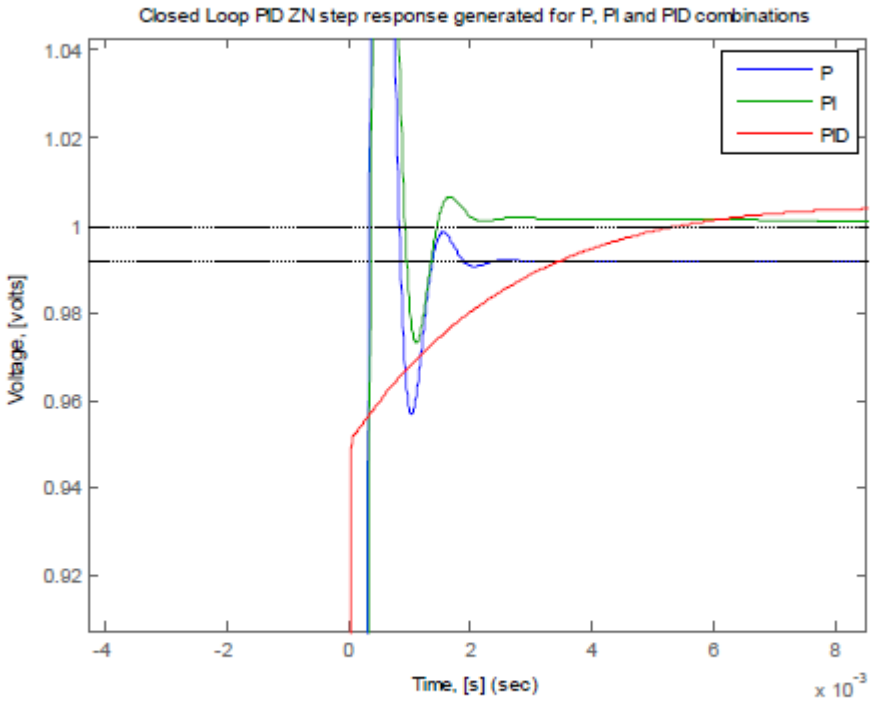

Figure 9.42 – Closed loop PID response for P, PI and PID (3rd Zoom)

9.4 Comparison effects of Trial and Error with Ziegler-Nichols tuning methods

Ziegler-Nichols 튜닝 방법을 이용한 시행 착오의 비교 효과

This is made by creating m-file, UpdatedPID_TErrznj.m for only the PID parameters effects. The generated figure is as shown below in figure--:
이는 m- 파일, PID 매개 변수 효과에 대해서만 UpdatedPID_TErrznj.m을 작성하여 이루어집니다. 생성 된 그림은 아래 그림과 같습니다.

UpdatedPPIPID_TErrznj.m

```
%
% Start of code
%
clear
close all

% includes constant parameters
constants
% includes evaluated constants
evaluatedconstants
num = 1/Ke;
den = [tm*te tm 1];

%Trial and Error PID parameters part
TErrorPID
%Ziegler-Nichols PID parameters part
ZNPIDcomp

% Plotting the new step-response
t = 0:0.00001:0.03;

% New G for overall closed loop transfer function
GZN = tf(numacZN, denacZN);

GTErr = tf(numacTErr, denacTErr);

% Plots the Step Response diagram
figure;
hold on
step(GZN, t);
hold on
step(GTErr, t);

legend('Trial and Error PID', 'Ziegler-Nichols PID');
title('Closed Loop PID for Trial and Error/Ziegler-Nichols step
response output for PID');
xlabel('Time, [s]')
ylabel('Voltage, [volts]')
% End of code
```

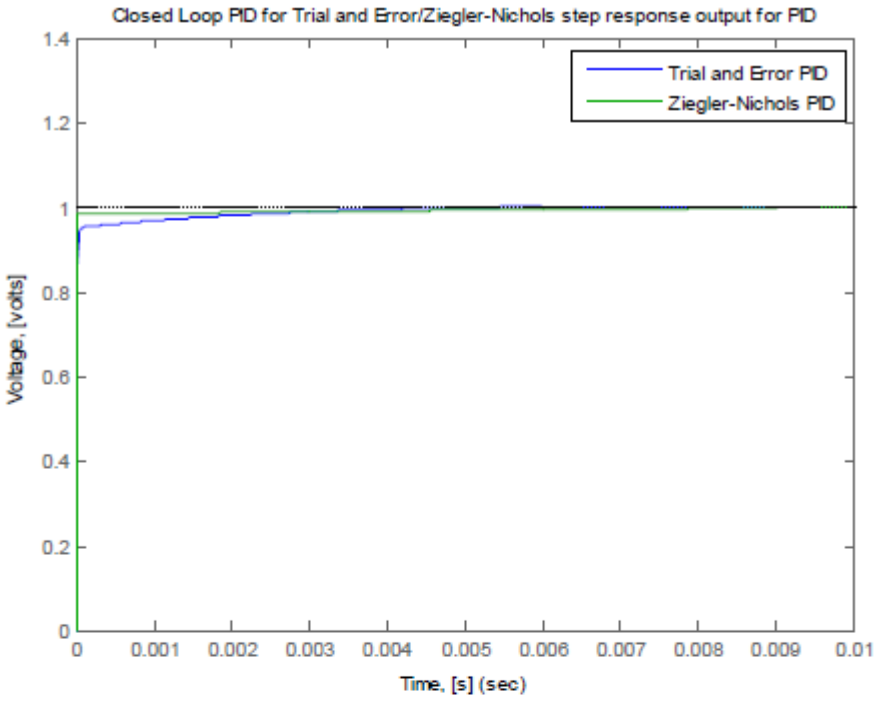

Figure 9.43 – Closed loop response for Trial and Error/Ziegler-Nichols tuning methods

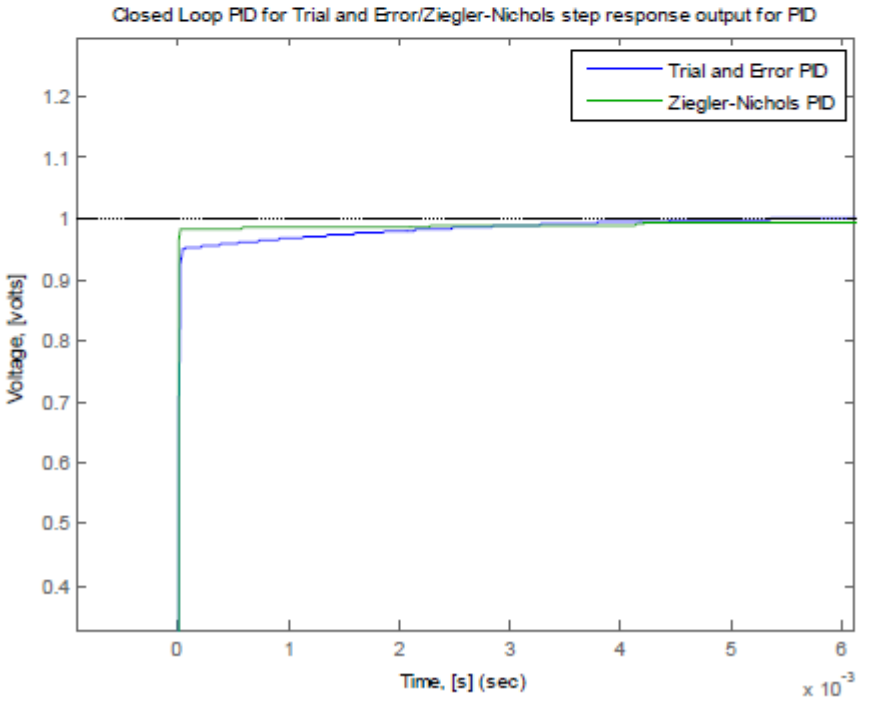

Figure 9.44 – Closed loop response for Trial and Error/Ziegler-Nichols tuning methods (1st zoomed)

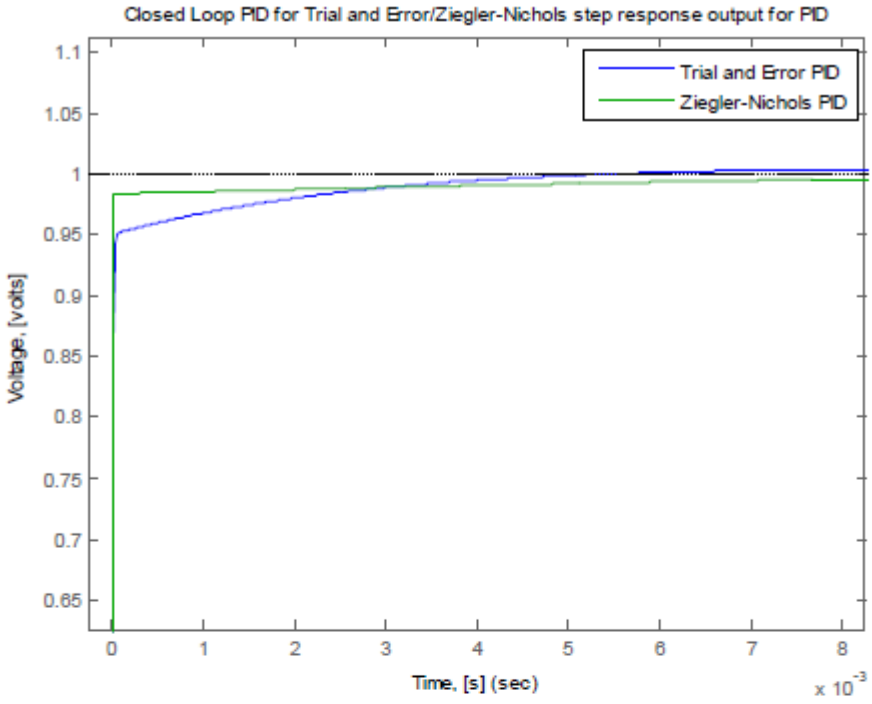

Figure 9.45 – Closed loop response for Trial and Error/Ziegler-Nichols tuning methods (2nd zoomed, right side)

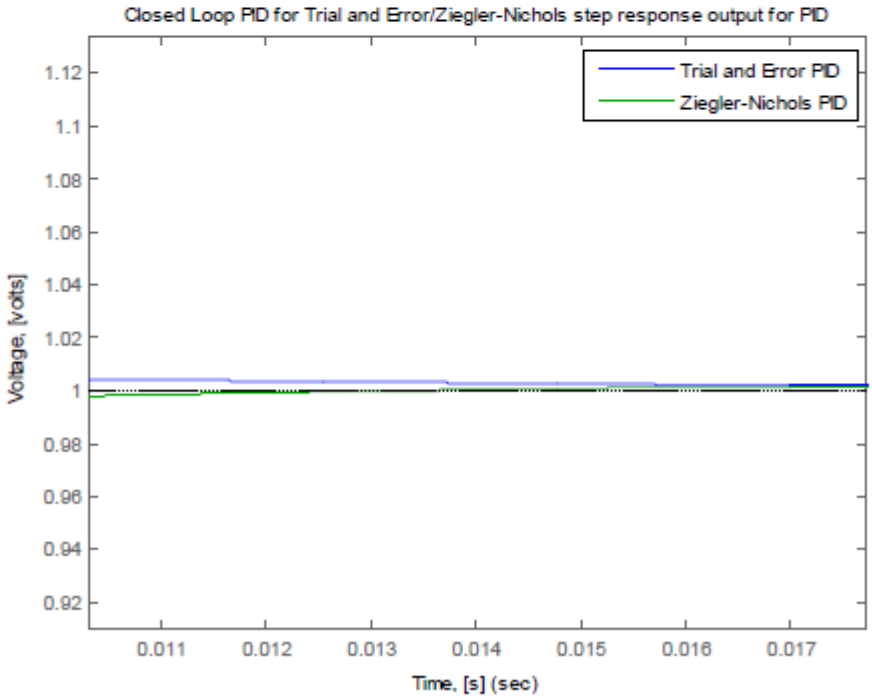

Figure 9.46 – Closed loop response for Trial and Error/Ziegler-Nichols tuning methods (3rd zoomed, left side, with t-max=0.03)

10 FOOTBALL PITCH LAYOUT MODEL

풋볼 피치용 레이아웃 모델

10.1 Dimensions of the Pitch
피치의 크기

The football pitch model serves as the background for the main plot of the trajectory path of the robot wheel.
football 피치 모델은 로봇 휠의 궤도 경로의 주 플롯을위한 배경 역할을합니다.

The basic dimensions (in millimetres) used are given below and are scaled down to metres in the model plot.
사용 된 기본 치수 (밀리미터)는 아래에 주어지며 모델 플롯에서 미터로 축소됩니다.

	Part Label	Dimension (mm)
1.	Length	6050
2.	Width	4050
3.	Centre circle (radius)	500

Table 10.1 – Dimensions of the Football pitch layout model

The figure 10.1 below shows the main target design based on the Laws of the F180 League 2009 [12].
아래의 그림 10.1은 F180 리그 2009의 법칙에 기반한 주요 목표 설계를 보여줍니다 [12].

The needed design for the thesis is the main pitch shown in white lines.
논문에 필요한 디자인은 흰색 선으로 표시된 주요 피치입니다.

Figure 10.1 – Dimension of the standard pitch required [12]

10.2 Football pitch MATLAB design implementation
foorball 피치 MATLAB 디자인 구현

The figure 10.2 shows the full part label of the football pitch layout.
그림 10.2는 football 피치 레이아웃의 전체 부분 레이블을 보여줍니다.

But the main target design is shown in figure 10.1.
그러나 주요 목표 설계는 그림 10.1에 나와 있습니다.

This follows with the m-files used to generate the whole pitch layout robotpatternUpdated.m, newFieldSpec.m, robotBlockpart.m, semiCircleBottomLeft.m, semiCircleBottomRight.m, semiCircleTopLeft.m, semiCircleTopRight.m and testcir2.m (for centre circle plot) used to generate the actual design and shown in figure 10.2.
이것은 전체 피치 레이아웃 robotpatternUpdated.m, newFieldSpec.m, robotBlockpart.m, semiCircleBottomLeft.m, semiCircleBottomRight.m, semiCircleTopLeft.m, semiCircleTopRight.m 및 testcir2.m (center circle plot의 경우)을 생성하는 데 사용되는 m 파일과 같습니다.)는 그림 10.2와 같이 실제 설계를 생성하는 데 사용됩니다.

The output football pitch generated is given under figure 10.3 below.
생성 된 축구 피치는 아래 그림 10.3에 나와 있습니다.

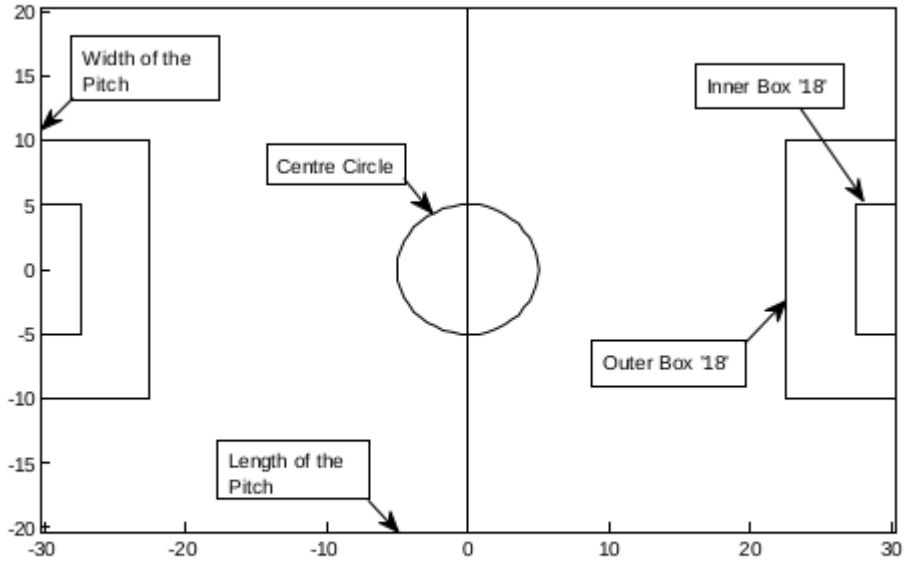

Figure 10.2 – Part label of the Football pitch layout model

testcir2.m

```
%draw circle code
%resolution of plot
t = linspace(0,2*pi,100000);

%assumed centre of the circle (cirX, cirY): sets at origin (0, 0)
cirX=30.25;
cirY=20.25;

%radius of the centre circle, 500mm=5m
r=5;

%circle dual equations
x = r*cos(t)+cirX;
y = r*sin(t)+cirY;

plot(x,y, 'Color','black');
%end of code
```

semiCircleBottomLeft.m

```
%draw circle code
%resolution of plot
t2 = linspace(2*pi, 3*pi/2,100000);
%assumed centre of the circle (cirX, cirY): sets at origin (0, 0)
cirX2=0;
cirY2=18.50;

%radius of the centre circle, 500mm=5m
r=5;

%circle dual equations
x2 = r*cos(t2)+cirX2;
y2 = r*sin(t2)+cirY2;

plot(x2, y2, 'Color', 'black')

%end of code
```

semiCircleBottomRight.m

```
%draw circle code
%resolution of plot
t3 = linspace(pi, 3*pi/2,100000);
%assumed centre of the circle (cirX, cirY): sets at origin (0, 0)
cirX3=60.50;
cirY3=18.50;

%radius of the centre circle, 500mm=5m
r=5;

%circle dual equations
x3 = r*cos(t3)+cirX3;
y3 = r*sin(t3)+cirY3;

plot(x3, y3, 'Color', 'black')

%end of code
```

semiCircleTopLeft.m

```
%draw circle code
%resolution of plot
t1 = linspace(0, pi/2,100000);
%assumed centre of the circle (cirX, cirY): sets at origin (0, 0)
cirX1=0;
cirY1=22.00;

%radius of the centre circle, 500mm=5m
r=5;

%circle dual equations
x1 = r*cos(t1)+cirX1;
y1 = r*sin(t1)+cirY1;

plot(x1, y1, 'Color', 'black')

%end of code
```

semiCircleTopRight.m

```
%draw circle code
%resolution of plot
t4 = linspace(pi/2, pi, 100000);
%assumed centre of the circle (cirX, cirY): sets at origin (0, 0)
cirX4=60.50;
cirY4=22.00;
%radius of the centre circle, 500mm=5m
r=5;
%circle dual equations
x4 = r*cos(t4)+cirX4;
y4 = r*sin(t4)+cirY4;
plot(x4, y4, 'Color', 'black')
%end of code
```

robotpatternUpdated.m

```
%refreshes figures for new ones
clear
close all

%activities needed on the robot field layout

%test plot sample
%-------------------------------------------
%-------------------------------------------
len=100;
robot=zeros(1,len);

%start position
X(1)=-40;
Y(1)=-40;

%move to cordinates
x_goal=0;
y_goal=0;

%
%just something to plot
%this will be for the actual robot movement path
%
for n=1:len
    robot(n)=sin(n)+10;
end
%

%plot robot movement
hold on
plot(robot, 'Color', 'red')

%plot the robot pitch layout
newFieldSpec
```

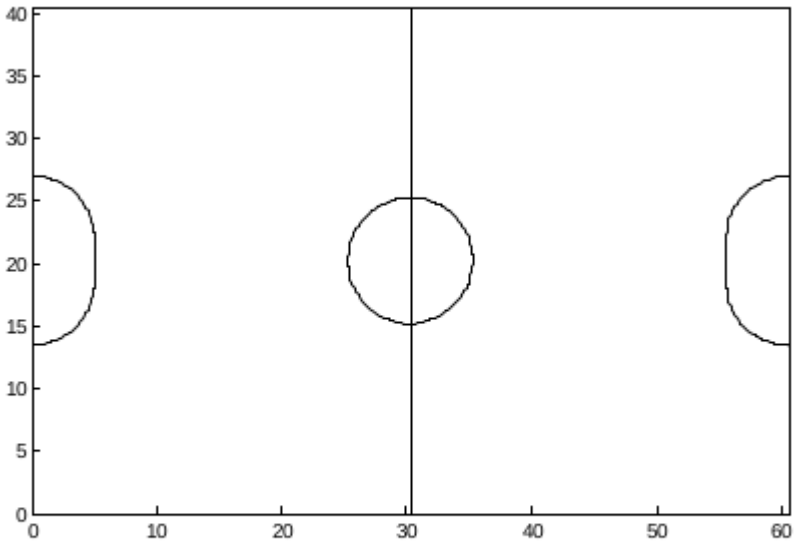

Figure 10.3 – Generated football pitch model using **`robotpatternUpdated.m`**

11 ROBOT 4-WHEEL MOTOR MODEL TRAJECTORY PLANNING 로봇 4륜 모터모델 TRAJECTORY PLANNING

This part involves the cascaded arrangement of all the four wheels with connection to the corresponding system blocks affecting the overall performance of the robot path movement.
이 부분은 로봇 경로 이동의 전반적인 성능에 영향을 미치는 해당 시스템 블록에 대한 연결을 통해 4 개의 바퀴가 모두 계단식으로 배열되어 있습니다.

The block arrangement used is as shown in figure 11.1 below:
사용 된 블록 배열은 그림 11.1 아래 :

After the necessary planning was done, the path simulation would be done on the football model developed with the MATLAB.
필요한 계획이 끝나면 경로 시뮬레이션은 MATLAB으로 개발 된 축구 모델에서 수행됩니다.

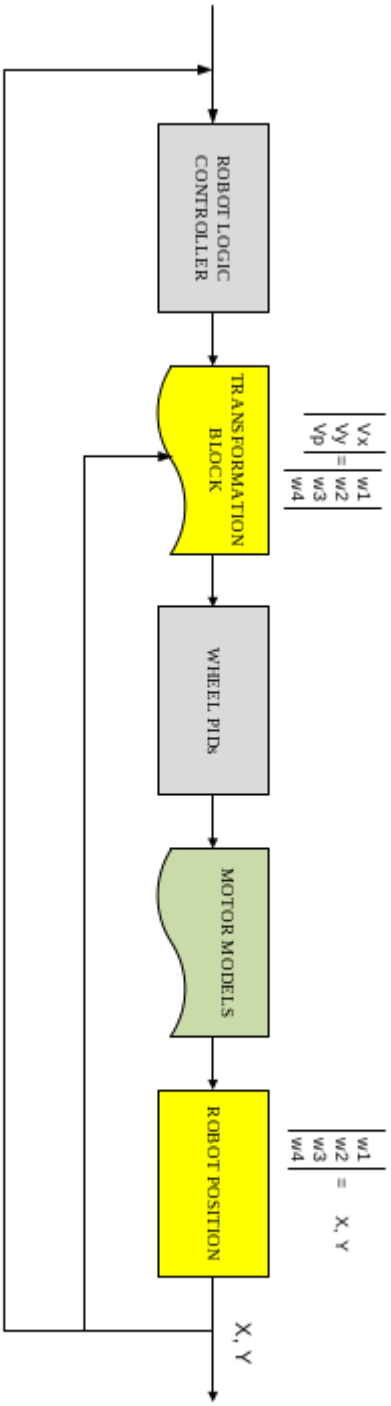

Figure 11.1 – Full robot implementation block

The robot wheels have the following wheel arrangement as shown in figure below figure 11.3 below:
로봇 바퀴는 아래 그림 11.3의 그림과 같이 다음 휠 배열을가집니다.

the wheels motors are in an asymmetrical arrangement;
휠 모터는 비대칭 배열이다.

this is a prototype drawing from the figure 11.2.
이것은 그림 11.2의 원형입니다.

With radian evaluation, the angles p1, p2, p3 and p4 are related to the angle of the wheel axis – 53 O , 53 O , 45 O , and 45 O .
라디안 평가에서 각도 p1, p2, p3 및 p4는 휠 축의 각도 - 53 O, 53 O, 45 O 및 45 O와 관련됩니다.

The whole evaluations as regards this were done in the codes –robotBlockpart.m, robotControllLogic.
이것에 대한 전체 평가는 -robotBlockpart.m, robotControllLogic 코드에서 수행되었습니다.

m, veloToWheel.m, wheelPIDs.m, and speedToXY.m based on figure 11.1 implementation plan.
m, veloToWheel.m, wheelPIDs.m 및 speedToXY.m을 그림 11.1 구현 계획에 따라 계산합니다.

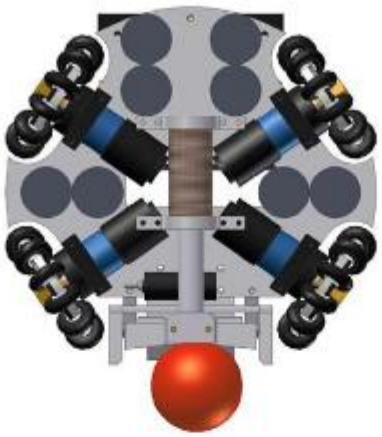

Figure 11.2 – An extract from “Omnidirectional control” [13]

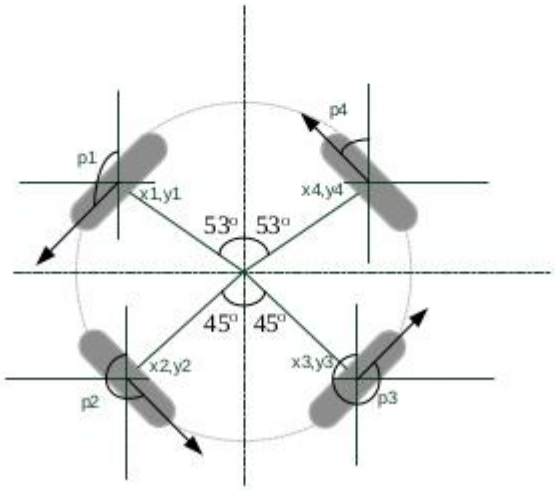

Figure 11.3 – Asymmetrical robot wheel arrangement based on the *Omnidirectional robot control*

robotTrajectoryPlan.m

```
%plots robot trajectory path
clear
close all
len=100;
robot=zeros(1,len);

X=zeros(1,100);Y=zeros(1,100);

%start position
X(1)=25;Y(1)=25;
a1(1)=0;a2(1)=0;a3(1)=0;a4(1)=0;
a1(2)=0;a2(2)=0;a3(2)=0;a4(2)=0;
a1(3)=0;a2(3)=0;a3(3)=0;a4(3)=0;

%move to cordinates
x_goal=0;y_goal=0;

n=1;k=3;
robotGain=0.00001;

%inlcudes motor constants
constants
% includes evaluated constants
evaluatedconstants

num = 1/Ke;
den = [tm*te tm 1];

%Ziegler-Nichols parameter computed
Kp = 11.327;    %Proportional gain
Ki = 1381.34;   %Integral gain
Kd = 0.0232;    %Derivative gain
% For the PID equation
numc = [Kd Kp Ki ];
denc = [1 0];

% convule "num with numc" and "den with demc"
numa = conv(num, numc);
dena = conv(den, denc);

sys = tf(numa,dena,1/1000);

integrationSums=[0, 0, 0, 0];

% robot block parts
robotBlockpart

%plot robot movement
hold on
plot(X,Y);

%plot the robot pitch layout
newFieldSpec
```

robotBlockpart.m

```
while n==1
    %done?
    %robot control logic - BLOCK 1
    [Vx(k),Vy(k),Vp(k)]=robotControlLogic(X(k-1),Y(k-
1),x_goal,y_goal,robotGain);

    %Done
    %transformation block  - BLOCK 2
    %transformation matrix from velocity vector to wheelspeeds
    [w1(k),w2(k),w3(k),w4(k)] = veloToWheel(Vx(k),Vy(k),Vp(k),X(k-
1),Y(k-1));

    %Done
    %Wheel PIDs - BLOCK 3
    %the idividual PID controllers for the wheels including
wheelmotor model
    temp=integrationSums;
    oldArray=[a1(k-1),a2(k-1),a3(k-1),a4(k-1)];
    oldoldArray=[a1(k-2),a2(k-2),a3(k-2),a4(k-2)];

[a1(k),a2(k),a3(k),a4(k),integrationSums]=wheelPIDs(w1(k),w2(k),w3
(k),w4(k),temp,oldArray,oldoldArray);

    %motor model

    %Done
    %robot position - BLOCK 4
    %converts actual wheel motor speed to robot X Y position
    [X(k),Y(k)]=speedToXY(a1(k),a2(k),a3(k),a4(k),X(k-1),Y(k-1));

    %check if close enough to the goal coordinates
    if abs(X(k)-x_goal)<0.1%% && abs(Y(k)-y_goal)<0.1
        n=0;
    end
    if abs(Y(k)-y_goal)<0.1%% && abs(Y(k)-y_goal)<0.1
        n=0;
    end

    if abs(X(k))>41
        n=0;
    end
    if abs(Y(k))>41
        n=0;
    end
    if k>99
        n=0;
    end
    %increment loop index
    k=k+1;
end
```

This will take the main part of the planning and trajectory simulation

robotControlLogic.m

```
function [Vx,Vy,Vp]=robotControlLogic(X,Y,x_goal,y_goal,k)

    Mag_x=x_goal-X;
    Mag_y=y_goal-Y;

    M=sqrt(Mag_x^2+Mag_y^2);

    Vx=k*Mag_x/M;
    Vy=k*Mag_y/M;
    Vp=0;

end
```

wheelPIDs.m

```
function
[a1,a2,a3,a4,intSums]=wheelPIDs(w1,w2,w3,w4,intSumsIn,y_old,y_oldo
ld)
    %actual PIDs here
    %Ziegler-Nichols parameter computed
    Kp = 11.327;    %Proportional gain
    Ki = 1381.34;   %Integral gain
    Kd = 0.0232;    %Derivative gain

    inputs=[w1, w2, w3, w4];

    %certainty problem
    for n=1:4
        PIDin=inputs(n);
        sumIn=intSumsIn(n);

        [y,sumOut]=myPID(PIDin,y_old(n),y_oldold(n),sumIn);

        intSums(n)=sumOut;
        outputs(n)=y;
    end

    a1=outputs(1);
    a2=outputs(2);
    a3=outputs(3);
    a4=outputs(4);
end
```

speedToXY.m

```
function [X,Y]=speedToXY(a1,a2,a3,a4,Xold,Yold)
%Calculate X Y position based on actual wheelspeeds since last
sample
% |W1|     |Vx|
% |W2| -> |Vy|
% |W3|     |Vp|
% |W4|

%Angle of each wheel in Rad, these angles does not change in this
simulation
p1=2.49582083; %143 deg
p2=3.92699082; %225 deg
p3=5.49778714; %315 deg
p4=0.645771823; %37 deg

%Co-ordinates of each wheel in Meter
[x1,x2,x3,x4,y1,y2,y3,y4]=wheelsXYfromXY(Xold,Yold,p1,p2,p3,p4);

  if 1

  %actual wheel speeds...
  W=[a1, a2, a3, a4];
  %transformation matrix
  A=[cos(p1),sin(p1),(-y1*cos(p1)+x1*sin(p1)),1;
       cos(p2),sin(p2),(-y2*cos(p2)+x2*sin(p2)),1;
       cos(p3),sin(p3),(-y3*cos(p3)+x3*sin(p3)),1;
       cos(p4),sin(p4),(-y4*cos(p4)+x4*sin(p4)),1];
  inversA=inv(A);
  else

  %actual wheel speeds…
  W=[a1, a2, a3];
  A=[cos(p1),sin(p1),(-y1*cos(p1)+x1*sin(p1));
       cos(p2),sin(p2),(-y2*cos(p2)+x2*sin(p2));
       cos(p3),sin(p3),(-y3*cos(p3)+x3*sin(p3));
       cos(p4),sin(p4),(-y4*cos(p4)+x4*sin(p4))];
  inversA=inv(A);
  end
  %use the inverse of A here since matrix division is not allowed

  B=W*inversA;

%??
X=Xold+B(1);
Y=Yold+B(2);
%rotation=B3(3); this is not needed if rotation is omitted
end
```

veloToWheel.m

```
function [w1,w2,w3,w4] = veloToWheel(Vx,Vy,Vp,X,Y)
%Calculate the individual wheelspeed based on the three component
%vector velocity of the robot
%|Vx|    |W1|
%|Vy| -> |W2|
%|Vp|    |W3|
%        |W4|
%Desired speed vectors

%Angle of each wheel in Rad
p1=2.49582083; %143 deg
p2=3.92699082; %225 deg
p3=5.49778714; %315 deg
p4=0.645771823; %37 deg

%Co-ordinates of each wheel in Meter
[x1,x2,x3,x4,y1,y2,y3,y4]=wheelsXYfromXY(X,Y,p1,p2,p3,p4);

if 0
    %Wheel 1
    x1=0.0677;y1=0.0511;

    %Wheel 2
    x2=-0.0599;y2=0.0596;

    %Wheel 3
    x3=-0.0599;y3=-0.0596;

    %Wheel 4
    x4=0.0677;y4=-0.0511;
end

%Transformation Matrix
A=[cos(p1),sin(p1),(-y1*cos(p1)+x1*sin(p1));
   cos(p2),sin(p2),(-y2*cos(p2)+x2*sin(p2));
   cos(p3),sin(p3),(-y3*cos(p3)+x3*sin(p3));
   cos(p4),sin(p4),(-y4*cos(p4)+x4*sin(p4));];

%3 component vector matrix
B=[Vx;Vy;Vp];
W=(A*B); %Matrix solution giving result for velocity of each wheel
w1=W(1);
w2=W(2);
w3=W(3);
w4=W(4);
end
```

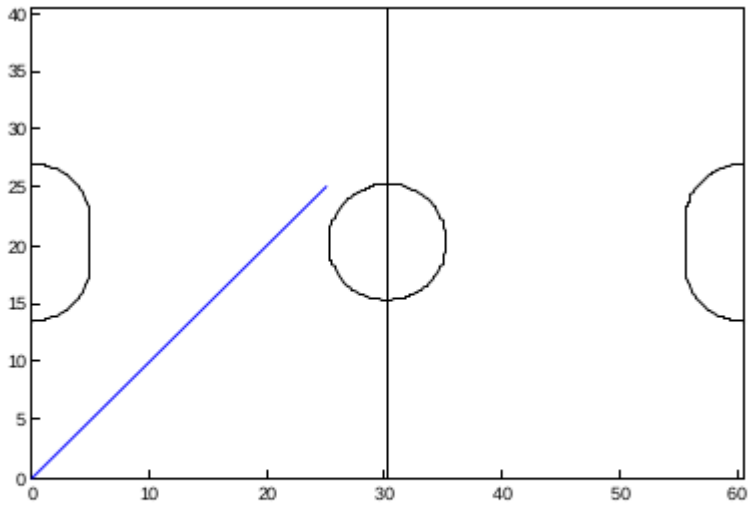

Figure 11.4 – The output of the robot path plotting

The blue line in figure 11.4 shows the planned path of the robot trajectory.

12 CONCLUSION, CHALLENGES AND RECOMMENDATION
결론, 도전 과제 및 권장 사항

12.1 Conclusion
결론

In this work, the PID controller was used as a vital technical tool used in system modelling and control.
이 작업에서 PID 컨트롤러는 시스템 모델링 및 제어에 사용되는 핵심 기술 도구로 사용되었습니다.

It started with the analysis and reasons why an absolute précised control is important in drives particularly the BLDC motors and then the mathematical modelling.
그것은 분석과 이유에서 절대적 정밀 제어는 드라이브, 특히 BLDC 모터와 수학적 모델링에서 중요합니다.

Also, the use of the MATLAB ® /SIMULINK ® to develop
the robot football pitch model and the trajectory planning were additional parts to this work.
또한 MATLAB® / SIMULINK®를 사용하여 로봇 축구 피치 모델과 궤적 계획은이 작업의 추가 부분이었습니다.

12.2 Challenges
도전 과제

Some of challenges faced include:
당면한 과제 중 일부는 다음과 같습니다.
the intensive study of MATLAB ® /SIMULINK ® and the various modelling techniques.
MATLAB® / SIMULINK® 및 다양한 모델링 기술에 대한 집중적 인 연구.

More also, the knowledge of mathematical methods was needed to enhanced my modelling ability in this thesis as it required more mathematical skills.
또한 더 많은 수학적 기술이 요구됨에 따라이 논문에서 모델링 능력을 향상시키기 위해서는 수학적 방법에 대한 지식이 필요했습니다.

In additional, some of areas of control systems engineering had to be studied to have a blend of understanding in the areas of system stability.
또한, 제어 시스템 엔지니어링 분야 중 일부는 시스템 안정성 분야에 대한 이해가 혼합되어있는 것으로 연구되어야했습니다.

And one major challenging part was the aspect of model the path of the robot from one point to another.
그리고 한 가지 중요한 도전 과제는 한 지점에서 다른 지점으로 로봇의 경로를 모델링하는 측면이었습니다.

This part required some advanced mathematical skills which could not be implemented.
이 부분에는 구현할 수없는 몇 가지 고급 수학 기술이 필요했습니다.
A straight path was gotten in the trajectory simulation.
탄도 시뮬레이션에서 직선 경로를 얻었습니다.

12.3 Recommendations – Possible improvement
권장 사항 - 가능한 개선

This work could be improved by incorporating the hardware testing and possible laboratory testing.
이 작업은 하드웨어 테스트 및 가능한 실험실 테스트를 통합하여 향상시킬 수 있습니다.

Also, to have a more précised PID parameters, new methods of
PID tuning (the use of genetic algorithms) could be employed for optimal values.
또한보다 정밀한 PID 매개 변수를 가지려면 PID 튜닝 (유전 알고리즘의 사용)은 최적의 값으로 사용될 수 있습니다.

In addition to the use of PID controller, another instance of Single-Input-Single- Output (SISO) could be used under the MATLAB versatile toolbox.
PID 컨트롤러 사용 외에도 MATLAB 다기능 도구 상자에서 SISO (Single-Input-Single-Output)의 또 다른 인스턴스를 사용할 수 있습니다.

More also, the real testing and program implementation of the BLDC motor could be harnessed by using the MATLAB ® /SIMULINK ® utilities and being able to incorporate C-programming with the microcontroller.
또한 BLDC 모터의 실제 테스트 및 프로그램 구현은 MATLAB® / SIMULINK® 유틸리티를 사용하고 마이크로 컨트롤러와 함께 C 프로그래밍을 통합함으로써 가능합니다.

And more technical resources should be available to the student for proper execution of the work.

또한 학생들이 적절한 업무 수행을 위해 더 많은 기술 자료를 이용할 수 있어야합니다.